WAVESTONE

Penetration testing course - ISEP

15th January 2024

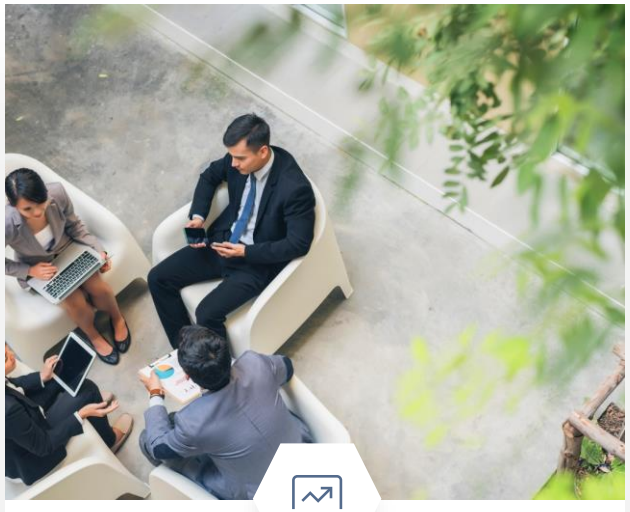
isep
École d'ingénieurs du numérique

Agenda

INTRODUCTION

Our mission is to partner with the world's largest enterprises on their most critical transformations

Competitive market challenges



Digital-first culture

Environmental emergency

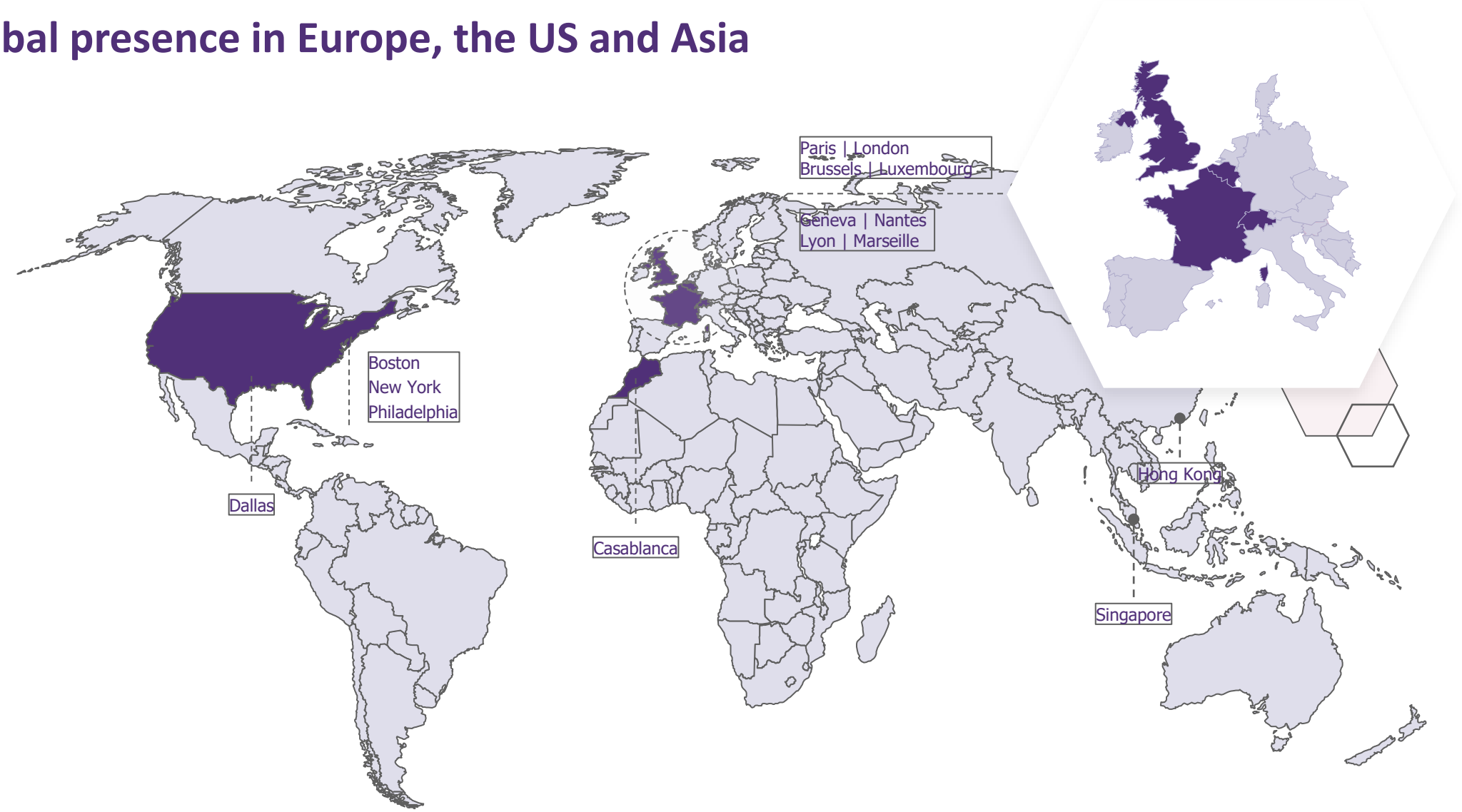
Key Sectors

- Financial Services
- Manufacturing & Life science
- Consumer Goods, Retail & Luxury
- Energy & Utilities
- Transportation & Services
- Government & International Institutions

Domains of Expertise

- Customer Experience & Service Design
- Finance, Performance & Procurement
- Industry 4.0 & IoT
- Data, Analytics & AI
- IT Strategy & CTO Advisory
- Operating Model Design & Agility
- Supply chain
- Sourcing & Services Optimization
- Cybersecurity
- Sustainability

A global presence in Europe, the US and Asia



Wavestone has a workforce of nearly 4,000 employees worldwide.

Win the **digital race** with **digital trust**

Digital trust is a **key business enabler** that will put you ahead to win the **digital transformation** race

900+
Consultants & Experts

1,000+
Engagements per year in **20+** countries

Our clients
Board, Business, CDO, CIO, CISO, BCM, DPO

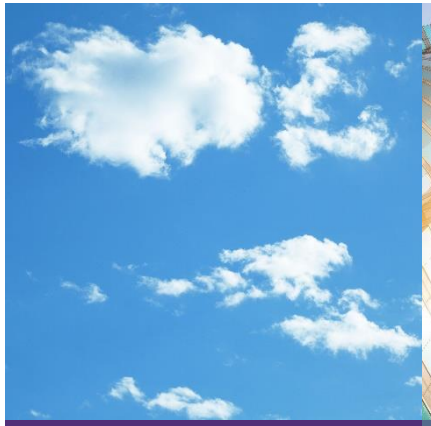
PROVEN EXPERTISE

- Digital Risk Strategy & Compliance
- Safe Business Transformation
- Security Design & Program Management
- Identity, Fraud & Trust Services
- Penetration Testing & Incident Response
- Business Continuity & Resilience
- Industrial Control Systems

ACTIONABLE INSIGHTS

- Industry-specific risk mapping
- AMT Master plan methodology
- Start-ups & CISO Radars
- ICS-Attacks demonstrator
- CERT-W & Bug Bounty

We look far ahead to address client-focussed topics



CLOUD

Deployment is a reality, adopt a new security model to take it into account

CRITICAL INFRASTRUCTURE (NIS/LPM)

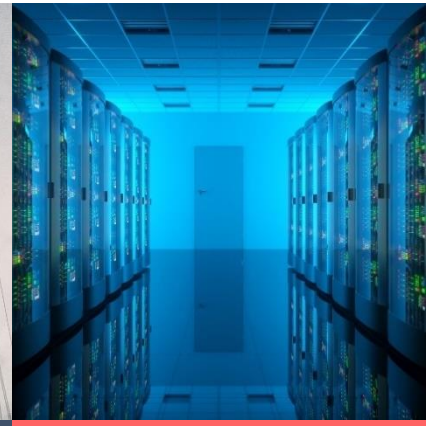
Find the appropriate perimeter to limit impact whilst ensuring national security

DATA PRIVACY (GDPR)

As compliance pressure increases, prioritise governance, Privacy By Design processes and incident notification handling

INDUSTRIAL CONTROL SYSTEM

A breakthrough in industrialised protection systems with new secure and certified solutions to assess



CYBER RESILIENCE

Include a scenario of IS-wide logical destruction in your BCP

Succeed in your **digital transformation** thanks to **digital trust**

A pool of more than 70 experienced auditors combining technical skills and consulting expertise to assist you in a pragmatic manner and address the business challenges

70+ Technical auditors

900+ Cybersecurity consultants

450+ Audits performed annually

Audit assignments and penetration tests performed in all business sectors **since 2005**

BANKING | INSURANCE | INDUSTRY | SERVICES | ENERGY | PUBLIC SECTOR

A wide coverage in terms of audit types

- Organisational and technical audits, penetration tests, code review, configuration review, physical security assessment

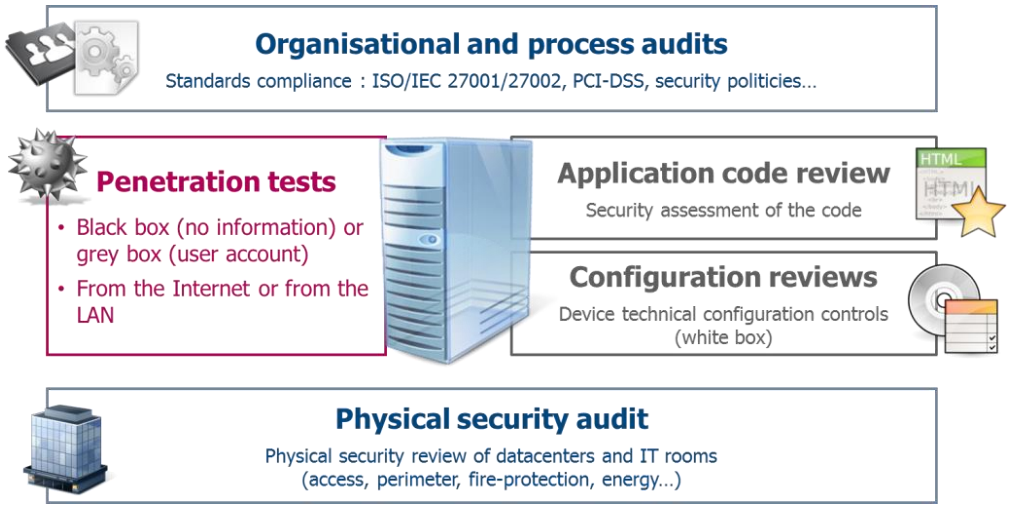
Prestigious references

A recognized expertise

- A "security audit and penetration tests" offer with **ISO 27001 / SMSI certification** and a **PASSI RGS / LPM qualification**
- An approach relying on well-known international norms (ISO2700x, EBIOS, OWASP, etc…)
- A business-risk oriented assessment
- A wide range of technical expertise : ICS, hardware hacking, IoT, AD security, Red Team …
- Internal toolbox and templates development

Our audit and penetration testing offer

A capacity to cover all security audit scope



An acknowledged auditor know-how

« An audit pool »: a team of experimented consultants (~40) dedicated to security audit and penetration tests activities

- Deontological ethics and professionalism
- Technical expertise on IT subjects
- A business risk-oriented security assessment

WAVESTONE
Prestations d'audit de sécurité et d'investigation numérique certifiées NF ISO/IEC 27001:2013
LSTI
Certificat n°18

A pragmatic methodology

Based on standards, methodologies, internationally approved approaches (ISO2700X, EBIOS, OWASP…)

- Development of environment specific toolboxes
- Definition of standard documents: interview and audit guides, reports
- An offer "security audits and penetration testing" ISO 27001 certified & PASSI qualified

Some prestigious credentials

More than **300 audits performed every year** allowing us to build **relevant sectoral benchmarks**

Finance

Industry

Services

Administration

An acknowledged and shared audit expertise

Developers

- **EDRSandblast :** tool to bypass EDR mechanisms
- **OPCUA-scan:** scanning tool for the ICS protocol OPCUA
- **Invoke-CleverSpray** Password Spraying Script detecting current and previous passwords of Active Directory User
- **Hadoop Attack Library:** A collection of pentest tools and resources targeting Hadoop environments

Writers

- **MISC**
  - *N°99 : Stealthy communication techniques with **Empire***
  - *N°96 : **PowervVew** or how to become domain Admin faster*
  - *N°82 : Introduction to **Burp** extensions development*
  - *N° 77 : « Let's hook » with **JavaSnoop**!*
  - *N°74 : Intrusion tests on industrial PLC*
- **RiskInsight blog**

Talkers

- **Abaddon, the redteam angel:** *2020-RSA Conference*
- **Pentesting Active Directory:** *B2018-Bsides Lisbon*
- **Hadoop Safari:** *2017-Bsides Las Vegas, HITB Singapour, PHDays, 2016-Zeronights*
- **ICS: pentesting PLCs 101:** *2016/18/22-DEFCON, 2017-Brucon, 2015-Bsides Las Vegas*
- **CI/CD the new Eldorado:** *2022-DEFCON, Bsides Las Vegas*
- **EDR detection mechanisms and bypass techniques:** *2022-DEFCON*

Enthusiasts

- **Events sponsoring**
  - **Le Hack**
  - **GreHack**
  - **SIGSEGv1 (RTFM)**

… and teachers

Courses, seminars about Information Security
… and trainings tackling **ICS, Mobile, or Web** intrusion tests

/ **02**

Web Application Security

What does Web Application mean?

Institutional website

Mark image

Online store

Financial loss
Fraud
Confidentiality issues

Internal application

Business trouble
People safety

What are the technical risk?

Information disclosure

The attacker obtains specific information on used technologies and data treatment

Session or identity theft

The attacker takes control of an applicative account without the user knowing

Unauthorized resource access

The attacker gains access to protected data and can create, modify, read or delete them

Privileges escalation

The attacker modify the access role of an account to access sensitive data or privileged functionalities

Denial of Service

The attacker awakened the company capabilities of assuring its service

Defacement

The attacker modifies the visual aspect of the Website

Server compromise

The attacker obtains administrator access to the server

Legal issues

Personal information are not correctly protected regarding to the law

Penetration testing: Our vision of the practice

3 different types of investigation

Black box

No information other than the IP address or the URL is given to the pentester

Grey box

Tests accounts credentials are given to the pentester in order to test privilege escalation and partitioning

White box

Privileged access to the perimeter is given to the pentester: access to admin, etc.

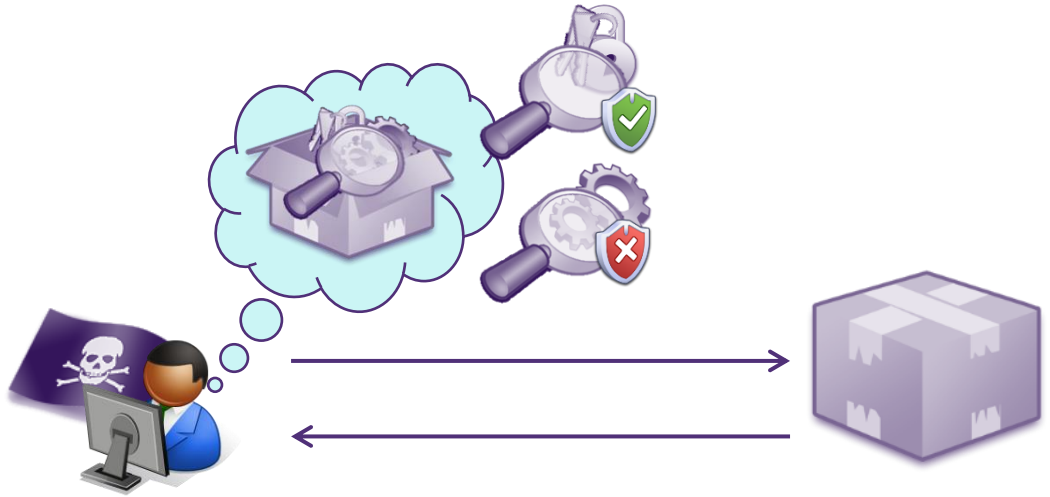
Connected types of investigation: *organizational review, configuration review, architecture review, code review*

Penetration testing: Our vision of the practice

Our penetration tests aim above all to **understand** how the target works in order to **deduce** and **exploit** security flaws

Following our philosophy:

- The **auditor does not perform** a set of tests to verify whether or not a type of vulnerability is present
- The auditor interacts with the target (a black box) **to understand its mechanisms** and, after analysis, discover relevant and exploitable flaws



Wavestone security auditor

The target

The evaluation of the security level is not limited to a battery of tests, but is based on **a behavioral analysis** of the target and its security mechanisms

Our practice of penetration testing **cannot be automated**: our philosophy leads us to follow a **manual approach with tools**

How does a Web Application work?

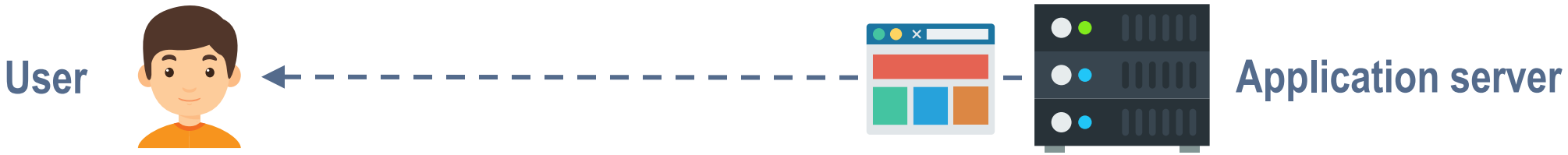
When the user **enters a URL or clicks on a webpage element** a **GET or POST request** containing some **parameters** is sent

```
GET /page.php?param=toto
```

```
POST /page.php

param=toto
```

How does a Web Application work?

