Department Elective - TY 2023-24

VULNERABILITY ANALYSIS AND PENETRATION TESTING

GUIDED BY : PROF. ASHWINI DALVI

CWE-639 Insecure Direct Object Reference

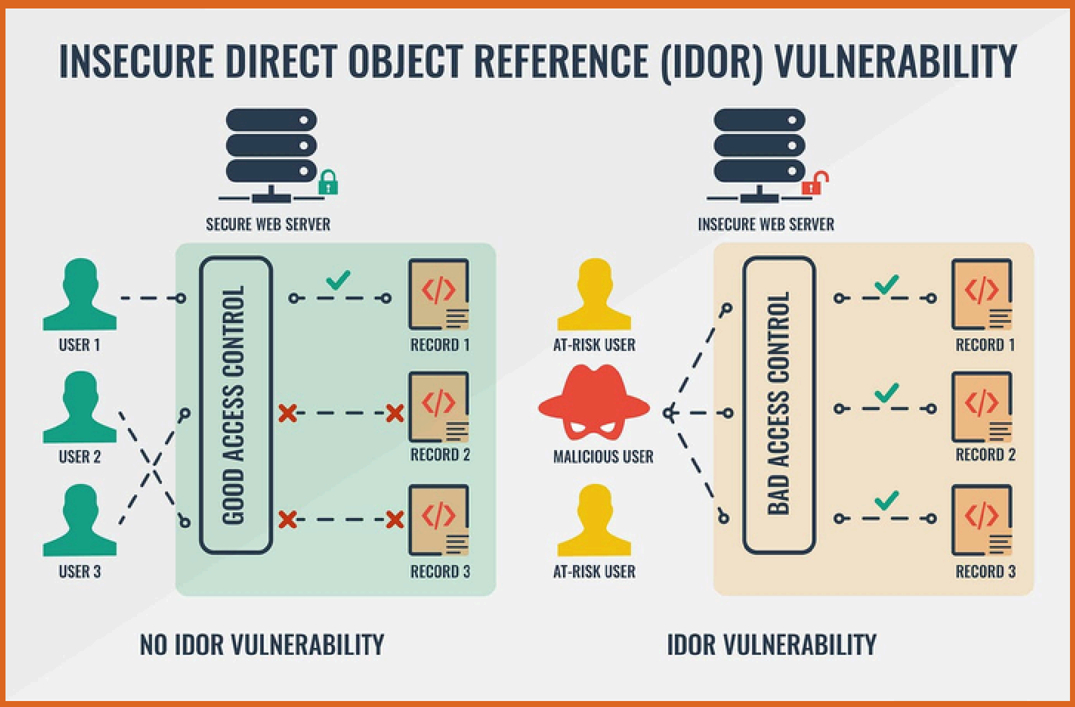
Introduction:

