Cloud security challenge VPBank Hackathon

Lê Quang Anh – nhóm 132

lequanganh97@gmail.com

Agenda

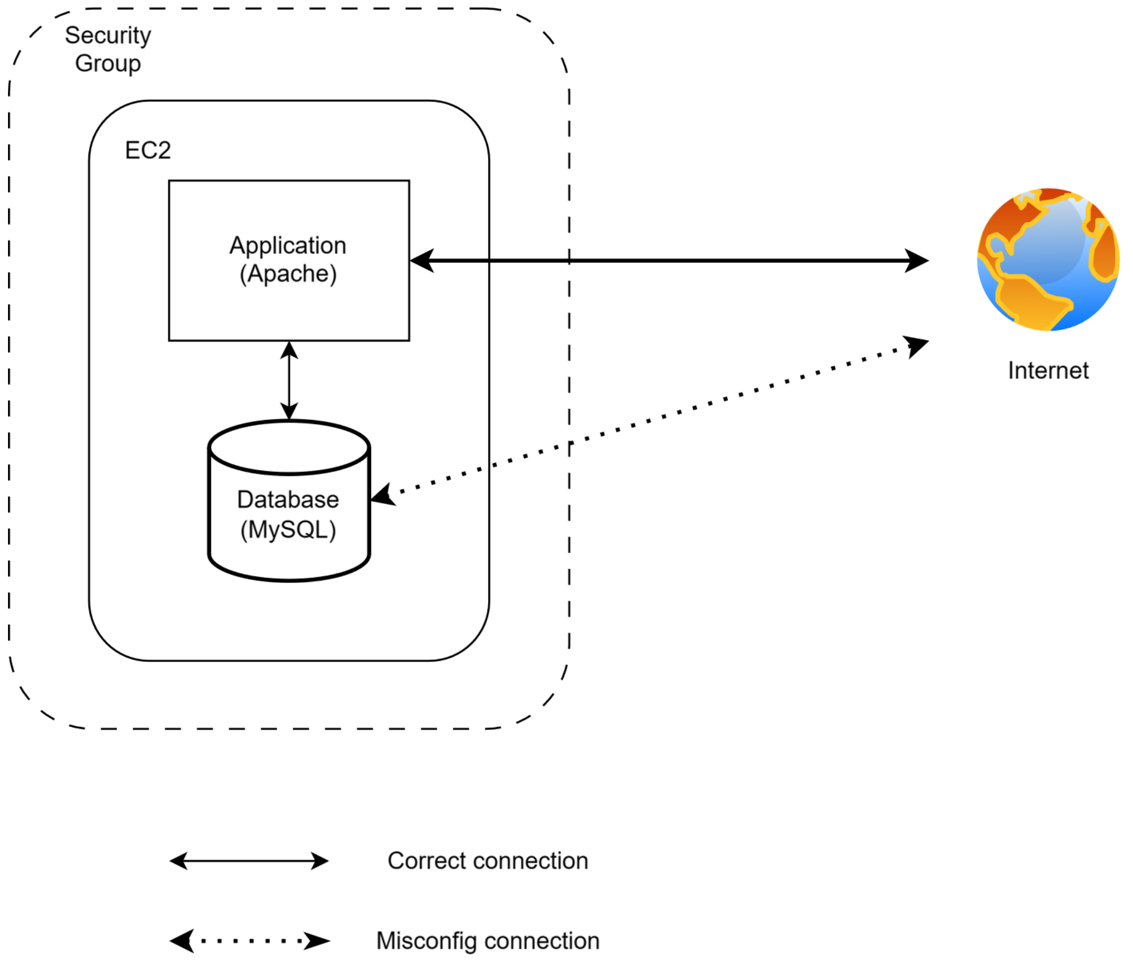
- Secure Cloud Web Apllication challenge (Threat, Solution, Architecture)
- IaC Solution
- Secure Cloud Web Apllication Security Solutions
- Demo

Secure Cloud Web Apllication challenge

- **Application**: Public information board web application with build-in network tool
- **Functions:**
  - Create post public message for other user to read.
  - Create private message for note.
  - Network tools (nslookup, dig, ping)
- **IaC:**
  - Application deploy using Ansible
- **Vulnerability:**
  - SQL injection (Owasp top 10 2021 – A03:2021)
  - Cryptography Failures (Owasp top 10 2021 – A02:2021)
  - Software and data integrity failures (Owasp top 10 2021 – A08:2021)
  - IDOR (Owasp top 10 2021 – A01:2021)
  - Security misconfiguration (Owasp top 10 2021 – A05:2021)
  - Remote Code Execution (Owasp top 10 2021 – A03:2021)