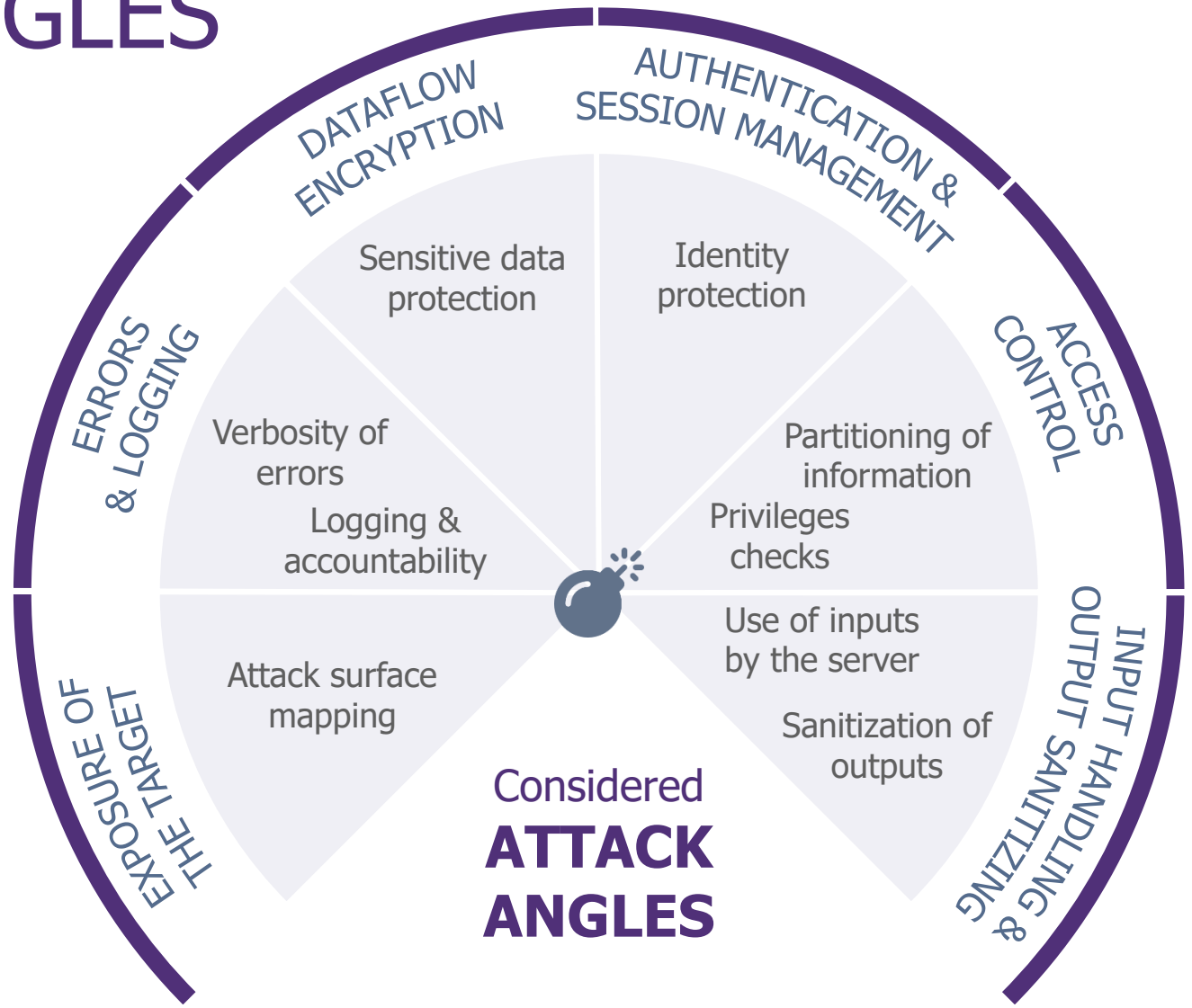


The server then **responds** with the **content of the new page**

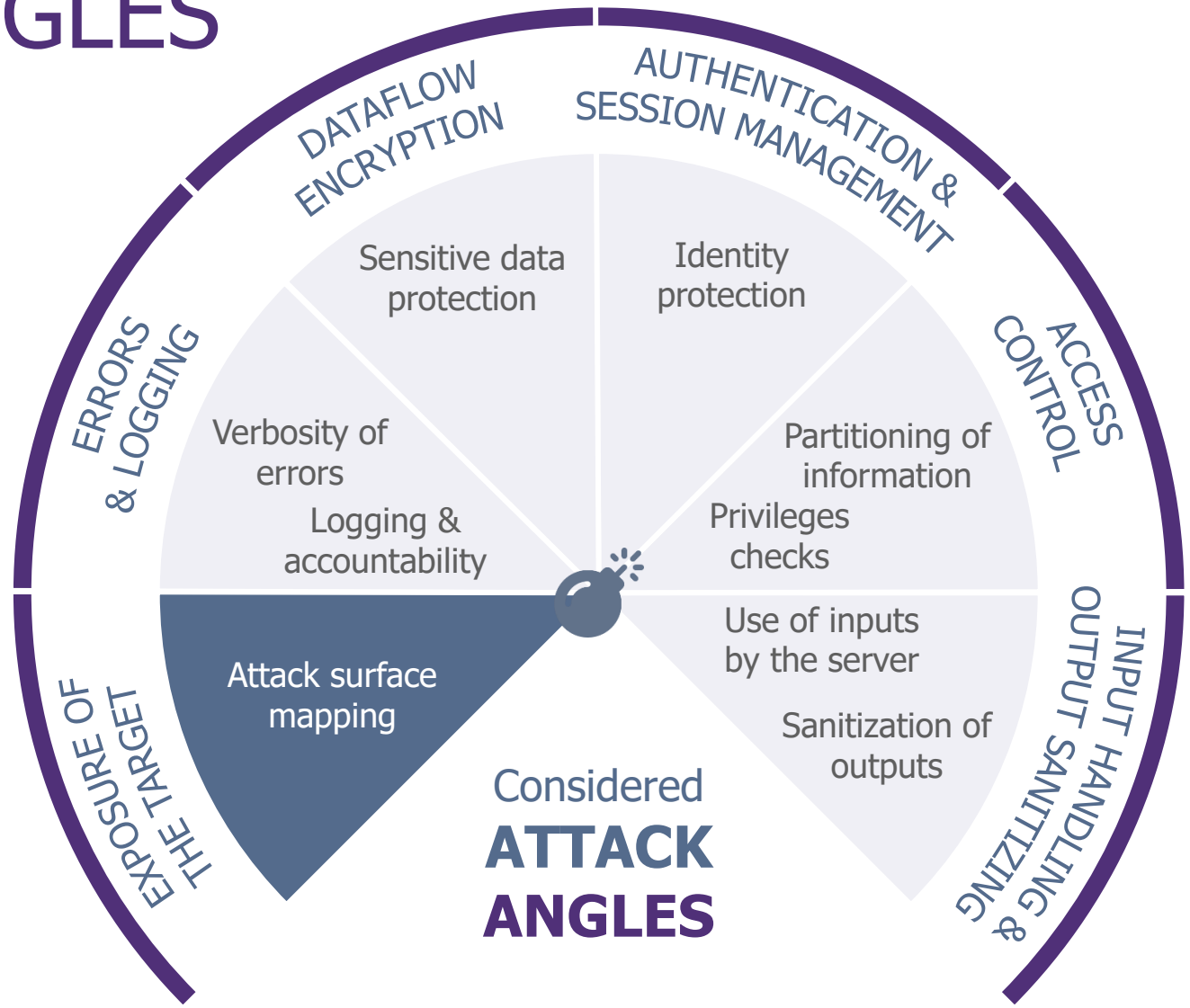
/ 03

Web Penetration Testing

ATTACK ANGLES



ATTACK ANGLES



INFORMATION EXPOSURE: Presentation

DEFINITION

Information exposure arises when **sensible information** is accessible by an **unauthenticated or unprivileged user**

Some information are particularly sensible, such as **personal information** which are strongly protected by the law

VULNERABLE CASES

Technical information regarding the application architecture contains the **password of the administrator account** of the Webserver

A **confidential document** is referenced and accessible using a **search engine**

Personal information of other users is accessible to any user of the application

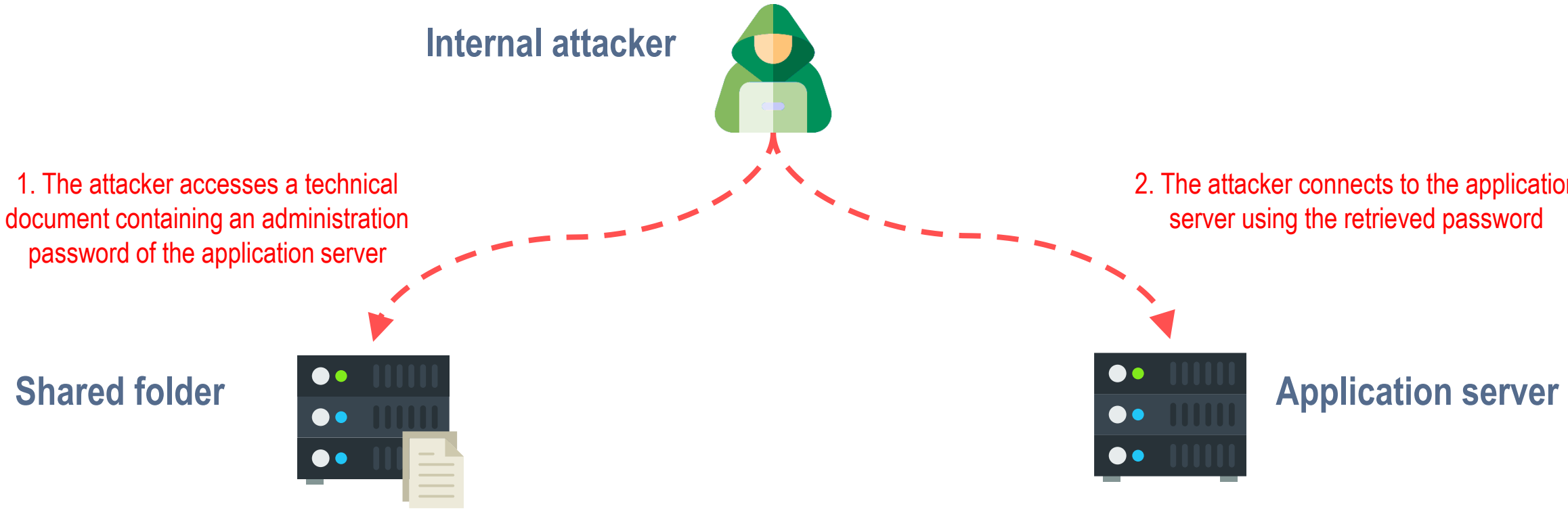
What risks if not or poorly done?

1 **Information disclosure:**
Technical documents accessible from the Internet permit an external attacker to better understand the application

2 **Unauthorized resource access:**
An attacker can directly access sensible information

3 **Legal issues:**
The noncompliance of data protection regarding laws or regulations such as the EU GDPR or local privacy laws can result in large monetary fines

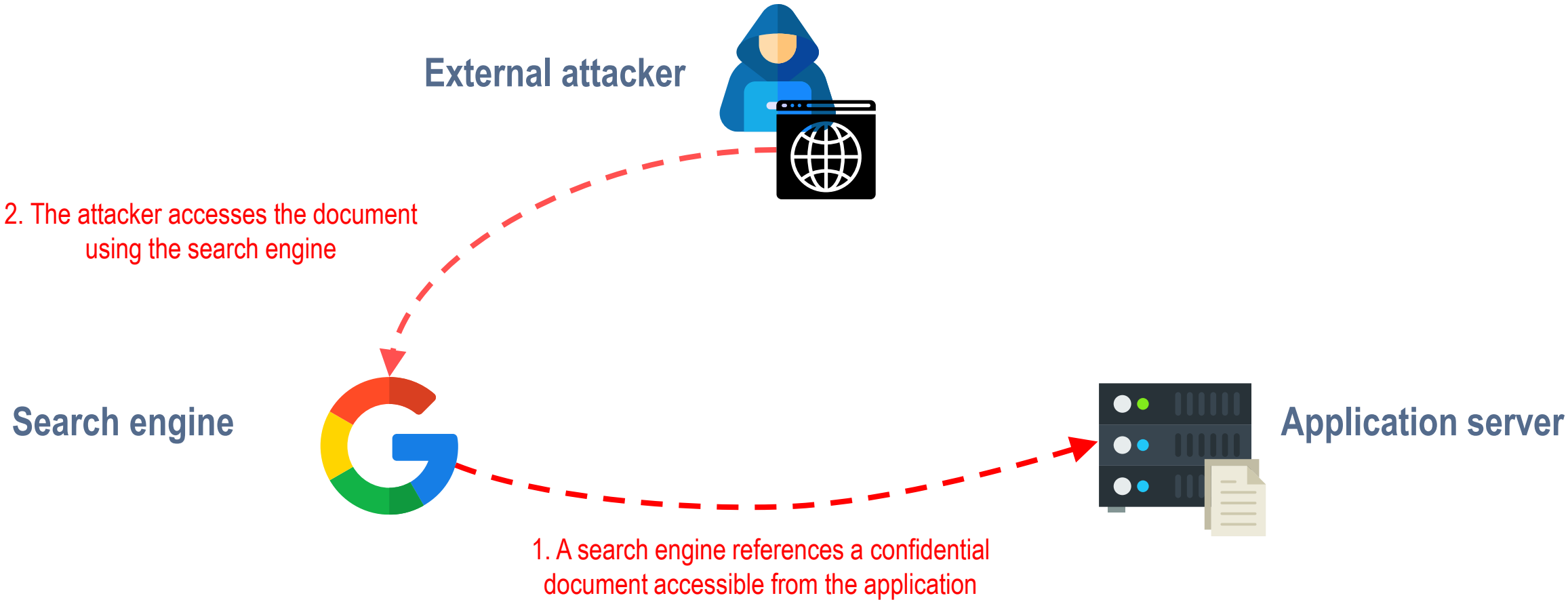
INFORMATION EXPOSURE: An attack scenario example



Scenario 1: obtention of an administration password stored in a technical document

Scenario 2: confidential documents recuperation

INFORMATION EXPOSURE: An attack scenario example



Scenario 1: obtention of an administration password stored in a technical document

Scenario 2: confidential documents recuperation

INFORMATION EXPOSURE: Testing methodology

TOOLS & METHODOLOGY

Search engine (aka Google):

→ **Use of google dorks:** find confidential documents or information referenced by the search engine

→ **Use of google cache :** access previous versions of the application where security changes where not already made

EXAMPLES

- site:site.com
- site:site.com filetype:txt txt (pdf, sql, sav, back, bak, doc, docx, xls, xlsx, etc.)
- site:site.com filetype:php php (jsp, aspx, asp, etc.)
- inurl:site.com
- site:site.com password (admin, username, login)
- http://webcache.googleusercontent.com/search?q=cache:site.com

Complementary resources:
https://www.exploit-db.com/google-hacking-database/

INFORMATION EXPOSURE: Prevention

- Do not reference sensitive data on search engines
- Do not include accounts information in technical documentation
- Protect personal information according to the law

SECURITY CONFIGURATION: Presentation

DEFINITION

- **Security misconfiguration** can happen at **any level** of an application stack
- **Illegitimate services exposure, lack of patch management, use of trivial or default password** are some example of such misconfiguration

VULNERABLE CASES

- An administration service (such as SSH) is **exposed over the Internet**
- The software is **out of date or vulnerable**
- **Illegitimate HTTP methods** (TRACE, PUT, DELETE) are accepted by the server
- **Default accounts and their passwords** are used

What risks if not or poorly done?

1

Unauthorized resource access:
The lack of hardening can provide access to default pages, services or features

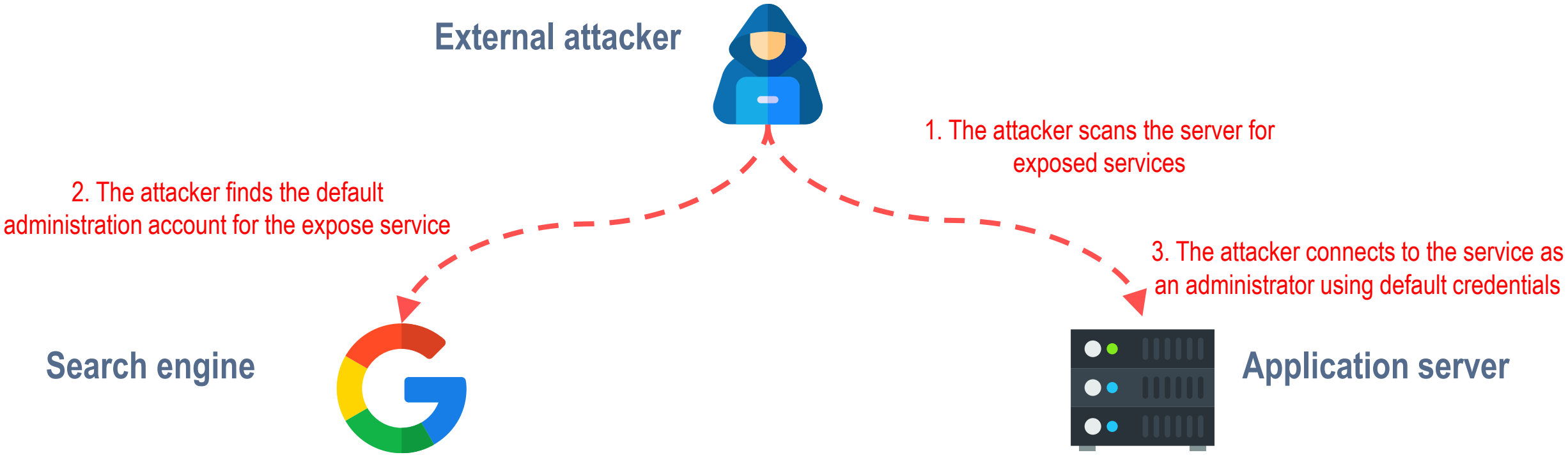
2

Server compromise:
Access to administration services can provide a full control over the server

3

Denial of Service:
Many known vulnerabilities on software and services permit a Denial of Service

SECURITY CONFIGURATION: An attack scenario example



Scenario 1: default account usage on administration service

Scenario 2: use of out-of-date service

SECURITY CONFIGURATION: Testing methodology

TOOLS & METHODOLOGY

NMAP:

→ Free and open source utility for **network discovery and security auditing**

→ Many **options** and **scripts** exists for this tool

```
root @ kali > nmap google.com

Starting Nmap 7.60 ( https://nmap.org ) at 2018-08-17 17:25 CEST
Nmap scan report for google.com (172.217.18.206)
Host is up (0.35s latency).
Other addresses for google.com (not scanned): 2a00:1450:4007:805::200e
rDNS record for 172.217.18.206: ham02s14-in-f206.1e100.net
Not shown: 999 filtered ports
PORT    STATE SERVICE
443/tcp open  https

Nmap done: 1 IP address (1 host up) scanned in 84.60 seconds
```

EXAMPLES

\ Default command:

→ `Nmap –p - -A –T4 site.com`

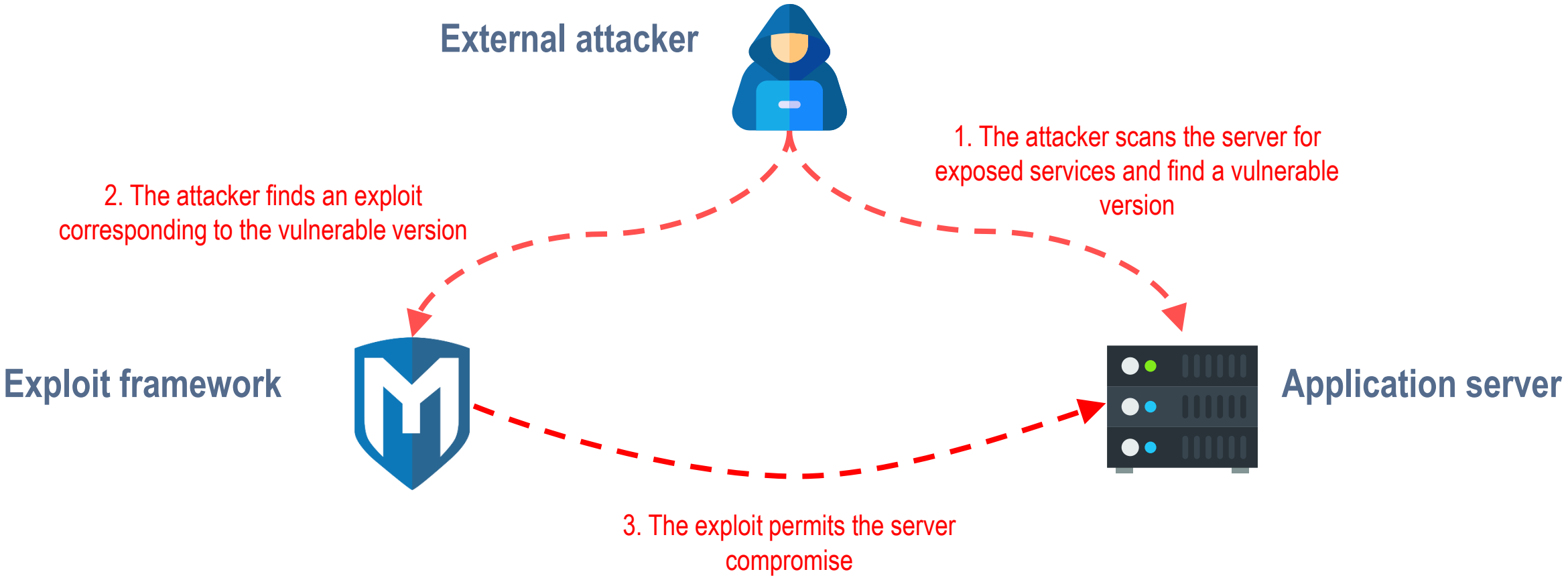
→ `"-p –" means all ports (1000 most used ports by default)`

→ `"-A" indicates usage of OS detection (not 100% reliable), version detection, script scanning and traceroute`

→ `"-T4" is used to increase scan speed`

Complementary resources:
https://nmap.org/
https://nmap.org/book/man-briefoptions.html

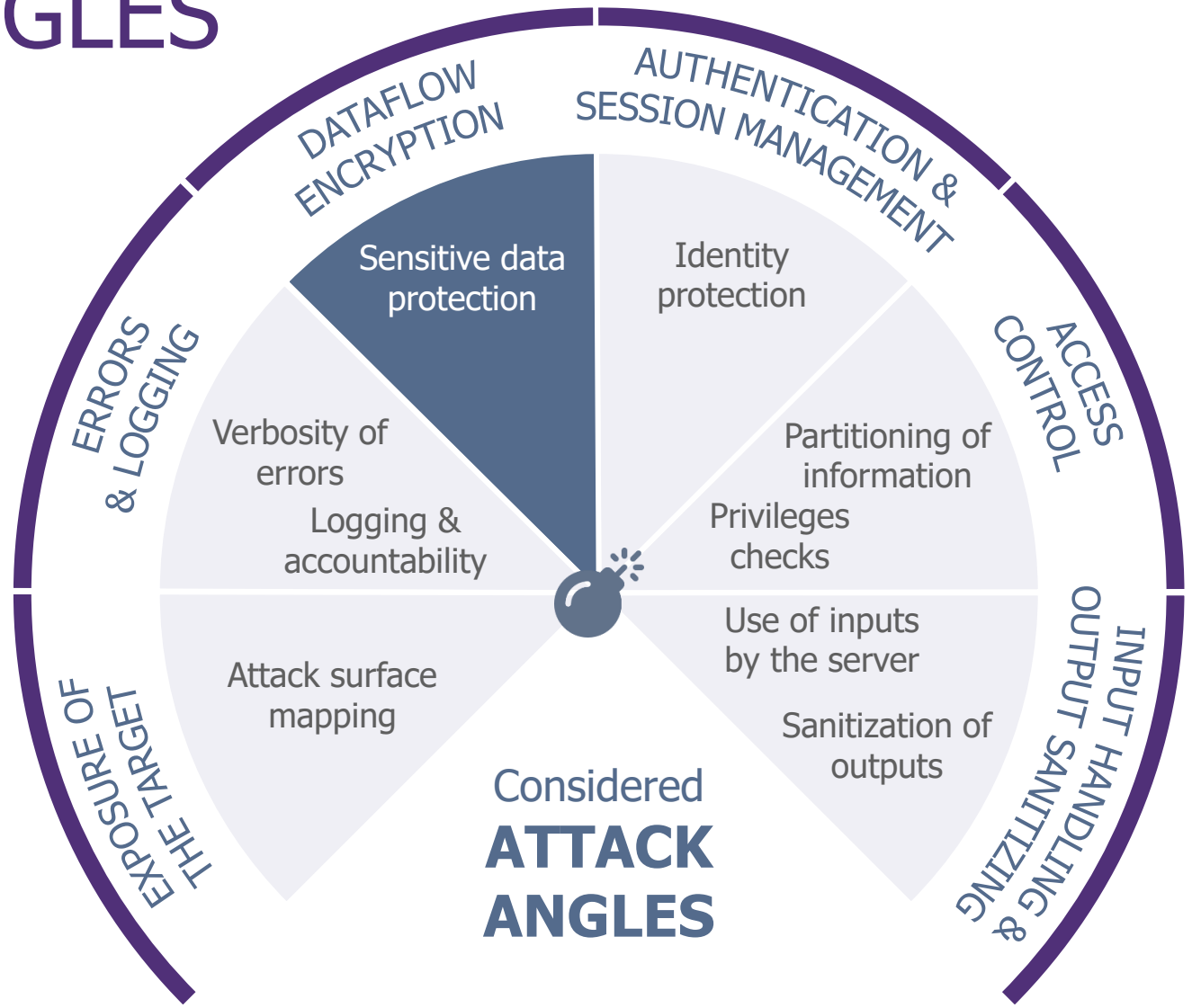
SECURITY CONFIGURATION: An attack scenario



Scenario 1: default account usage on administration service

Scenario 2: use of out of date service

ATTACK ANGLES



DATAFLOW ENCRYPTION: Presentation

DEFINITION

A data flow encryption vulnerability occurs when it is possible for an attacker **intercepting** a data flaw to **retrieve the data in transit**

In some cases, it is even possible to **modify the data** on the fly

This vulnerability can occur if **weak encryption protocols or algorithms** are used, or if the application **certificate is not robust**

VULNERABLE CASES

Applicative data are sent in **clear text**

Old and **weak encryption protocols and algorithms** are used

Weak crypto keys are generated and reused

What risks if not or poorly done?