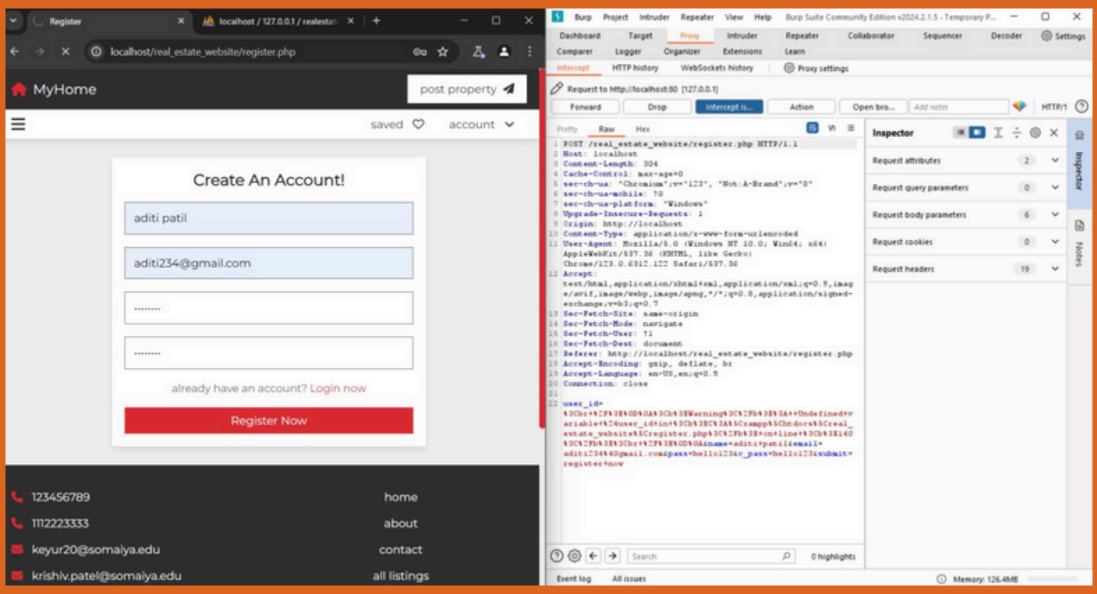
In today's interconnected world, data is the lifeblood of business and personal interactions. However, with great convenience comes great risk. One such peril is Insecure Direct Object Reference (IDOR), a vulnerability that threatens the integrity and confidentiality of your data.

What is IDOR?

IDOR occurs when an application uses user-supplied input to access internal objects or data without proper validation of the user's authorization. This allows attackers to bypass access controls and access sensitive data or resources.

- **Common Scenario:** Attackers manipulate identifiers (e.g., document or account numbers) in URLs or parameters to access data they are not authorized to view. This can lead to data breaches, unauthorized access, and impersonation of users .
- **Impact:** Data breaches, unauthorized access to resources, impersonation of legitimate users, and account takeover are potential consequences of IDOR vulnerabilities




Register
localhost / 127.0.0.1 / realestat
localhost/real_estate_website/register.php
MyHome
post property
saved
account
Create An Account!
already have an account? Login now
Register Now
home
about
contact
all listings
Burp Project Intruder Repeater View Help Burp Suite Community Edition
Dashboard Target Proxy Intruder Repeater Collaborator Sequencer Decoder Settings
Comparer Logger Organizer Extensions Learn
Intercept HTTP history WebSockets history Proxy settings
Forward Drop Intercept is... Action Open bro...
Pretty Raw Hex
Inspector
Request attributes
Request query parameters
Request body parameters
Request cookies
Request headers
Search
0 highlights
Event log All issues
Memory: 126.4MB

aditi patil

aditi234@gmail.com

123456789

1112223333

keyur20@somaiya.edu

krishiv.patel@somaiya.edu

Request to http://localhost:80 [127.0.0.1]

```
POST /real_estate_website/register.php HTTP/1.1
Host: localhost
Content-Length: 304
Cache-Control: max-age=0
sec-ch-ua: "Chromium";v="123", "Not:A-Brand";v="8"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Windows"
Upgrade-Insecure-Requests: 1
Origin: http://localhost
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/123.0.6312.122 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Referer: http://localhost/real_estate_website/register.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Connection: close

user_id=%3Cbr+%2F%3E%0D%0A%3Cb%3EWarning%3C%2Fb%3E%3A++Undefined+variable+%24user_id+in+%3Cb%3EC%3A%5Cxampp%5Chtdocs%5Creal_estate_website%5Cregister.php%3C%2Fb%3E+on+line+%3Cb%3E140%3C%2Fb%3E%3Cbr+%2F%3E%0D%0A&name=aditi+patil&email=aditi234%40gmail.com&pass=hello123&c_pass=hello123&submit=register+now
```

Examples:

- A shopping application using sequential product IDs that an attacker can modify to view other customers' orders.
- A document management system exposing direct references to files that allow unauthorized access by guessing file names.
- A banking application allowing unauthorized fund transfers by modifying account numbers in requests.

Real Estate Listing Platform

We Exposed a vulnerability for IDOR in create account page using burpsuite for email by intercepting a request at proxy and redirecting the submission of form to phpmyadmin database.

Why is it a serious issue?

- ⚠ Violates the principle of least privilege and access control.
- ⚠ Exposes sensitive data like personal records, financial information, credentials.
- ⚠ Enables unauthorized modification, deletion or viewing of data.
- ⚠ Compliance issues (e.g., GDPR, HIPAA).
- ⚠ Reputational damage and loss of trust.

Mitigation

- Use index values or opaque references instead of exposing real object keys.
- Implement proper access control and authorization checks.
- Avoid exposing sensitive information in URLs, forms, or direct object references.

References:

- https://bluegoatcyber.com/blog/insecure-direct-object-references-idor-an-overlooked-web-vulnerability/
- https://medium.com/@vulnerable19/idor-insecure-direct-object-reference-cb4a54fd8a0a

Made By:
Keyur Patel-16010421073
Krishiv Patel-16010421074
IA-2

Department Elective - TY 2023-24

Vulnerability Analysis and Penetration Testing

Guided By : Prof. Ashwini Dalvi

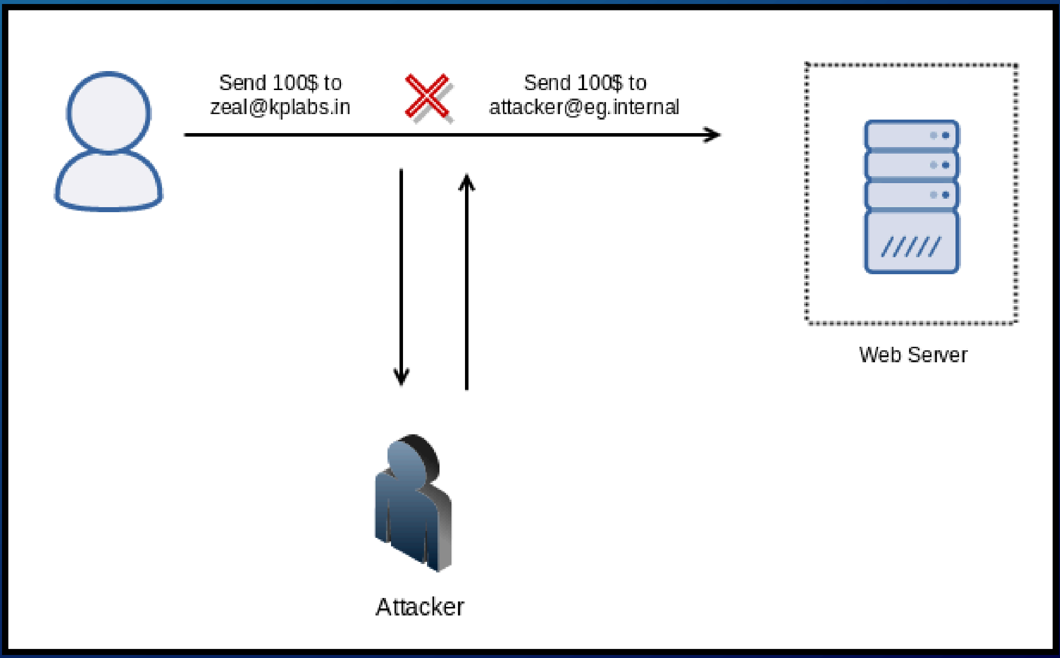
SOMAIYA VIDYAVIHAR UNIVERSITY

K J Somaiya College of Engineering

CWE-353 Missing Support for Integrity Check

INTRODUCTION

The Silent Threat Lurking in Your Systems In the digital realm, data is king. Yet, without proper safeguards, your precious information can be silently compromised, leaving you blind to the intrusion. CWE-353, or Missing Support for Integrity Check, is a vulnerability that quietly allows data tampering, code injection, and unauthorized modifications to occur unchecked.