1

Unauthorized resource access:
Known vulnerabilities on weak encryption protocols and algorithms can be used to access data in transit

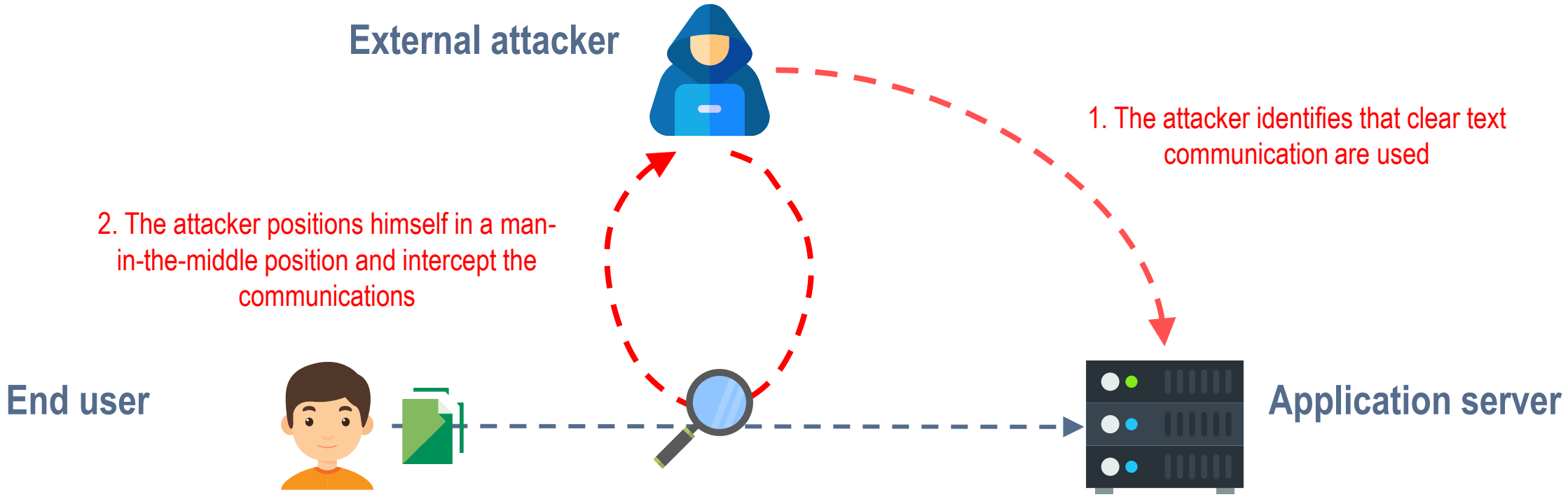
2

Session or identity theft:
An attacker who intercept accounts information transmitted in clear text can reuse them

3

Legal issues:
The non compliance of data protection regarding laws or regulations such as the EU GDPR or local privacy laws can result in large monetary fines

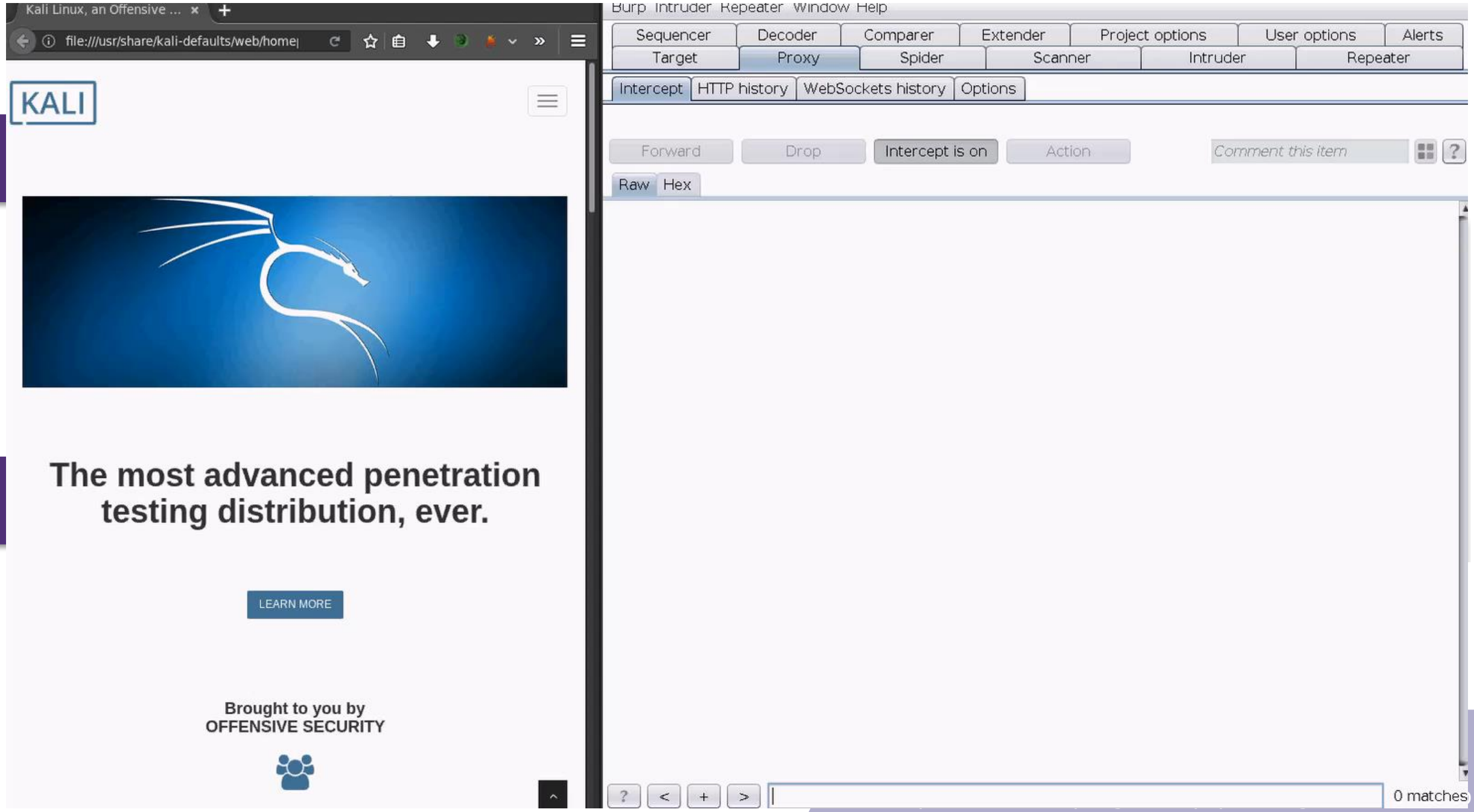
DATAFLOW ENCRYPTION: An attack scenario example



Scenario 1: clear text data flaw

DATAFLOW ENCRYPTION: Prevention

- Encrypt all data in transit with secure protocols
- Ensure up-to-date and strong standard encryption protocols and algorithms
- Use proper key management


Kali Linux, an Offensive ...
file:///usr/share/kali-defaults/web/home
KALI
The most advanced penetration testing distribution, ever.
LEARN MORE
Brought to you by
OFFENSIVE SECURITY
Burp Intruder Repeater Window Help
Sequencer
Decoder
Comparer
Extender
Project options
User options
Alerts
Target
Proxy
Spider
Scanner
Intruder
Repeater
Intercept
HTTP history
WebSockets history
Options
Forward
Drop
Intercept is on
Action
Comment this item
Raw
Hex
0 matches

REALLY USEFUL TOOL PRESENTATION: Burp Suite

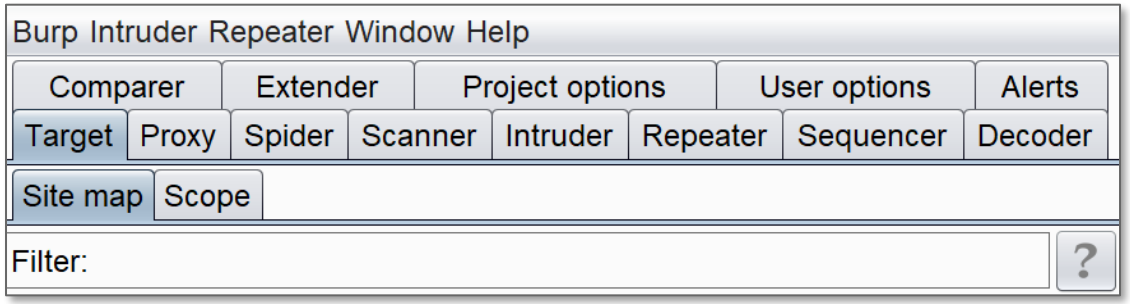
PRESENTATION

Burp Suite:

→ Burp is a **graphical tool** developed by PortSwigger Security for **testing Web application** security with a **free edition**

→ The tool includes **numerous functionalities** with different purposes

→ A **free version** of the tool exists but does not include the scanner functionality and does not permit backup



Download link:
https://portswigger.net/

REALLY USEFUL TOOL PRESENTATION: Burp Suite

PRESENTATION

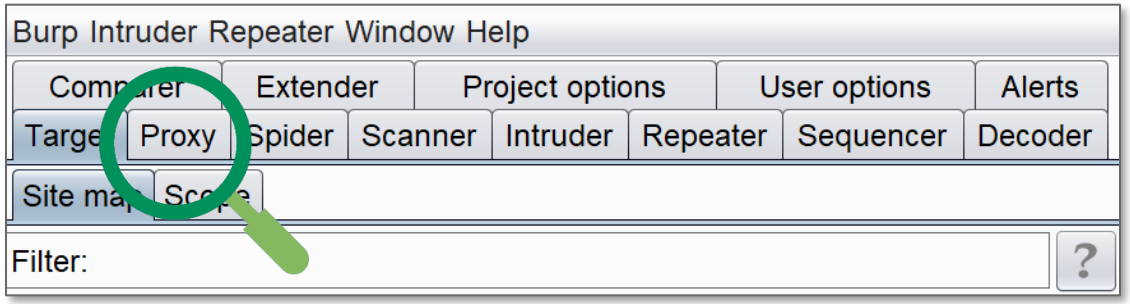
Proxy:

→The purpose of the **Burp Proxy** is to allow testers to **intercept requests and responses** between the browser and the target application, **even when HTTPS is being used**

→ To use it, **the browser must be configured** to use Burp as proxy (by default on **127.0.0.1:8080**)

→ If the **interception is off**, **every requests and responses** will be **visible in the proxy tab** without altering the navigation

→ If the **interception is on**, **every request will be blocked** until the user **forward** it (with or without **modification**)



Download link:
https://portswigger.net/

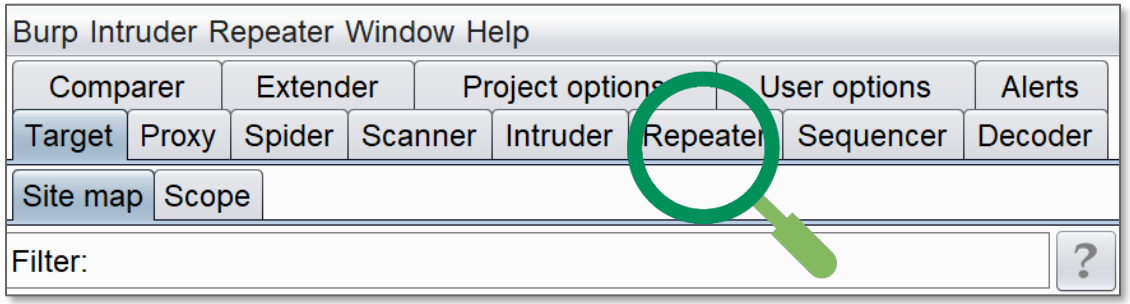
REALLY USEFUL TOOL PRESENTATION: Burp Suite

PRESENTATION

Repeater:

→ The **Burp repeater** allows testers to **replay a previously intercepted request**

→ **Changes in all part of the request** can be done before replaying the request



Download link:
https://portswigger.net/

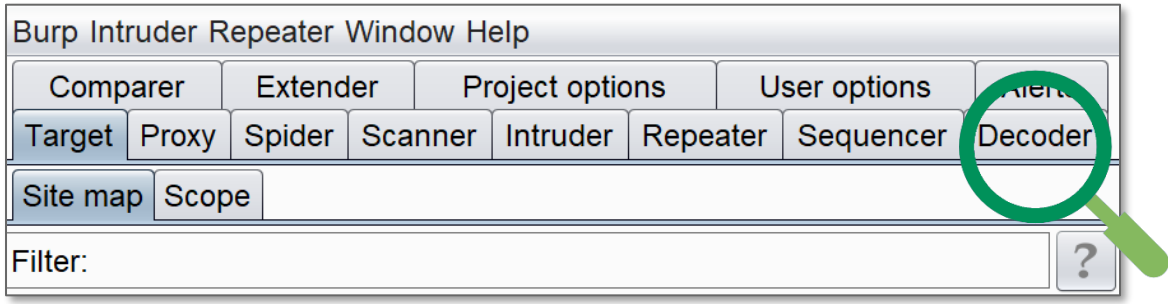
REALLY USEFUL TOOL PRESENTATION: Burp Suite

PRESENTATION

Decoder:

→ **Burp Decoder** is a simple tool for **transforming encoded data into its canonical form**, or for **transforming raw data into various encoded and hashed forms**

→ It is capable of intelligently **recognizing several encoding formats** using heuristic techniques



Download link:
https://portswigger.net/

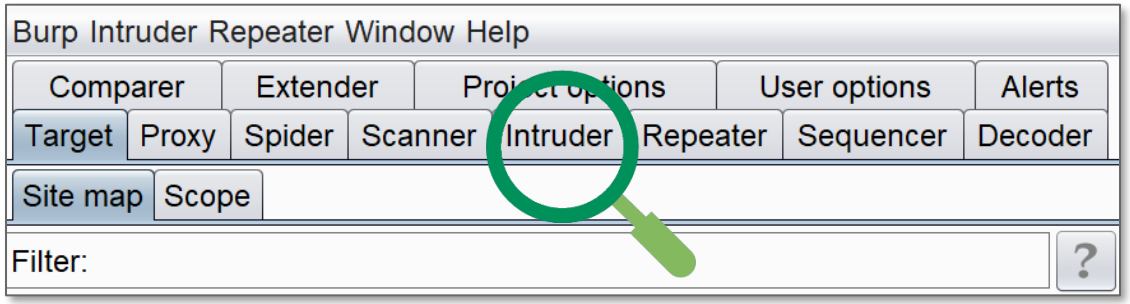
REALLY USEFUL TOOL PRESENTATION: Burp Suite

PRESENTATION

Intruder:

→ **Burp Intruder** is a powerful tool for a**utomating customized attacks** against web applications

→ It is mainly used in penetration testing to perform **a wordlist attacks** (e.g. brute force using a predefined set of payloads)

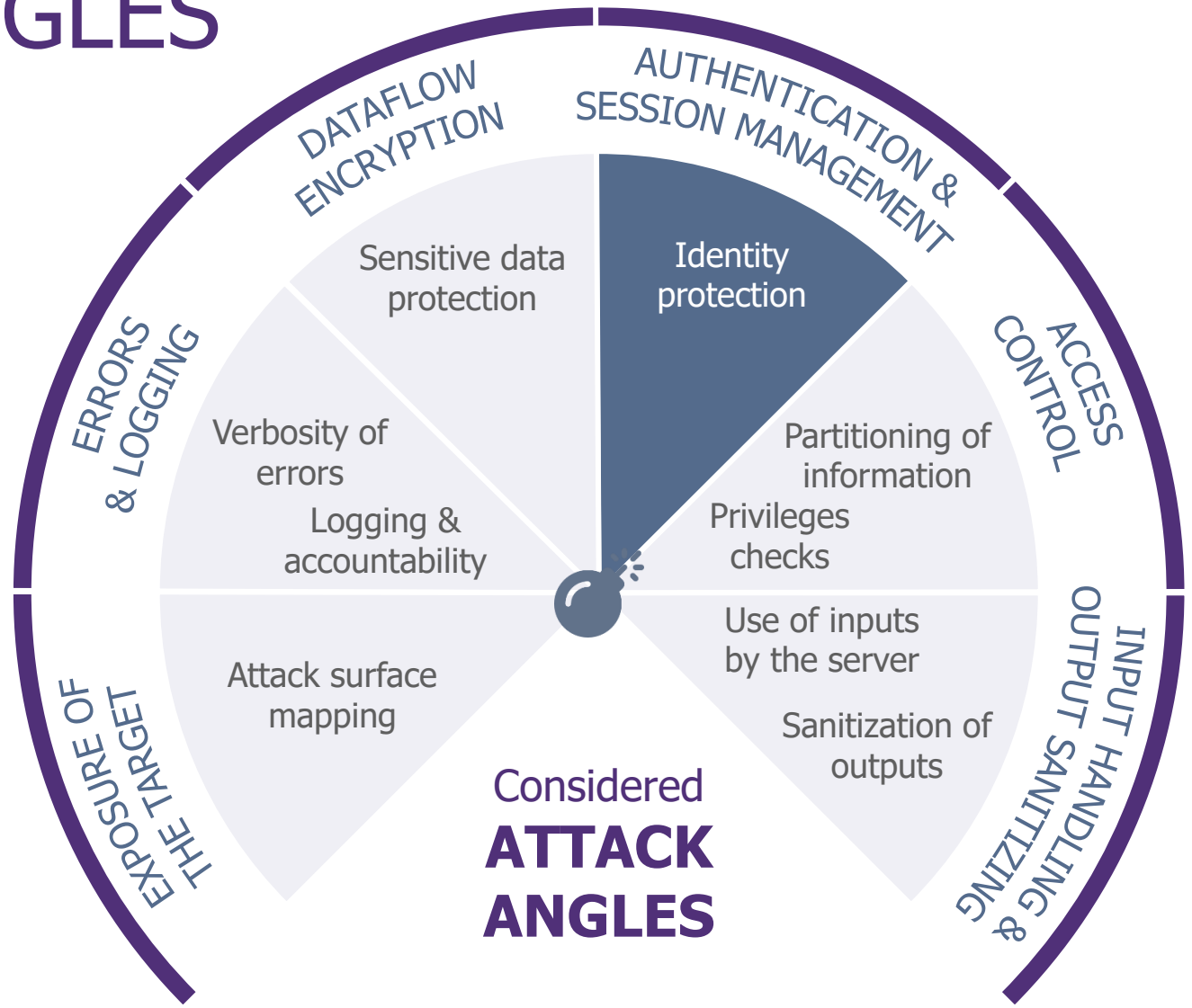


Download link:
https://portswigger.net/

SECURITY CONFIGURATION: Prevention

- **Restrict administration services exposure**
- **Change default accounts and passwords during installation**
- **Regularly update installed services**
- **Disable or uninstall unnecessary features**

ATTACK ANGLES



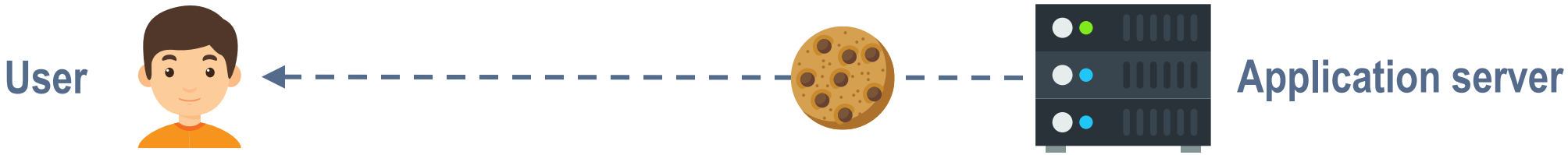
WEB AUTHENTICATION WORKING

To **authenticate**, the user **sends his credentials** to the web application, preferably in a **POST request**

```
POST /login.php

username=toto&
password=azerty123
```

WEB AUTHENTICATION WORKING



The server sends a **session cookie** (a simple string) back to the user

This cookie will then be **automatically added by the user browser** in **every request** and represents the **user identity** on the web application

Set-cookie: ab3562f123aeb1325cd235

COOKIES MANAGEMENT: Presentation

DEFINITION

- Session cookies are equivalent to the **user identity** for the time of the session
- **A cookie management vulnerability** occurs when it is possible for an attacker to **guess or retrieve a cookie**

VULNERABLE CASES

- A **session cookie** is created thanks to **guessable information** (timestamp, role, etc.)
- The **session cookie** does not possess the **Secure** and **httpOnly** attributes
- The session is still **valid after logoff**

What risks if not or poorly done?

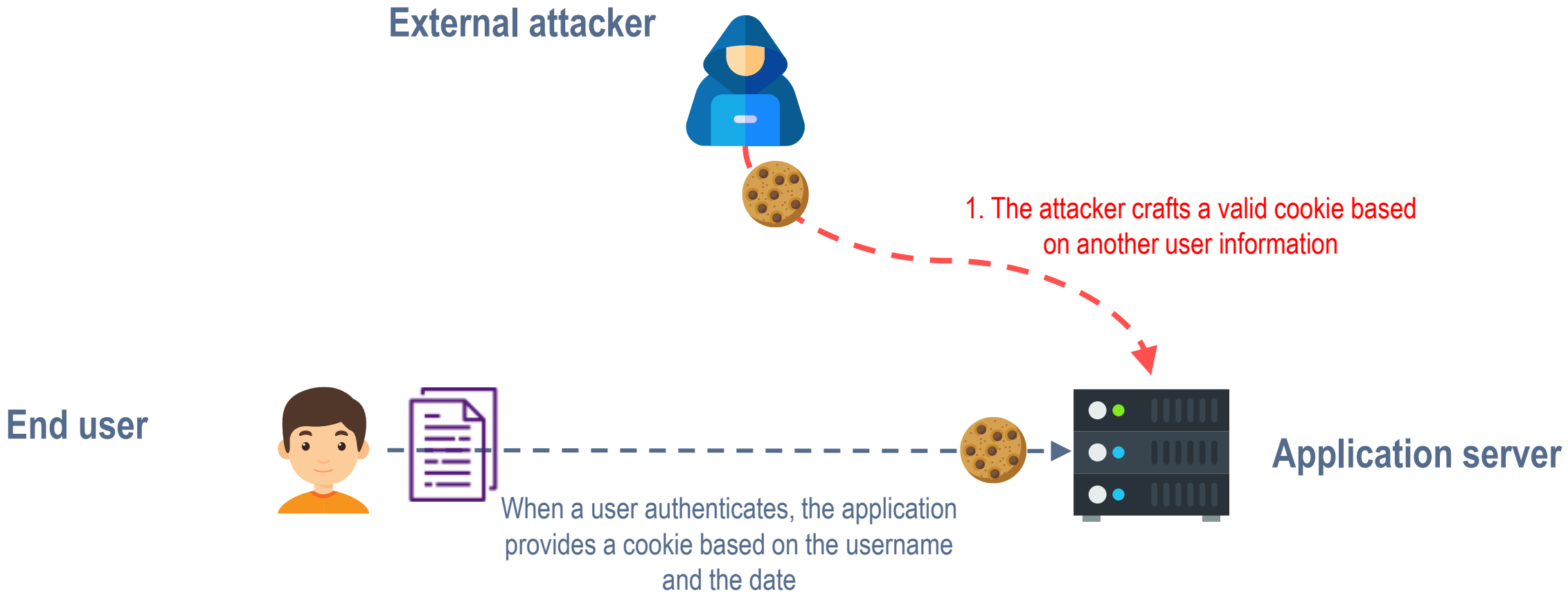
1

Session or identity theft:
Combining with a XSS, the lack of the httpOnly attribute can permit an attacker to obtain a user cookie

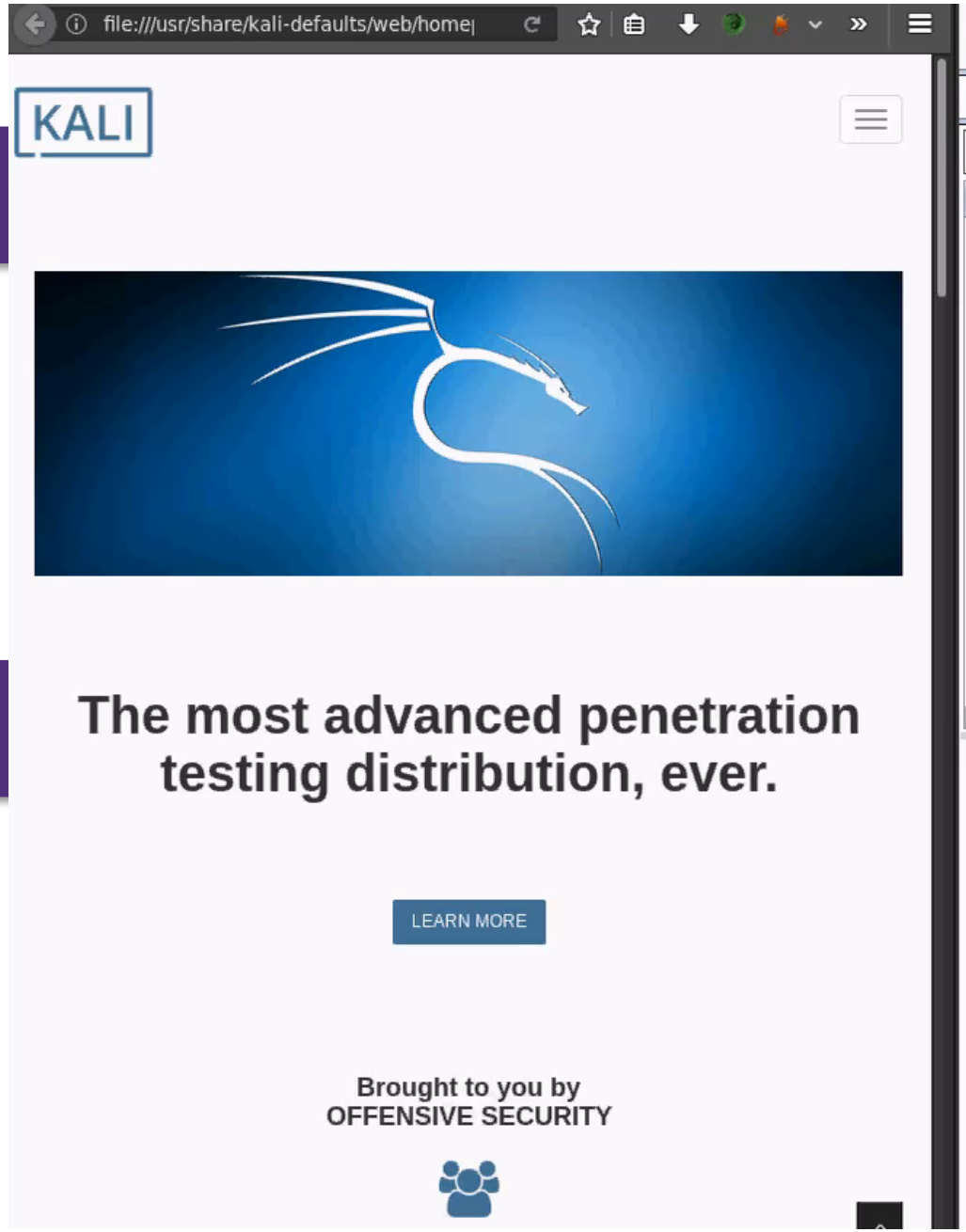
2

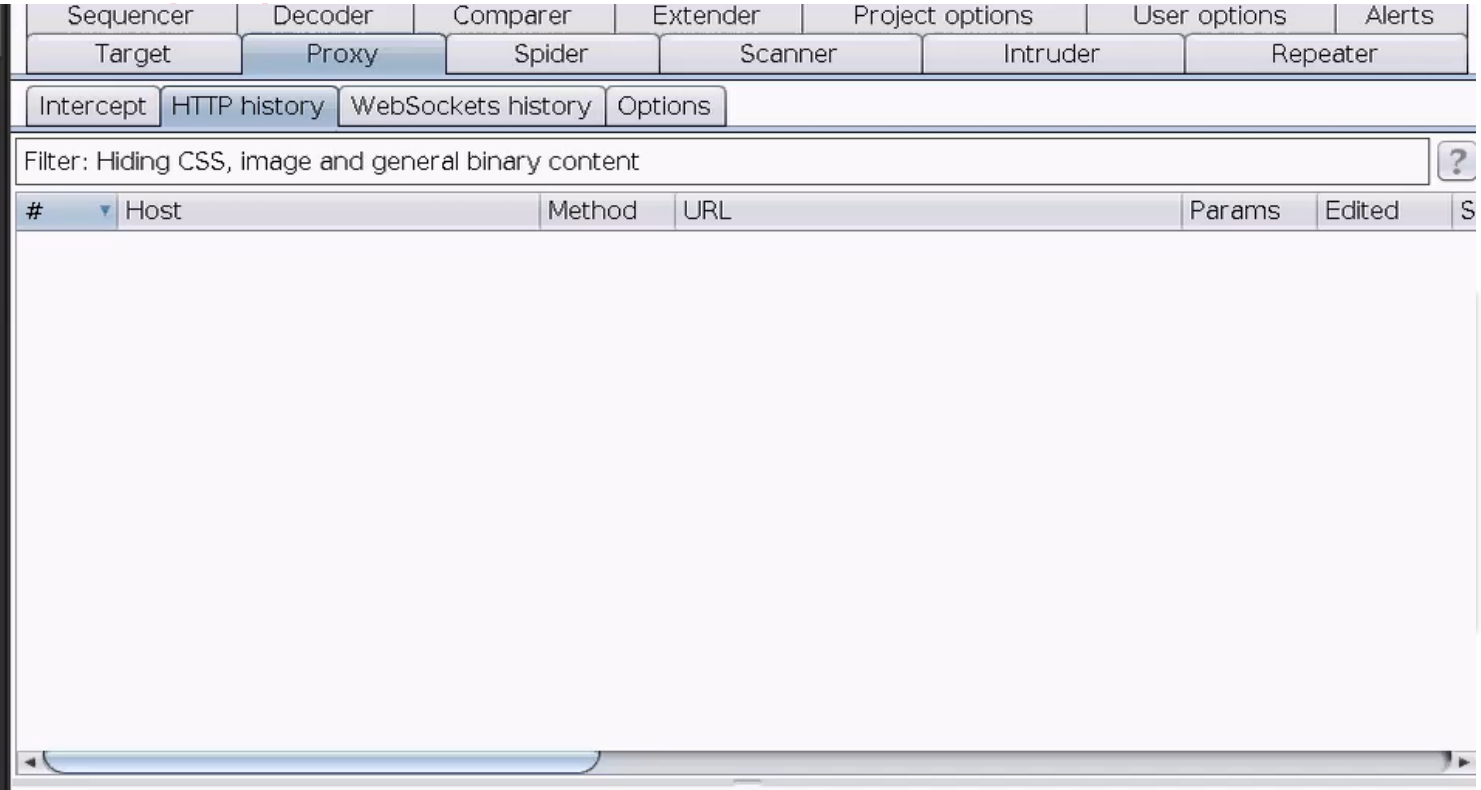
Unauthorized resource access:
An attacker guess another user cookie and access resources available to this user

COOKIES MANAGEMENT: An attack scenario example



Scenario 1: use of guessable cookies


file:///usr/share/kali-defaults/web/home
KALI
The most advanced penetration testing distribution, ever.
LEARN MORE
Brought to you by
OFFENSIVE SECURITY

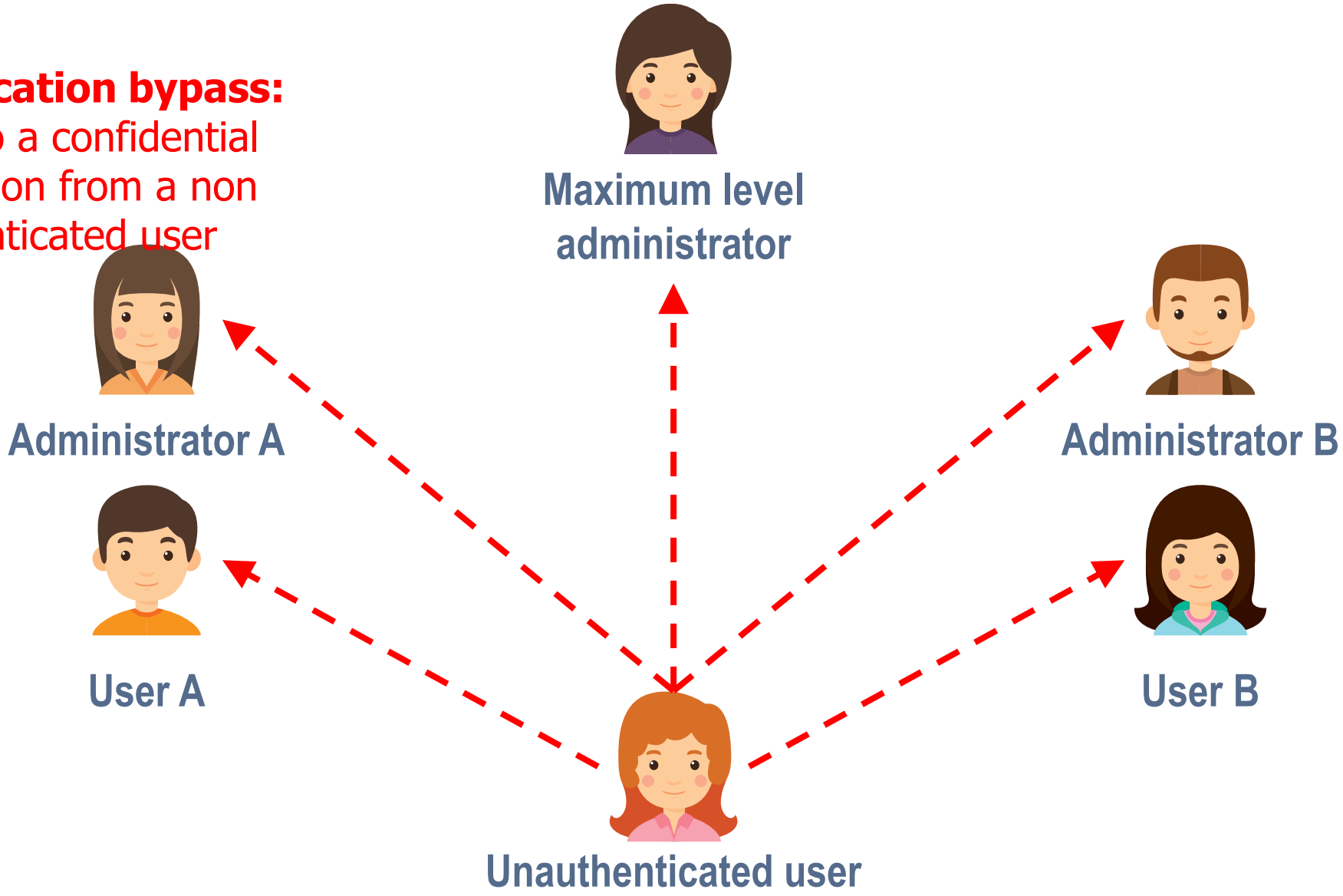

Sequencer
Decoder
Comparer
Extender
Project options
User options
Alerts
Target
Proxy
Spider
Scanner
Intruder
Repeater
Intercept
HTTP history
WebSockets history
Options
Filter: Hiding CSS, image and general binary content
#
Host
Method
URL
Params
Edited

COOKIES MANAGEMENT: Prevention

- Ensure the randomness of session cookies
- Set the Secure and httpOnly parameters on every session cookies
- Invalidate cookies upon logoff or inactivity

AUTHENTICATION BYPASS: Schematic

Authentication bypass: access to a confidential information from a non authenticated user



AUTHENTICATION BYPASS: Presentation

DEFINITION

Authentication bypass is a kind of vulnerability where **unauthenticated users** access **authenticated functionalities**

This can occur while bypassing the authentication mechanism, or by **finding a valid account**

VULNERABLE CASES

Weak passwords are accepted, and **no anti-brute force mechanism** is in place

The application uses **weak or ineffective credential recovery and forgot-password processes**, such as “knowledge-based answers”

The application exposes the **session ID in the URL**

What risks if not or poorly done?

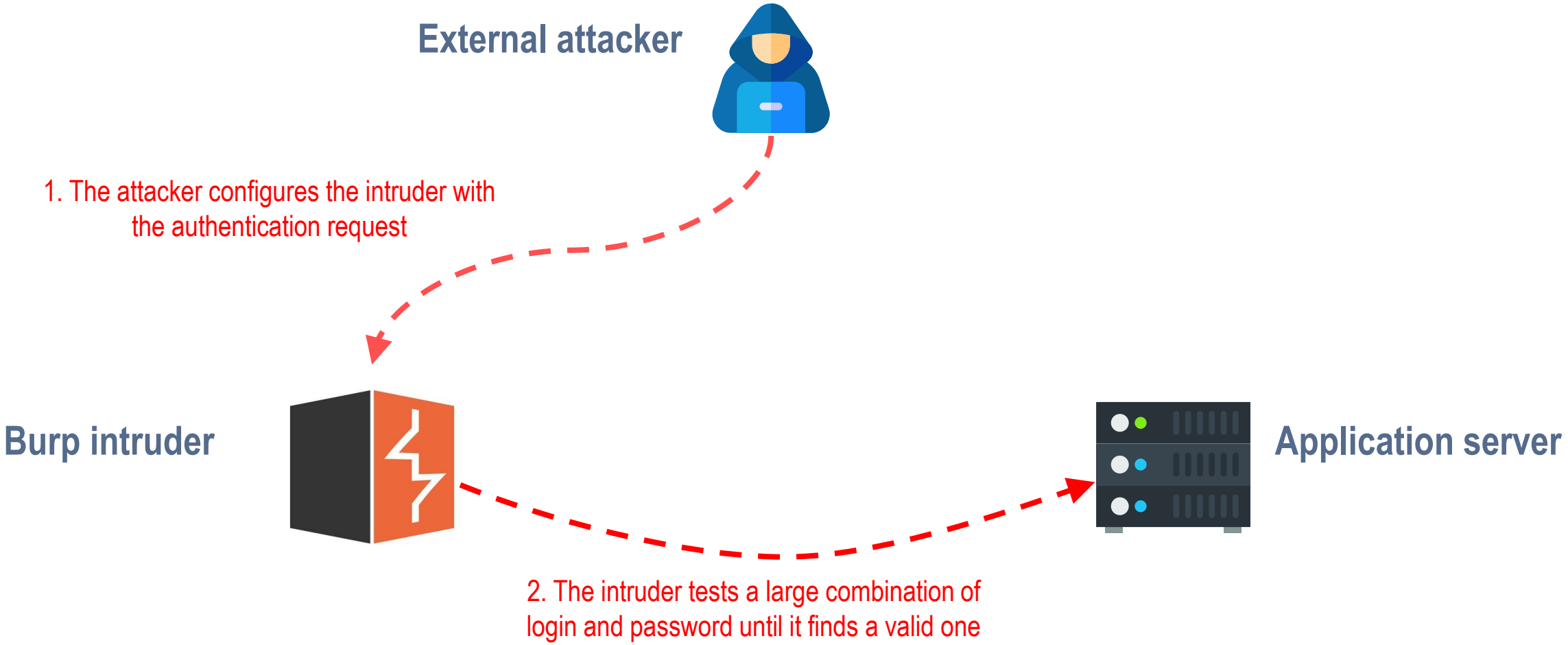
1

Unauthorized resource access:
After bypassing authentication, an attacker can access sensible information

2

Session or identity theft:
If credentials are guessable or brute forceable, an attacker can steal a user identity

AUTHENTICATION BYPASS: An attack scenario example



Scenario 1: accounts password brute force

Scenario 2: access to authenticated data without cookies

Scenario 3: modification of access control parameter

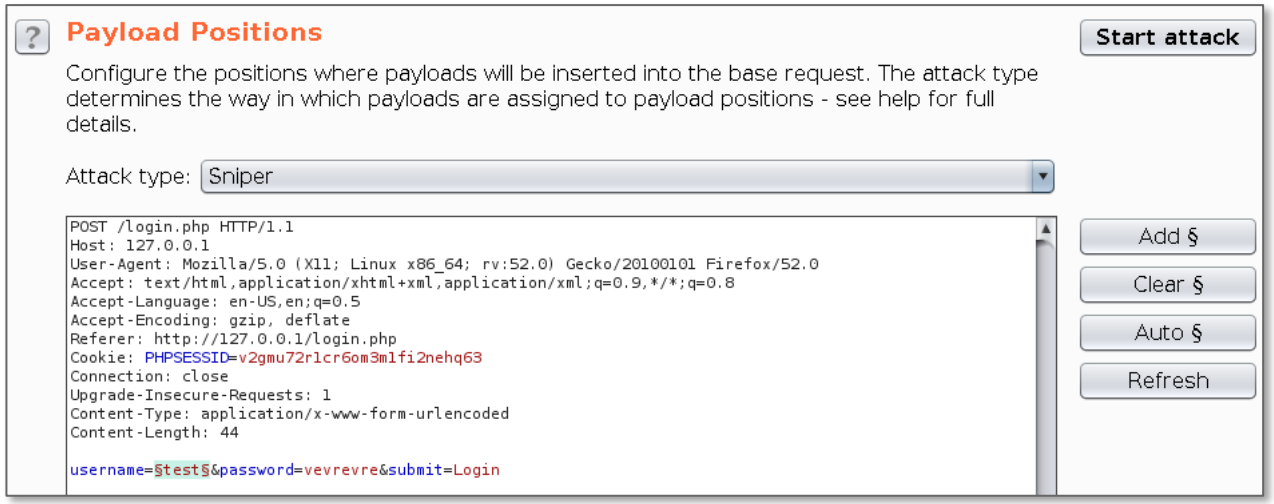
AUTHENTICATION BYPASS: Testing methodology