Send 100$ to zeal@kplabs.in
Send 100$ to attacker@eg.internal
Web Server
Attacker

What is CWE-353?

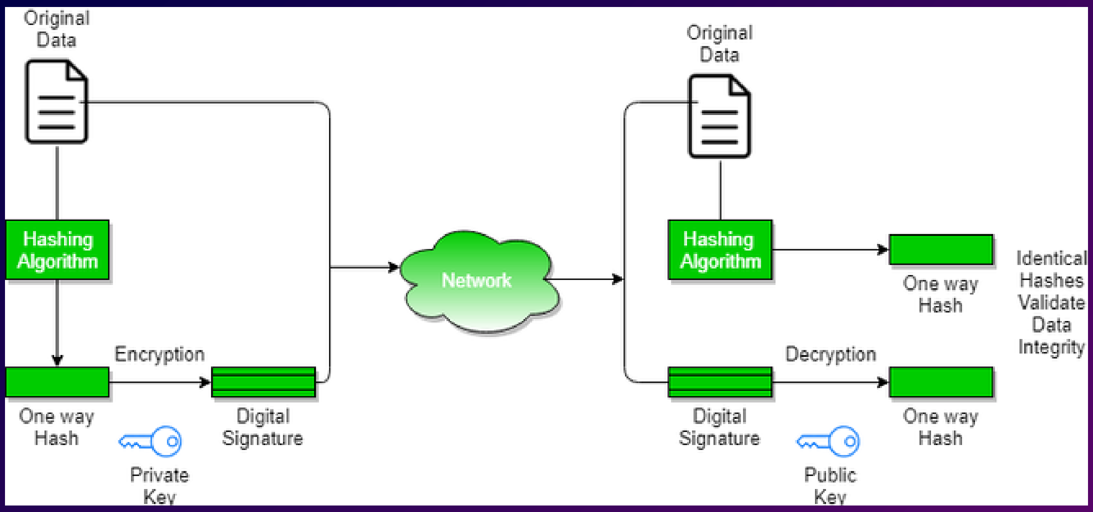
- CWE-353, or Missing Support for Integrity Check, is a software vulnerability that occurs when data or instructions are transmitted or received without verifying their integrity.
- This can lead to various attacks, such as data tampering, code injection, or unauthorized modifications.

Examples:

- Unencrypted file transfers without integrity checks, allowing file content tampering.
- Software updates downloaded without verifying integrity, enabling injection of malicious code.

The Consequences of Ignoring Integrity

- Data corruption: Your information loses its sanctity, rendering it unreliable and potentially dangerous.
- Malicious code execution: Attackers can inject malicious code, granting them unauthorized access and control.
- Unauthorized modifications: Sensitive data can be altered without your knowledge, leading to disastrous consequences.



Digital signatures provide data integrity and authentication by cryptographically binding a sender's identity to the data. To ensure integrity:

1. Sender computes hash (e.g., SHA-256) of data.
2. Sender encrypts hash with their private key, creating a digital signature.
3. Signature is sent with the data.

Receiver verifies integrity by:

1. Computing hash of received data.
2. Decrypting signature with sender's public key to get original hash.
3. Comparing hashes - match confirms data was not modified.

Mitigation:

- Implement cryptographic hash functions (e.g., SHA-256, SHA-3)
- Use digital signatures or message authentication codes (MACs)
- Verify data integrity at both the sender and receiver ends
- Establish secure communication channels (e.g., TLS, HTTPS)
- Regularly update and patch your systems and applications
-

References:

- https://bluegoatcyber.com/blog/insecure-direct-object-references-idor-an-overlooked-web-vulnerability/
- https://medium.com/@vulnerable19/idor-insecure-direct-object-reference-cb4a54fd8a0a

Made By:

Keyur Patel-16010421073

Krishiv Patel-16010421074

IA-2