TOOL & METHODOLOGY

Manual testing with Burp Suite:

→ **Authentication bypass** can be done with several methods where burp **repeater** and **intruder** can be useful:

- Direct page request
- Parameter modification (such as a "authenticated" parameter) or session ID prediction
- Login and password brute force



EXAMPLES

\ Set your Internet proxy with Burp (`127.0.0.1:8080`)

\ Find an interesting request, and send it to the intruder (`right click > send to Intruder` or `CTRL + I`)

\ Configure the `payloads positions` (e.g. username and password) and the `payloads` to test (e.g. username and password list)

\ Click `Start attack` and working payloads

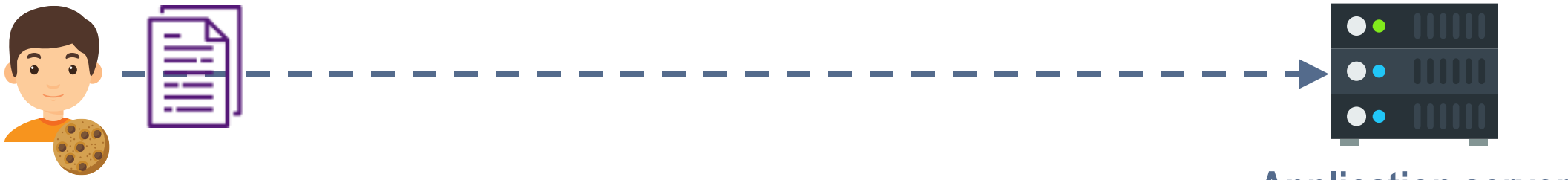
Complementary resources:
https://portswigger.net/burp
https://www.owasp.org/index.php/Testing_for_Bypassing_Authentication_Schema_(OTG-AUTHN-004)

AUTHENTICATION BYPASS: Prevention

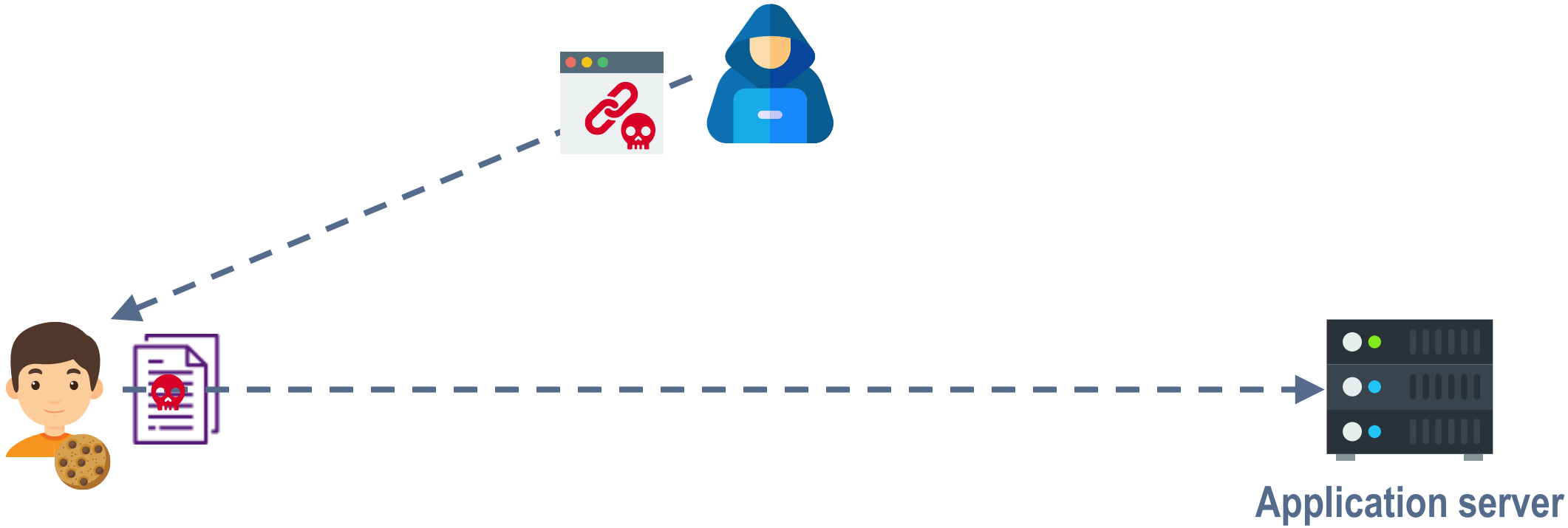
- Implement weak-password checks, or at least enforce a strong password policy
- Activate or implement anti-brute force mechanisms
- Use secured credential recovery and forgot password functionalities
- Implement multi-factor authentication where possible

XSRF (OR CSRF): Workflow

When a user is **already logged** in an application and **sends a new request**, **the browser automatically adds the session cookie** to the request



XSRF (OR CSRF): Workflow



If a request is **fully predictable**, an attacker can trick a user to send a **malicious preformed request**...

… which will be sent with the **user cookie**

XSRF (OR CSRF): Presentation

DEFINITION

XSRF is a kind of vulnerability where it is possible to **force an end user to execute unwanted actions**

As the **Web browser automatically add the end user cookie** when performing a request, an attacker who force this user to execute a **forged request** will result in this request being **executed with the end user cookie**

VULNERABLE CASES

A parameter which indicate the **user id** is included inside an **action request** to prevent XSRF

A **POST request** use a **specific cookie** to protect against XSRF

A **random parameter** is correctly added inside the content of the request, but **no validation** of this parameter is made

What risks if not or poorly done?

1

Unauthorized resource access:
An attacker trick a user to make him create, modify or delete some data

2

Session or identity theft:
An attacker use a XSRF on a change-password functionality and take control of a user account

XSRF (OR CSRF): Testing methodology

TOOL & METHODOLOGY

Burp Suite:

→ The Burp **repeater** can be used to **modify an anti XSRF token** in order to test if it is **correctly validated**

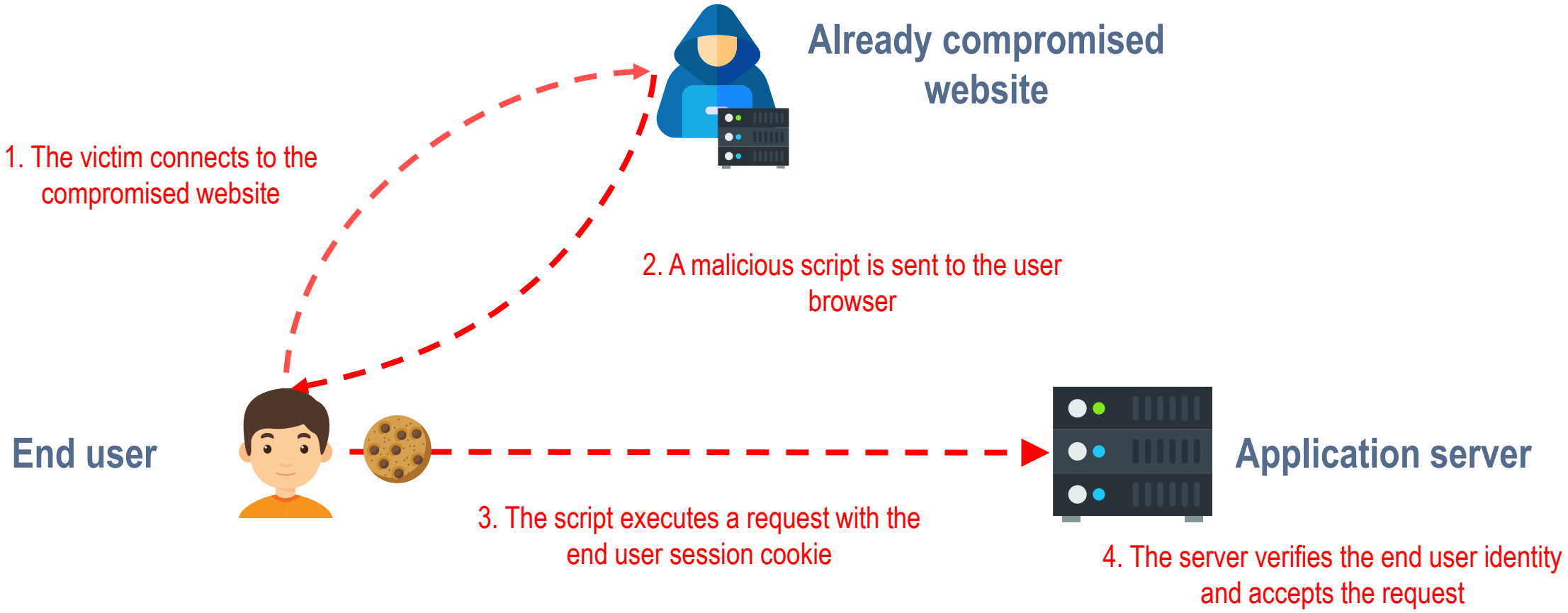
→ The **sequencer** can also be used in order to **test the token entropy**

EXAMPLES

- Set your Internet proxy with Burp (`127.0.0.1:8080`)
- Find an action request, and identify if there is a likely random token in the request
- Send the request to the repeater (`right click > send to Repeater` or `CTRL + R`) and try to replay it with a bad value for this token
- Eventually send the request which provides the random token to the Sequencer (`right click > send to Sequencer`) in order to test the token entropy

Complementary resources:
https://portswigger.net/burp
https://www.owasp.org/index.php/Testing_for_CSRF_(OTG-SESS-005)

XSRF (OR CSRF): An attack scenario example

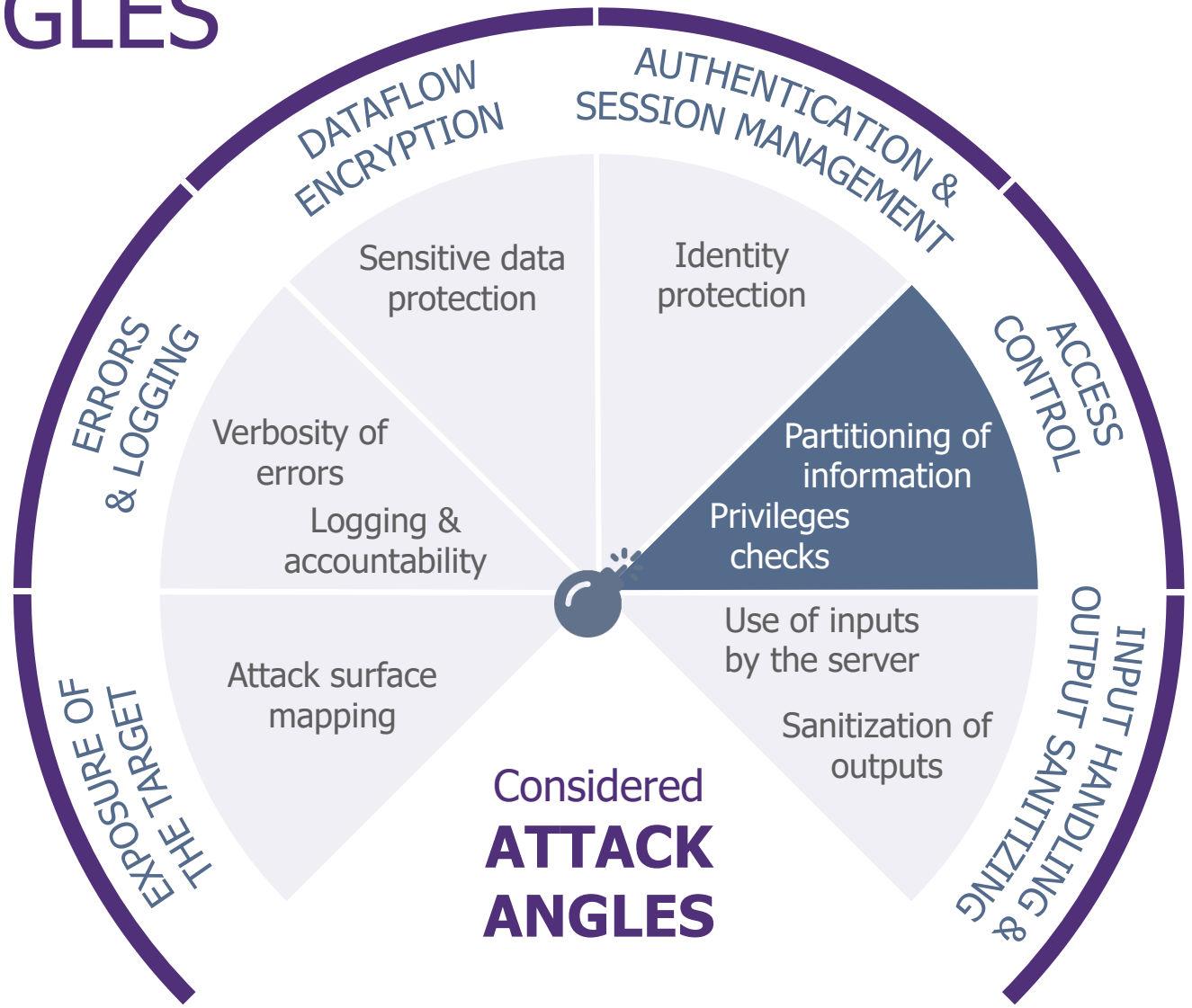


Scenario 1: cross site request forgery scenario

XSRF (OR CSRF): Prevention

- Include a random parameter on every action request
- Correlate this parameter with the session cookie
- For sensitive functionalities, ask the password to validate user identity

ATTACK ANGLES



VERTICAL ESCALATION: Schematic

Vertical escalation:
privilege elevation to access advanced functionalities

VERTICAL ESCALATION: Presentation

DEFINITION

Vertical escalation occurs when it is possible for a **limited privileged user** to access **privileged functionalities**

VULNERABLE CASES

A **standard user** access a **privileged functionality** by **accessing the URL**

The **administration menu** is disabled using **HTML** but no protection is implemented on the **server side**

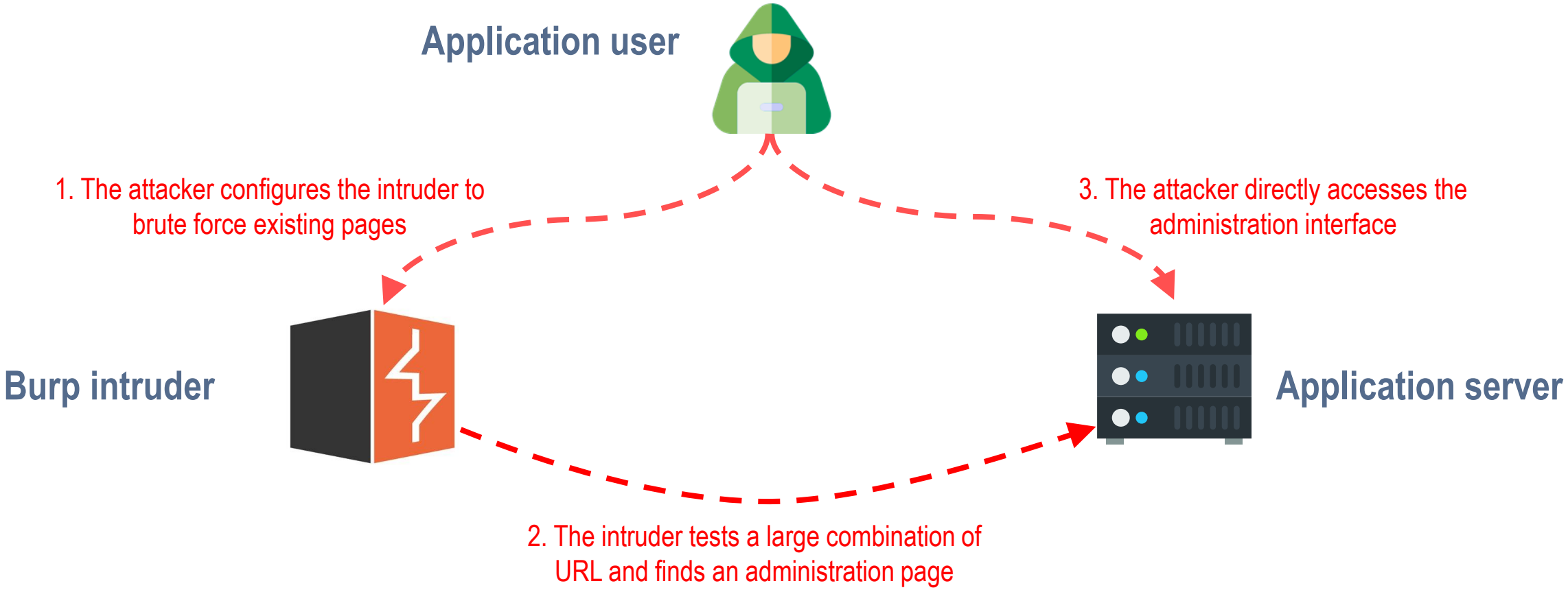
The **access privileges** are only determined by the **request parameters** and not by the session cookie

What risks if not or poorly done?

1

Privileges escalation:
Bypassing access control, an attacker access some administrative functionalities

VERTICAL ESCALATION: An attack scenario



Scenario 1: administration page brute force

Scenario 2: HTML access control bypass

Scenario 3: cookie based access control

VERTICAL ESCALATION: Testing methodology

TOOL & METHODOLOGY

dirsearch:

→ **dirsearch** is a simple command line tool designed to **brute force directories and files in websites**

```
root @ kali > python3 dirsearch.py -u http://sequoia.wavestone.com -e php

 _|. _ _  _  _  _ _|_    v0.3.8
(_||| _) (/_(_|| (_| )

Extensions: php | Threads: 10 | Wordlist size: 5963

Error Log: /root/Bureau/dirsearch-master/logs/errors-18-08-20_15-20-11.log

Target: http://sequoia.wavestone.com

[15:20:13] Starting:
[15:20:19] 302 -    3B  - /php  ->  http://sequoia.wavestone.com/wp-login.php?redirect_to=/php
[15:20:19] 301 -    3B  - /%2e%2e/google.com  ->  http://sequoia.wavestone.com/../google.com
[15:20:19] 403 -    9B  - /.Administration
[15:20:19] 403 -    9B  - /.admin
[15:20:19] 403 -    9B  - /.adm
```

EXAMPLES

\ Usage with default files database of a specific extension (recursively and ignoring 403 status code):

→ `python3 dirsearch.py -u <URL> -e <file_extension> -r -x 403`

\ Specify a wordlist of folders and files:

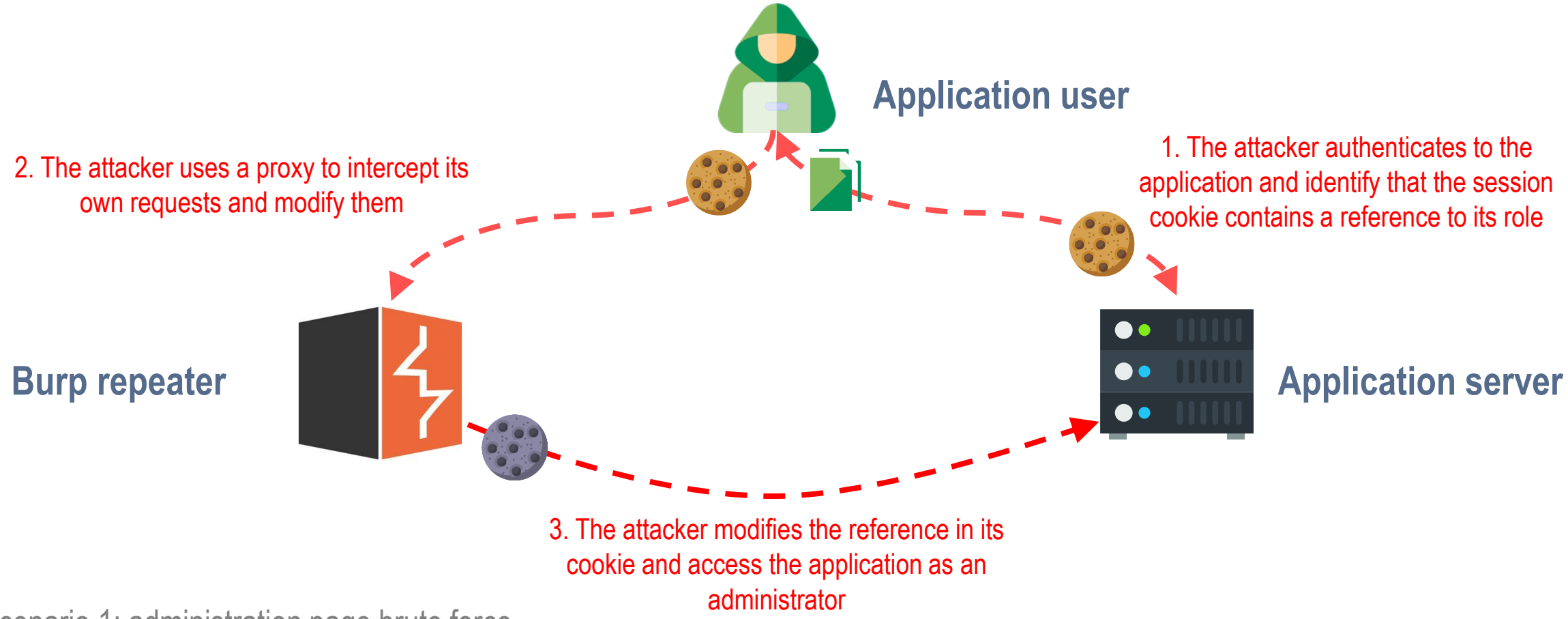
→ `python3 dirsearch.py -u <URL> -w <wordlist> -r -x 403`

\ Authenticated usage:

→ `python3 dirsearch.py -u <URL> -e <file_extension> -c <session_cookie> -r -x 403`

Complementary resources:
https://github.com/maurosoria/dirsearch
https://www.owasp.org/index.php/Testing_for_Privilege_escalation_(OTG-AUTHZ-003)

VERTICAL ESCALATION: An attack scenario



Scenario 1: administration page brute force

Scenario 2: HTML access control bypass

Scenario 3: cookie based access control

VERTICAL ESCALATION: Prevention

- Validate user privileges on access using the session cookie
- Use a model access control to enforce privileges
- Use server side mechanisms to perform access control

HORIZONTAL ESCALATION: Presentation

DEFINITION

Horizontal escalation arises when a user can **access resources granted to a similarly configured account**

VULNERABLE CASES

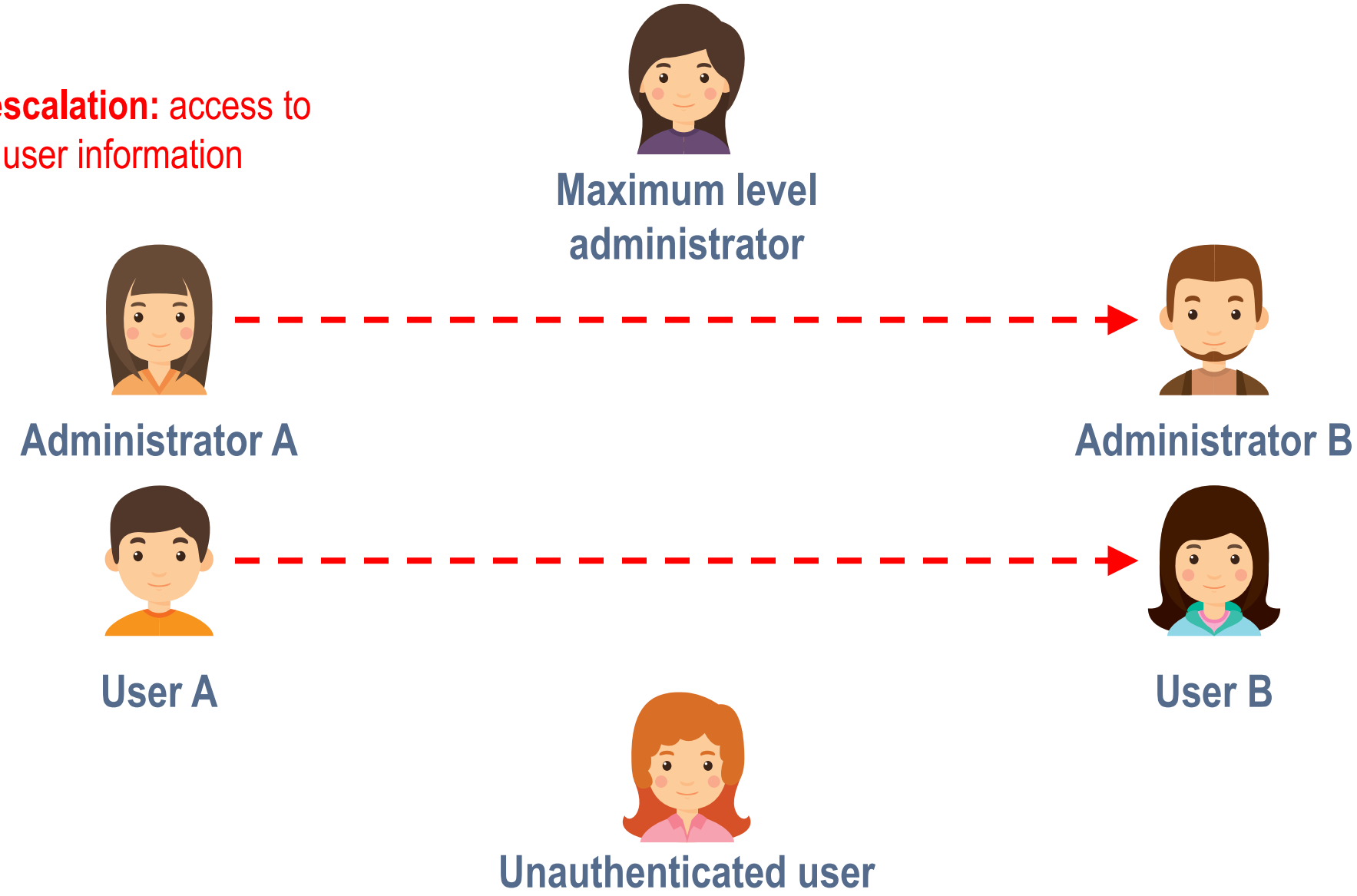
Access to user information is made using an **identity reference** and **no verification is made on the user identity**

What risks if not or poorly done?

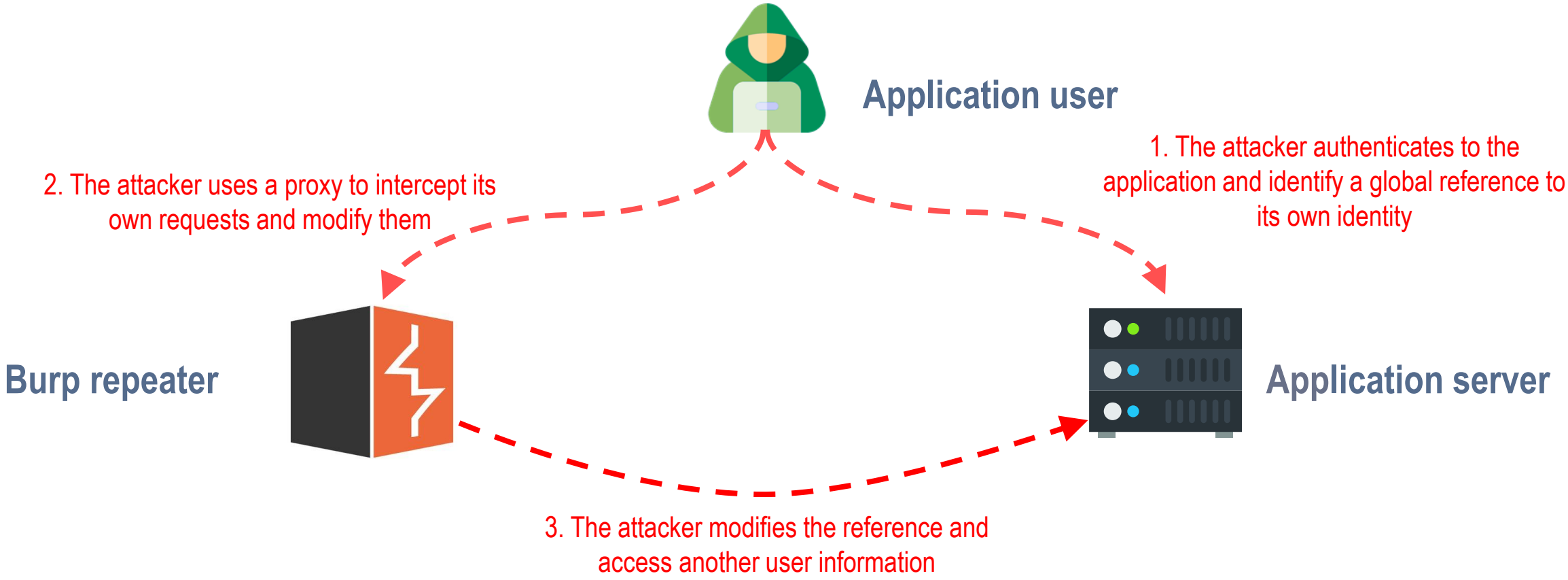
1 **Unauthorized resource access:**
Access to personal resources of another user having the same privileges

HORIZONTAL ESCALATION: Schematic

Horizontal escalation: access to another user information



HORIZONTAL ESCALATION: An attack scenario example



Scenario 1: access to another user information

HORIZONTAL ESCALATION: Testing methodology

TOOL & METHODOLOGY

Burp Suite:

→ The Burp **repeater** can be used to replay a request while **modifying a parameter referencing a specific user**

→ The **intruder** can also be used to **iterate other such a parameter** in order to access the information of all users

EXAMPLES

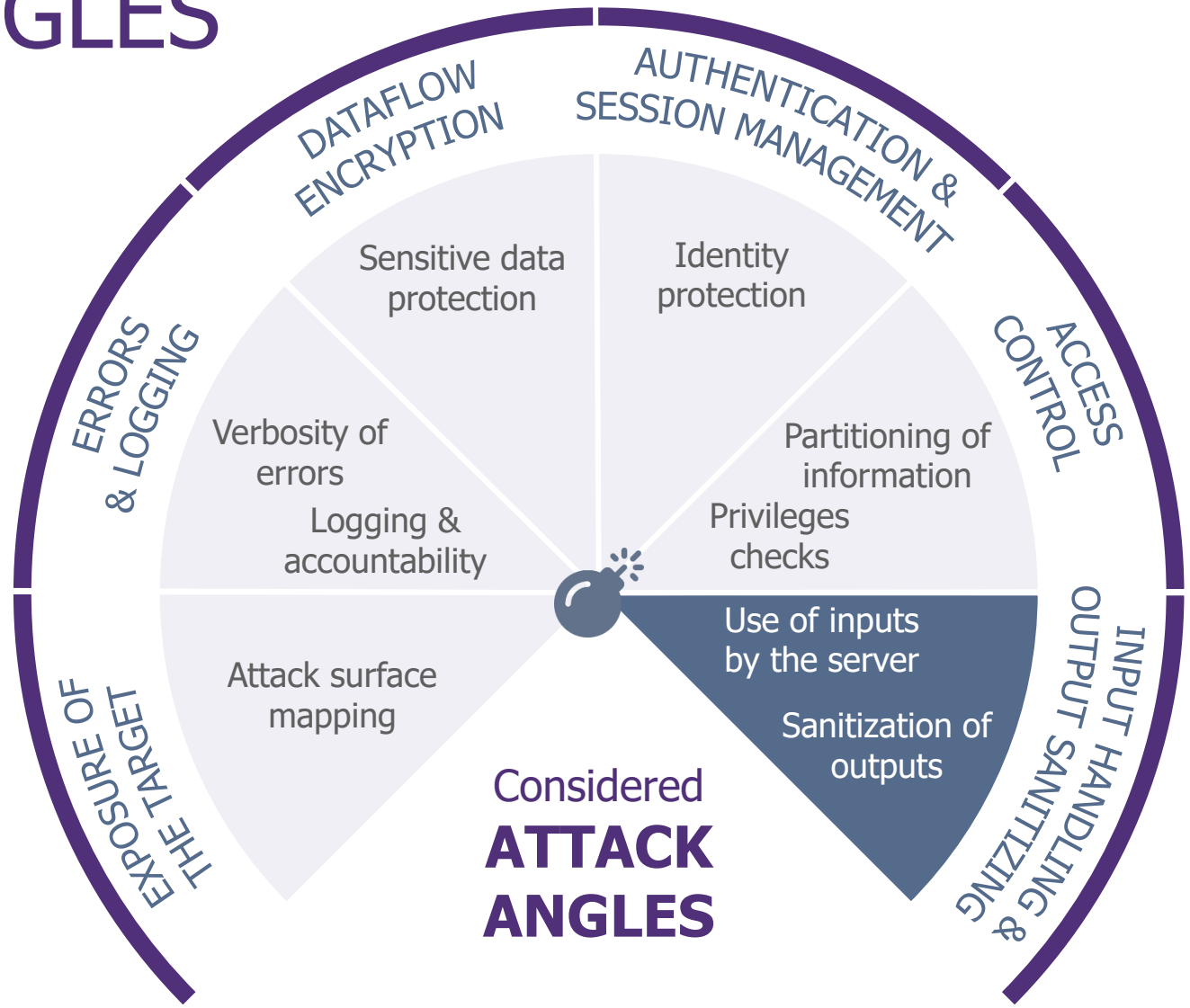
- Set your Internet proxy with Burp (`127.0.0.1:8080`)
- Find an interesting request with a user id as parameter
- Send the request to the repeater (`right click > send to Repeater` or `CTRL + R`) and try to replay it with another user id
- Eventually send the request to the Intruder (`right click > send to Intruder` or `CTRL + I`) in order to brute force existing ids

Complementary resources:
https://portswigger.net/burp
https://www.owasp.org/index.php/Testing_for_Insecure_Direct_Object_References_(OTG-AUTHZ-004)

HORIZONTAL ESCALATION: Prevention

Validate user identity when accessing a resource using the session cookie

ATTACK ANGLES



INJECTION: Workflow

Most of the time, **application data** are **stored** in a **database**

When a user **accesses or updates a data** a request containing some **parameters** is sent to the server

The application server **uses these parameters** to create a **database query**

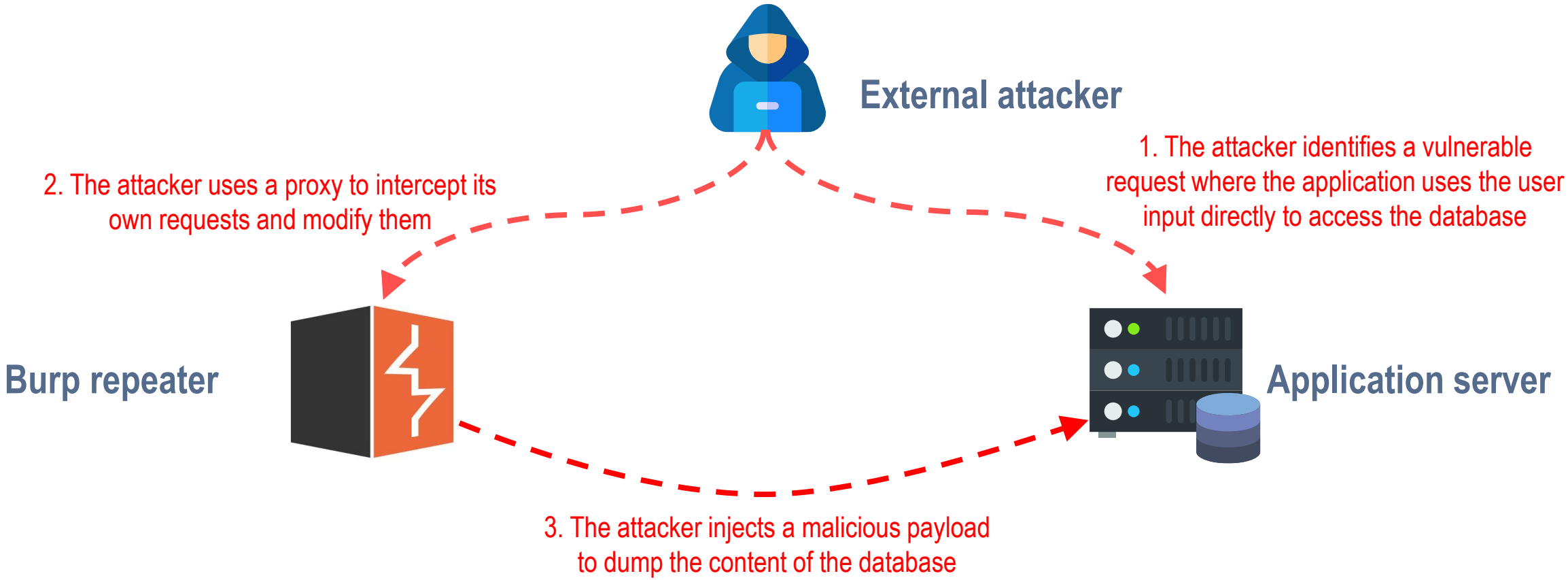
End user

Application server

Database

If the application **does not sanitize the parameters** to create the database query, it is possible for the user to **inject some parameters** to control the database query

INJECTION: An attack scenario example



Scenario 1: SQL injection

INJECTION: Presentation

DEFINITION

Injection flaws occur when an attacker can send **hostile data to an interpreter**

This kind of vulnerability is often found in **SQL, NoSQL or LDAP**

VULNERABLE CASES

An input parameter is **interpreted in the database without sanitization of the content**

A **LDAP directory** is used to reference the users of the application. An **LDAP query** is directly sent with the **input of the search form**

What risks if not or poorly done?

1. **Unauthorized resource access:**
   Injections can result in data loss, corruption or disclosure
2. **Server compromise:**
   Depending on the database management system configuration, a SQL injection can sometime result in a system shell
3. **Privilege escalation:**
   Sometimes, the user privileges can be changed in the database using a SQL injection

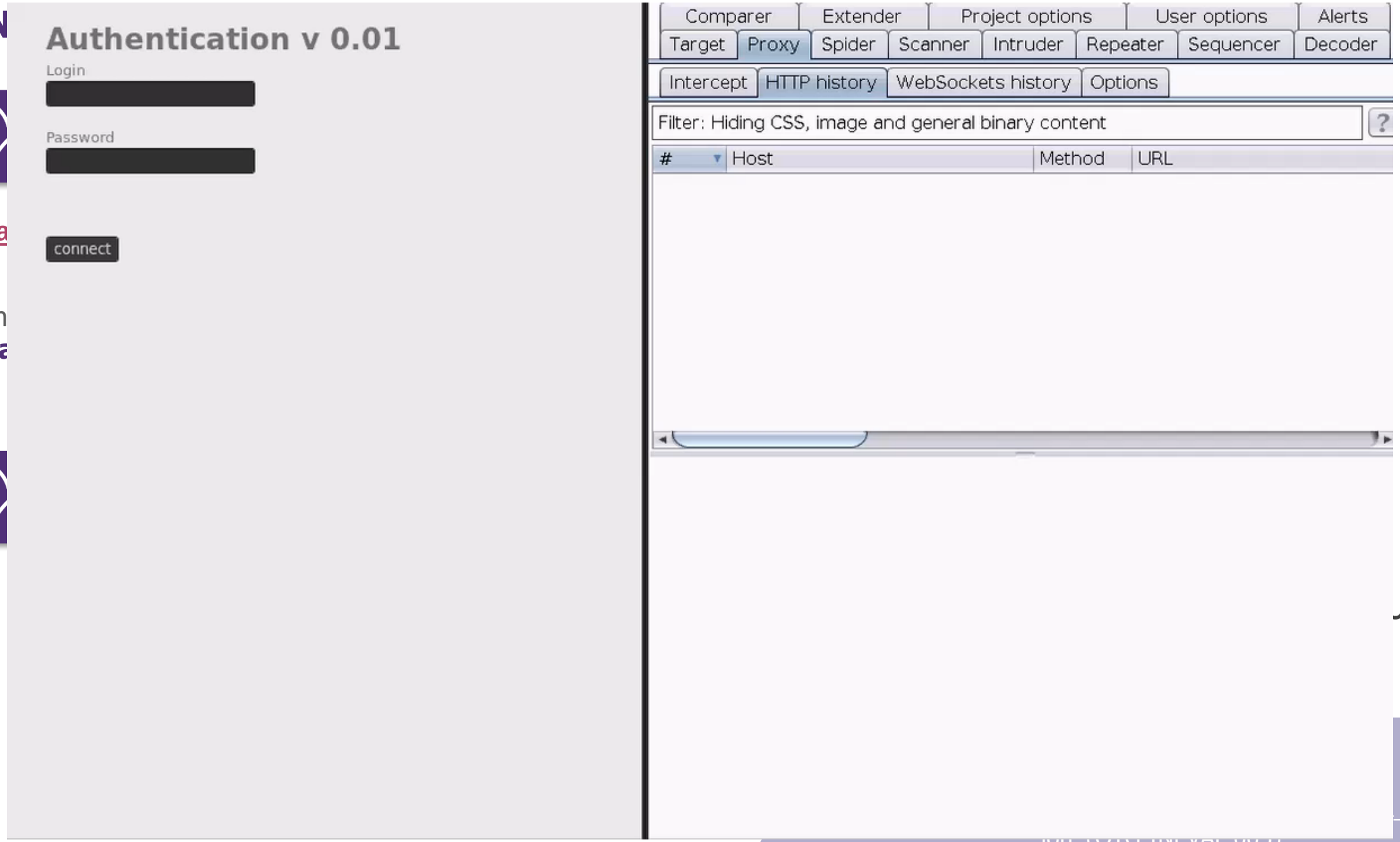
IN

Ma

→
wh
ma

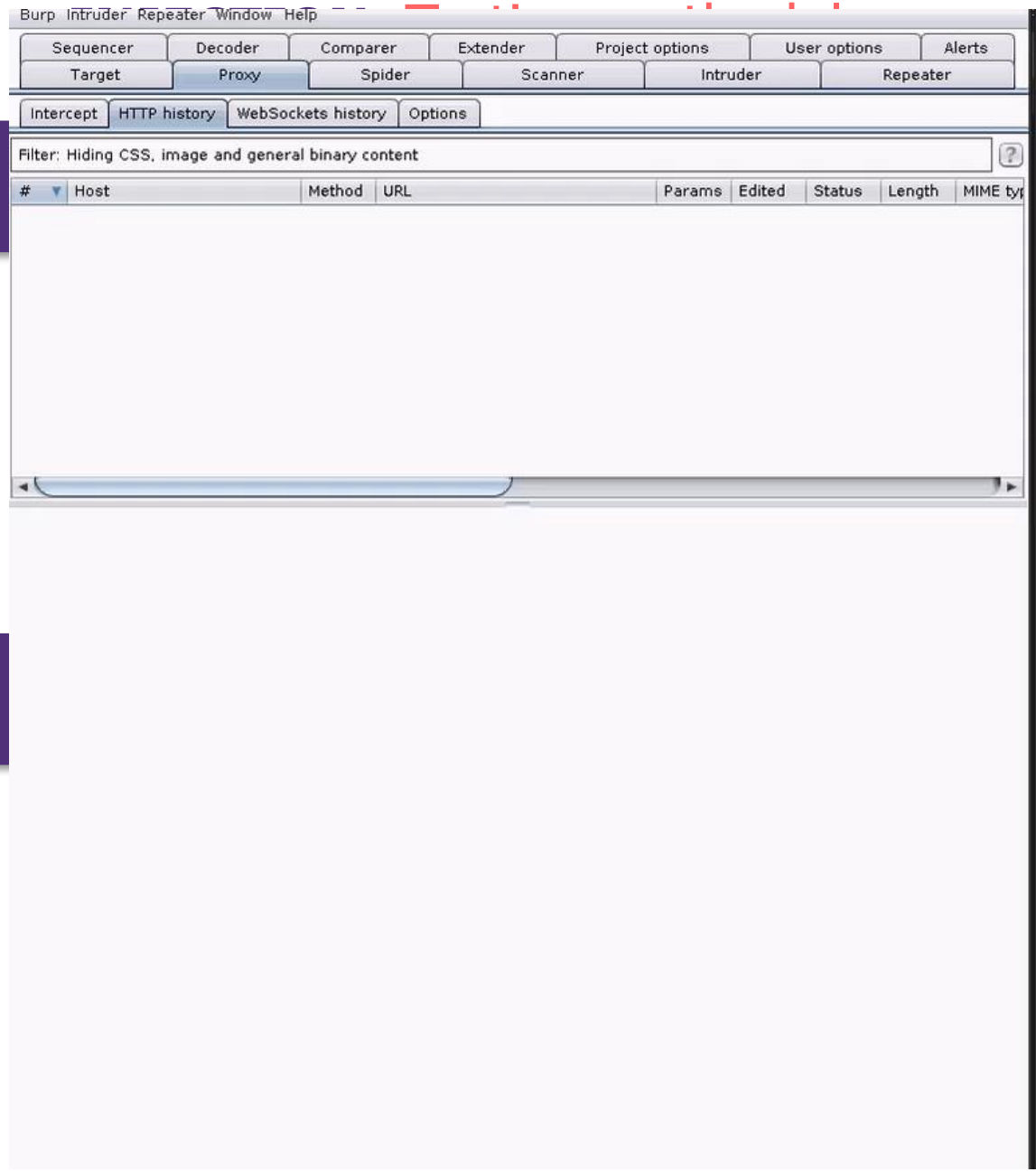
\
\

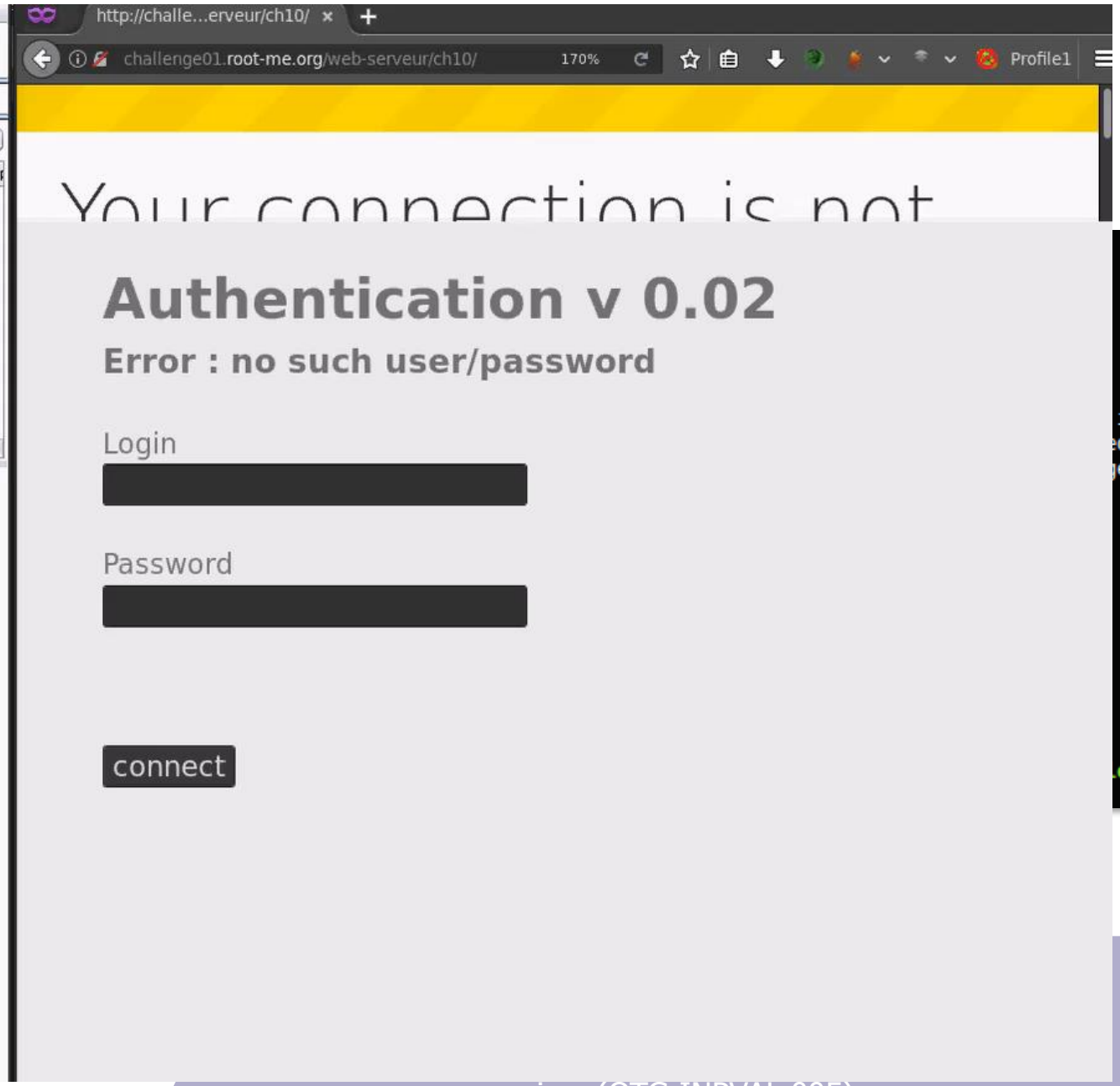
\



JITE

_Inject


Burp Intruder Repeater Window Help
Sequencer
Decoder
Comparer
Extender
Project options
User options
Alerts
Target
Proxy
Spider
Scanner
Intruder
Repeater
Intercept
HTTP history
WebSockets history
Options
Filter: Hiding CSS, image and general binary content
Host Method URL Params Edited Status Length MIME typ


http://challe...erveur/ch10/
challenge01.root-me.org/web-serveur/ch10/
170%
Profile1
Your connection is not
Authentication v 0.02
Error : no such user/password
Login
Password
connect

INJECTION: Database peculiarities

	Oracle	MySQL	MSSQL	PostgreSQL
Concatenate two strings	te'||'st te'%7C%7C'st	te' 'st te'%20'st	te'+'st te'%2B'st	te'||'st te'%7C%7C'st
List users	SELECT username FROM all_users ORDER BY username; SELECT name FROM sys.user$;	SELECT user FROM mysql.user;	SELECT name FROM master..syslogins	SELECT usename FROM pg_user
List databases	SELECT DISTINCT owner FROM all_tables;	SELECT schema_name FROM information_schema.schemata;	SELECT name FROM master..sysdatabases;	SELECT datname FROM pg_database
List columns	SELECT column_name FROM all_tab_columns WHERE table_name = 'blah';	SELECT table_schema, table_name, column_name FROM information_schema.columns	SELECT name FROM syscolumns WHERE id = (SELECT id FROM sysobjects WHERE name = 'mytable');	SELECT relname, A.attname FROM pg_class C, pg_namespace N, pg_attribute A, pg_type T
List tables	SELECT table_name FROM all_tables;	SELECT table_schema,table_name FROM information_schema.tables	SELECT name FROM master..sysobjects WHERE xtype = 'U';	SELECT c.relname FROM pg_catalog.pg_class c LEFT JOIN pg_catalog.pg_namespace n ON n.oid = c.relnamespace
Complementary resources	https://www.owasp.org/index.php/Testing_for_Oracle http://pentestmonkey.net/cheat-sheet/sql-injection/oracle-sql-injection-cheat-sheet	https://www.owasp.org/index.php/Testing_for_MySQL http://pentestmonkey.net/cheat-sheet/sql-injection/mysql-sql-injection-cheat-sheet	https://www.owasp.org/index.php/Testing_for_SQL_Server http://pentestmonkey.net/cheat-sheet/sql-injection/mssql-sql-injection-cheat-sheet	https://www.owasp.org/index.php/OWASP_Backend_Security_Project_Testing_PostgreSQL http://pentestmonkey.net/cheat-sheet/sql-injection/postgres-sql-injection-cheat-sheet

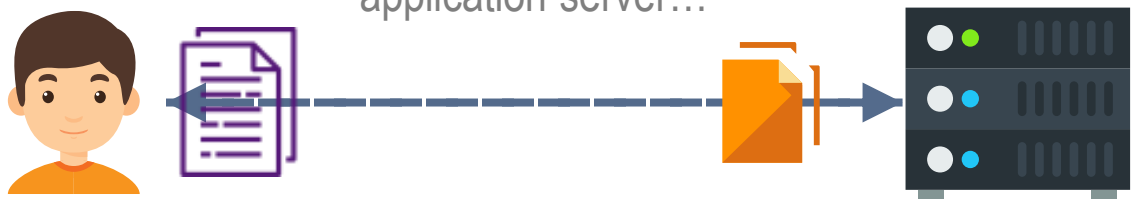
INJECTION: Prevention

- When applicable, only accept a whitelist of values
- If possible, validate input data using regular expressions
- Escape special characters using the escape syntax of the interpreter
- Use prepared statements

CROSS SITE SCRIPTING (XSS): Workflow

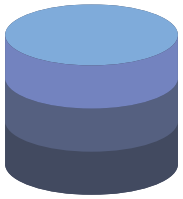
Sometimes, **information provided by the user** are **reused in another page**

The information can be **directly returned** by the application server…



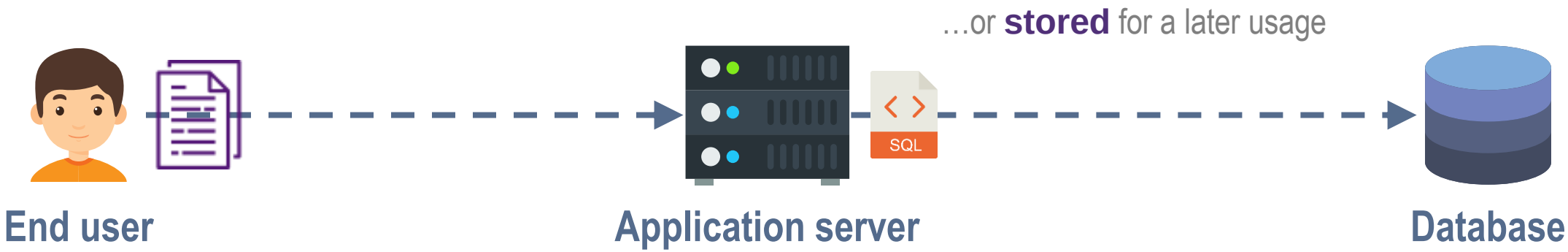
End user

Application server



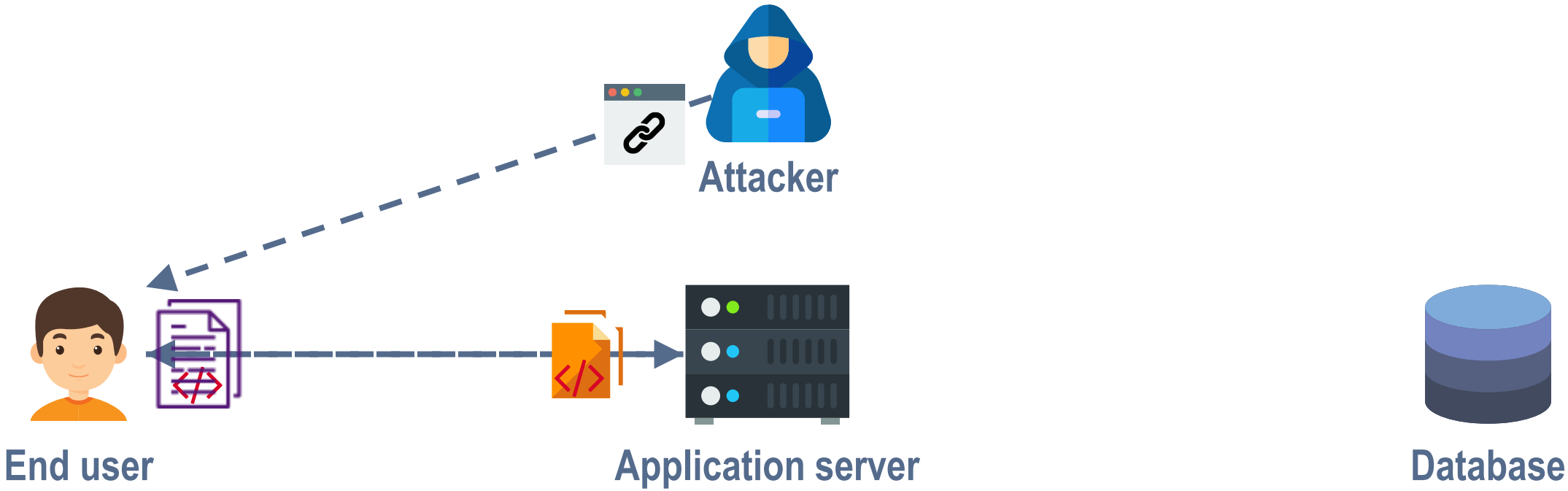
Database

CROSS SITE SCRIPTING (XSS): Workflow



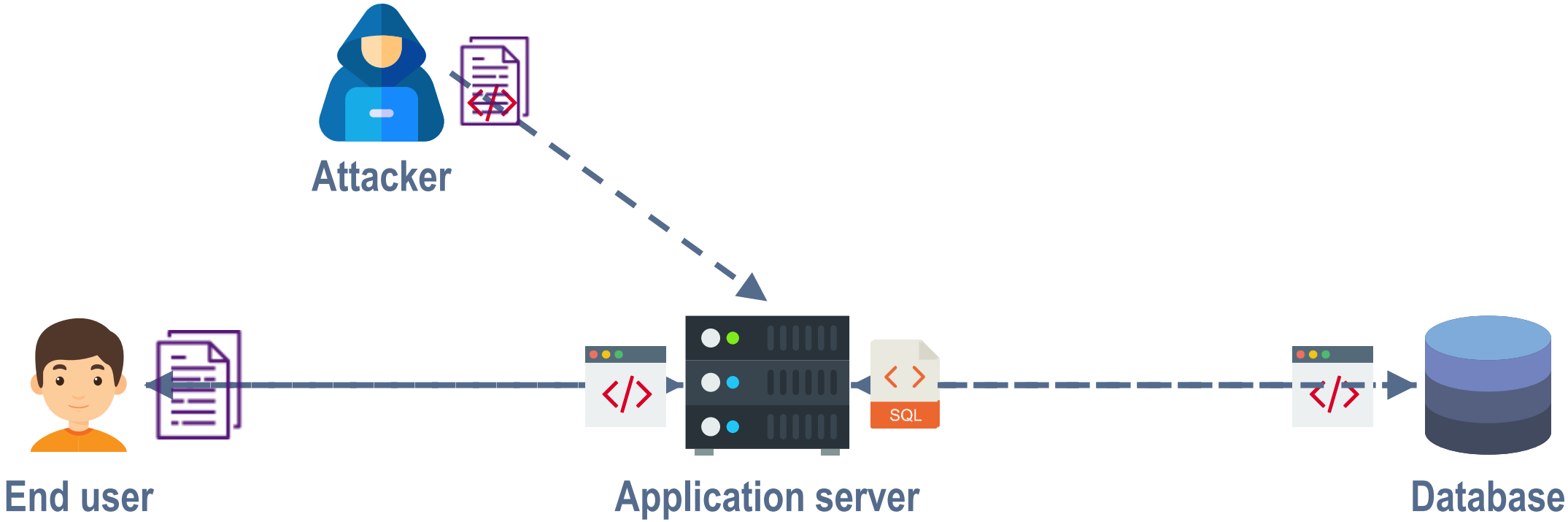
If the application **does not sanitize the parameters before sending them back**, it is possible for the attacker to **inject some code** that will be **executed by the browser**

CROSS SITE SCRIPTING (XSS): Workflow



Either by **tricking the user**…

CROSS SITE SCRIPTING (XSS): Workflow



Or by **injecting malicious data** to the server that will be sent back to **other users**

CROSS SITE SCRIPTING (XSS): Presentation

DEFINITION

XSS is a kind of vulnerability where the user input is **returned without encoding**

There are three types of XSS: **reflected** (the input is only returned for the malicious request), **stored** (the input is stored in database and returned for each access of the page) and **DOM** (the client side code executes differently due to the malicious modification of the DOM environment)

VULNERABLE CASES

Reflected XSS: the application includes unvalidated and unescaped user input as part of the HTML output

Stored XSS: the application stores non-sanitized user input that is viewed at a later time by another user

DOM XSS: a single-page application uses a user input to create a DOM object for the page

What risks if not or poorly done?

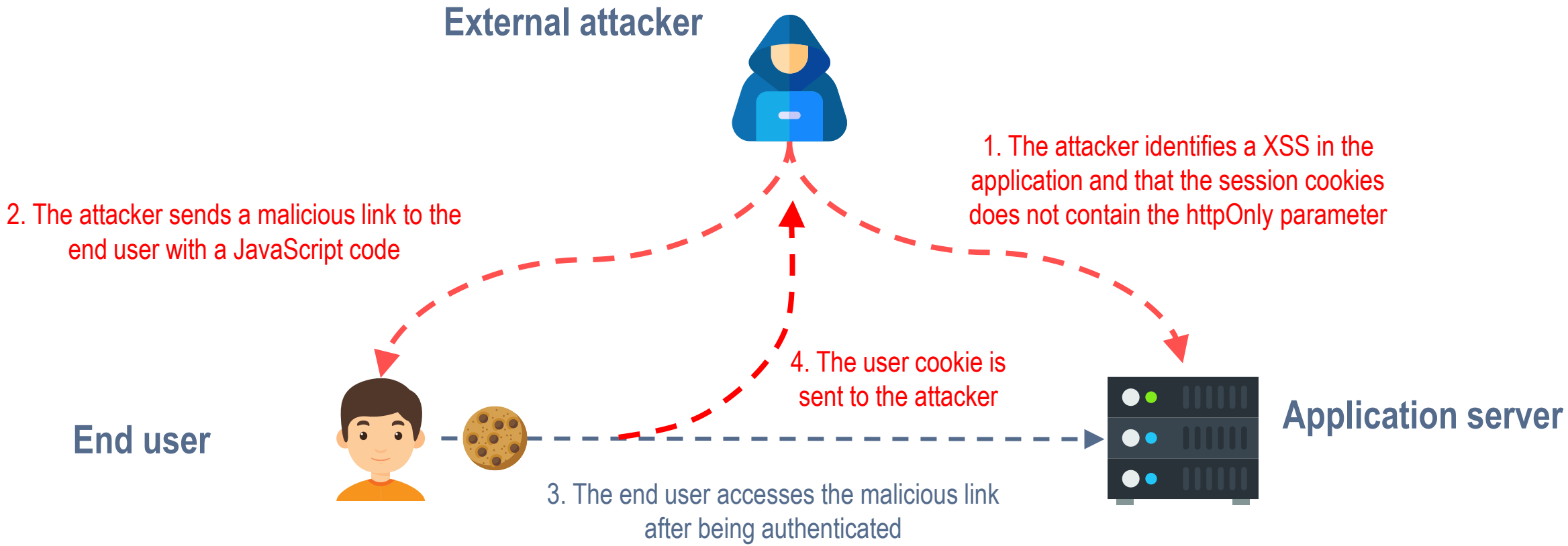
1

Session or identity theft:
If the session cookies are not well protected, an attacker can use a XSS flaw to steal it

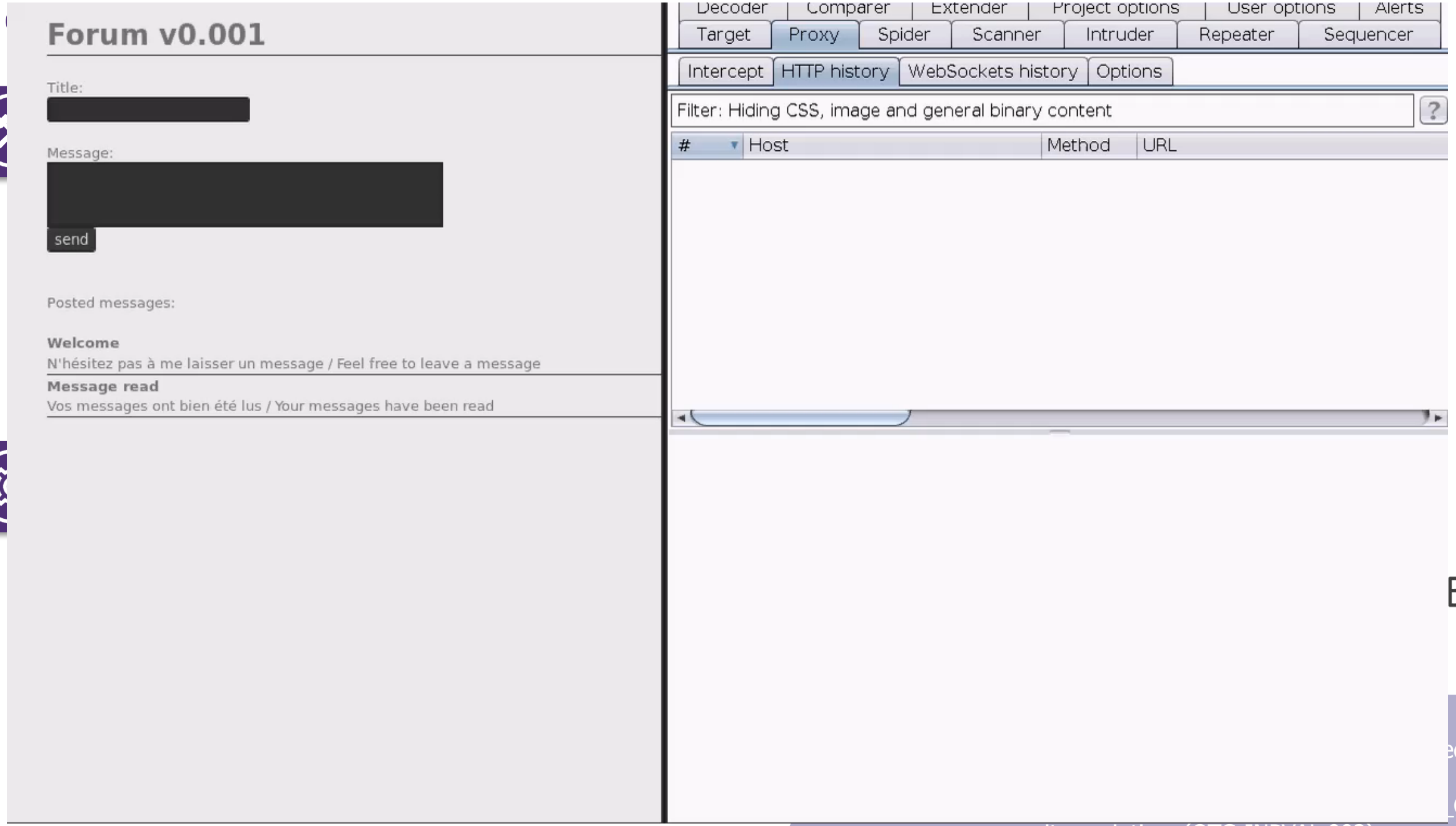
2

Defacement:
A stored XSS can be used to modify the application appearance

CROSS SITE SCRIPTING (XSS): An attack scenario example



Scenario 1: XSS based session stealing


Forum v0.001
Title:
Message:
send
Posted messages:
Welcome
N'hésitez pas à me laisser un message / Feel free to leave a message
Message read
Vos messages ont bien été lus / Your messages have been read
Decoder
Comparer
Extender
Project options
User options
Alerts
Target
Proxy
Spider
Scanner
Intruder
Repeater
Sequencer
Intercept
HTTP history
WebSockets history
Options
Filter: Hiding CSS, image and general binary content
#
Host
Method
URL

CROSS SITE SCRIPTING (XSS): Prevention

- Use frameworks that automatically escape XSS
- Manually escape HTTP request data in the HTML output

PATH TRAVERSAL: Prevention

- Use identifier for file downloads

UNRESTRICTED FILE UPLOAD: Presentation

DEFINITION

An unrestricted file upload flaw occurs when it is possible for un attacker to **get some code to the system**

Then, the attacker only needs to find a way to **get the code executed**

An unrestricted file upload can be the result of the acceptation of a completely **malicious file**, or a legitimate file containing **malicious metadata**

VULNERABLE CASES

Upload a **webshell** to control the Web server

Upload a **huge file** to cause a **file space Denial of Service**

Upload a **virus** to infect the machine

What risks if not or poorly done?

1

Server compromise:
The upload of a webshell or a virus can result in the recuperation of an administrative access

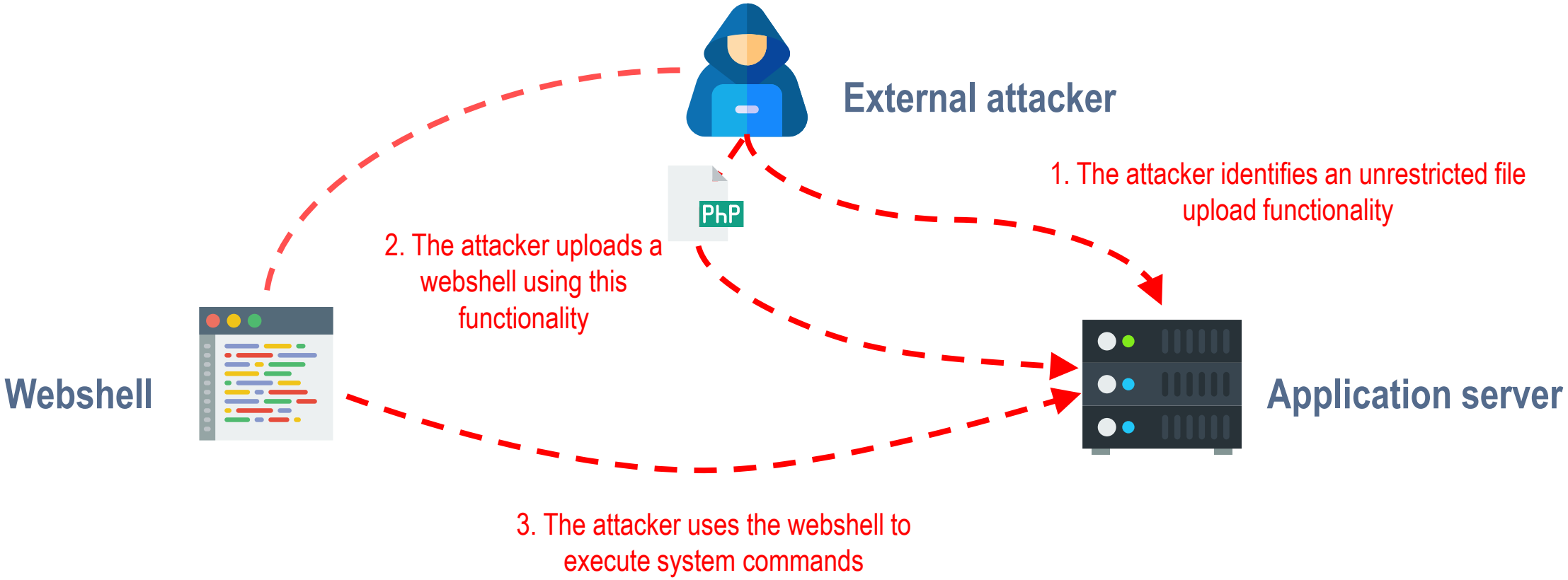
2

Denial of Service:
If the uploaded file are stored in the same partition than the system, the upload of a huge file can result in a Denial of Service of the server

3

Unauthorized resource access:
The execution of controlled code can permit an attacker to access the resources stored in the server

UNRESTRICTED FILE UPLOAD: An attack scenario example



Scenario 1: webshell upload

UNRESTRICTED FILE UPLOAD: Testing methodology

TOOL & METHODOLOGY

Burp Suite:

→ **Unexpected files** can be uploaded by **using directly the Web functionality**

→ The Burp **repeater** can be used on more complex file upload functionality to modify the request in order to **bypass basic protections**

EXAMPLES

- Set your Internet proxy with Burp (`127.0.0.1:8080`)
- Find an upload file request
- Send the request to the repeater (`right click > send to Repeater` or `CTRL + R`) and try to replay it with an unexpected file while changing the "`Content-Type`" in the HTTP request

Complementary resources:
https://www.owasp.org/index.php/Test_Upload_of_Unexpected_File_Types_(OTG-BUSLOGIC-008)
https://www.owasp.org/index.php/Test_Upload_of_Malicious_Files_(OTG-BUSLOGIC-009)

UNRESTRICTED FILE UPLOAD: Prevention

- Whitelist file extensions
- Validate “Content-Type” header
- Verify the MIME type of the uploaded files
- Use an antivirus on the server

/ **04**

Key statistics

HOW COMMON ARE THESE VULNERABILITIES?

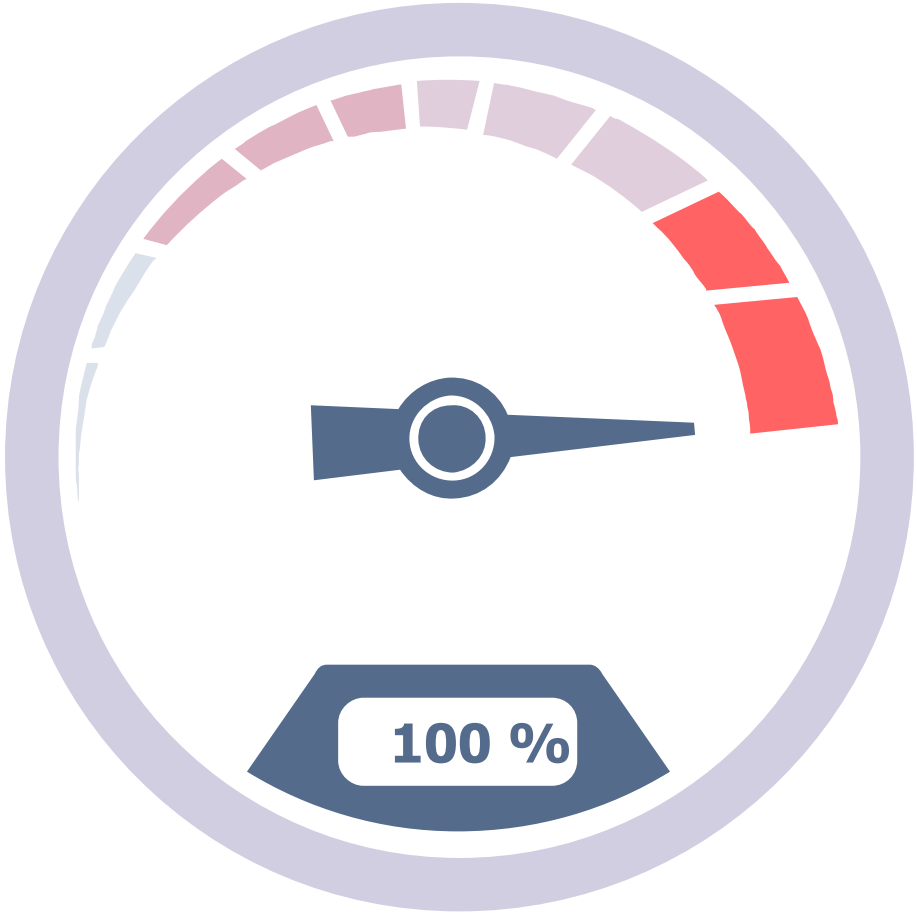
4 highlights about website security

1. **All** tested websites were vulnerable, across all contexts and business sector
2. **50 %** of public (Internet) facing websites had at least one **critical** vulnerability
3. **40 %** of already pentested websites are still vulnerable
4. **90%** Of the tested websites were already online when the audit was performed

4 key design issues relating to critical vulnerabilities

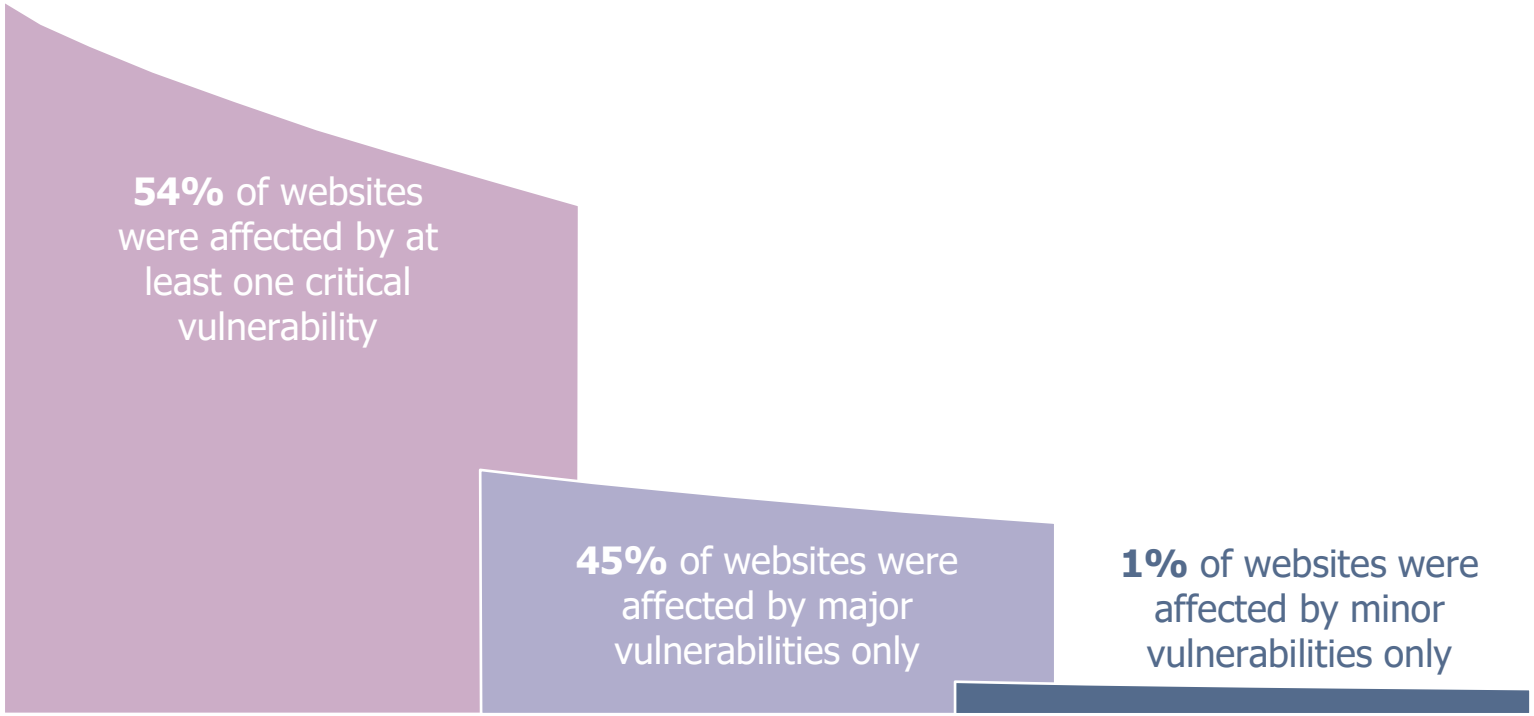
1. **Access control** An attacker can, with a basic user account, access data from all users
2. **File upload** File upload functions frequently allow an attacker to fully take control of a website
3. **Sessions** An attacker can, with an open tab, interact with a website opened in another tab
4. **Language** The development language does not change the amount of vulnerabilities… but their criticality!

ALL TESTED WEBSITES WERE VULNERABLES!



100% On all 128 tested websites, **at least one** vulnerability has been discovered during each test campaign

CRITICAL VULNERABILITES IN MORE THAN HALF OF CASES



Critical vulnerability
Allows full access to website content and/or server compromise

Access to all data from the website, code execution on server, user A accessing data from user B, etc.

Major vulnerability
Allows access other users' data on a reduced scope or only with a complex technique

Session theft, weakness in encryption protocol, unwanted actions performed by users, etc.

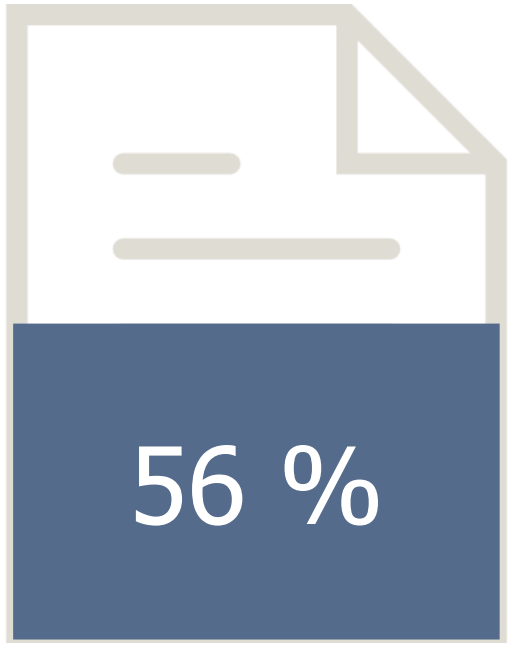
Minor vulnerability
Largely limited to providing more information to allow continued attack

Unnecessary technical messages, unsecure cookies, incomplete disconnection, etc.

AN ACCESS CONTROL NOT ALWAYS UNDER CONTROL…

45% of tests performed in **grey box** mode (where a standard user account was available) demonstrated a bypass access control vulnerability, allowing access to unauthorised data or functions (horizontal or vertical privilege escalation)

FILE UPLOAD? DANGREOUS CROSS ROAD!



In **56%** of cases where a file upload function was available, a vulnerability was identified that allowed custom code to be placed and executed on the web server

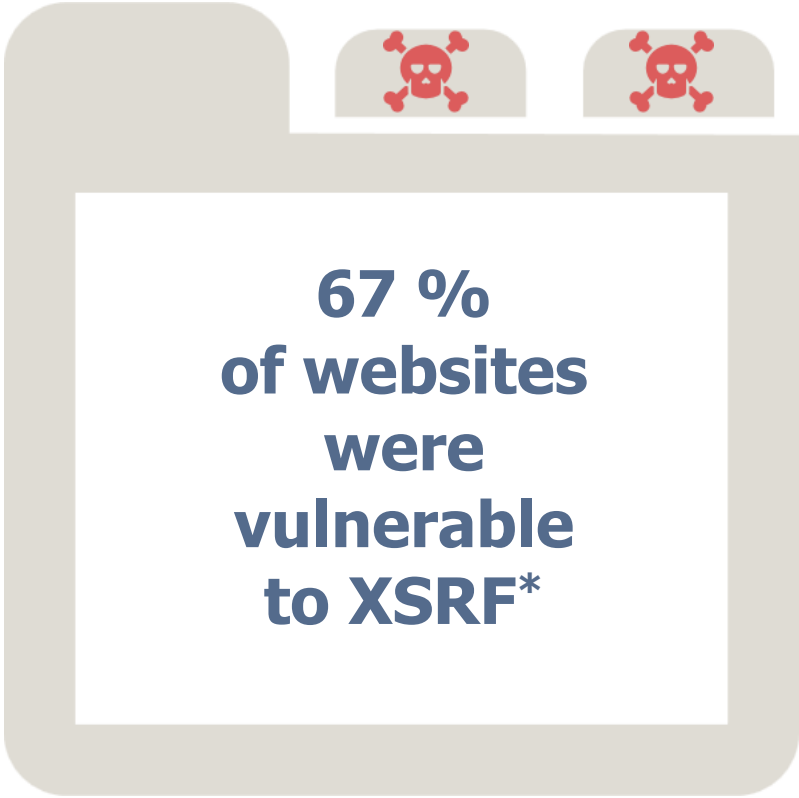
This type of vulnerability provides an attacker with the means to jump from the web server to other information system devices

BROWSING MULTIPLE WEBSITES: A BAD HABIT

1/2 of websites tested were vulnerable to CSRF* (or XSRF*):

- ➔ While using a sensitive website, you decide to open a new browser tab
- ➔ The website in this new tab, if malicious, is capable performing actions without your consent on your sensitive website, such as changing the **email address** used to reset your **password**

**CSRF or XSRF: Cross Site Request Forgery*



/ 05

Security best practices

HOW TO avoid common vulnerabilities?

A1 : Injection	Use prepared statements (parameterized queries) and stored procedures while coding
A2 : Broken authentication and Session Management	Verify the actions requested by the server
A3 : Sensitive Data Exposure	Make sure that no sensitive data is present in the test environment or in the logs
A4 : XML External Entities	Verify that XML or XSL file upload functionality validates incoming XML using XSD validation
A5 : Broken Access Control	Verify access rights on the server side before performing actions
A6 : Security Misconfiguration	Define and enforce hardening guides
A7 : Cross Site Scripting (XSS)	Escape untrusted HTTP request data based on the context in the HTML output
A8 : Insecure Deserialization	Implement integrity checks such as digital signatures on any serialized objects
A9 : Using Unknown Vulnerable Components	Implement patch management
A10 : Insufficient Logging & Monitoring	Ensure all login, access control failures, and server-side input validation failures can be logged

Know the **OWASP Top 10**

WAVESTONE

Timon Glasser
Consultant

M +33 (0)7 63 62 84 16
timon.glasser@wavestone.com

PARIS

LONDON

NEW YORK

HONG KONG

SINGAPORE *

DUBAI *

SAO PAULO *

LUXEMBOURG

MADRID *

MILANO *

BRUSSELS

GENEVA

CASABLANCA

ISTANBUL *

EDINBURGH

LYON

MARSEILLE

NANTES

* Partnerships

WAVESTONE