Insecure Web Application Architecture

- **Architecture**
    - Application build with PHP running on Apache server
    - Application using a SQL database to store user(s) login data, and user(s) post(s).
- **Deployment**
    - Web application deployed on a single EC2 machine on AWS with Security group set to allow traffic SSH, HTTP, HTTPS and MySQL



Vulnerabilities

- **Security misconfiguration**: When deploy to AWS, the VM running with misconfigured security group leading to increment of application attack surface.

- **SQL injection**: Application query on the database without validate the user untrusted inputs which cause SQL injection.

- **Cryptography Failures**: Application encrypt user password and store its inside database but using a weak algorithm like md4, md5 which is easy to crack.

- **Software and data integrity failures** : Application failure to validate the source of interaction to the app, making its possible for user to create data on different user account.

- **IDOR**: When handle information request, application failed to validate whether the request is from correct user which lead to user can read data of other user

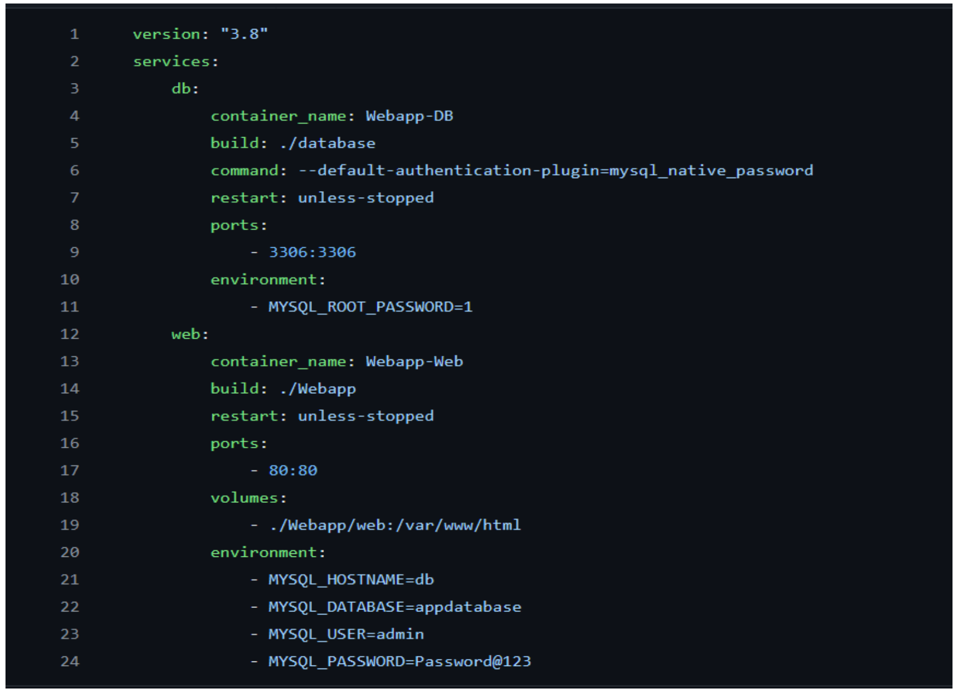
- **Remote code execution:** Application used “Shell_exec” function in php without filter input which leads to attack be able to execute code on the server, even able to create a web shell.

Solutions to vulnerability in the web application

Function	Vulnerability	Solution	Referance
Login	SQL injection	Validate user input and escape, filter out dangerous input	A03 Injection - OWASP Top 10:2021
User ID	Cryptography Failure	Use new and robust cryptographic algorithm	A02 Cryptographic Failures - OWASP Top 10:2021
Create post	Software and data integrity failures	Validate the source of the data before accept it into the system	A08 Software and Data Integrity Failures - OWASP Top 10:2021
Read post	Insecure direct object reference (IDOR)	Validating request that access user data to make sure user can only read what it available to them	A01 Broken Access Control - OWASP Top 10:2021
Network tool	Remote Code Execution	Validate user input and escape, filter out dangerous input.	A03 Injection - OWASP Top 10:2021
	Security misconfiguration	There is no automated solutions to mitigate or prevent security misconfiguration. There can only code review and rigid checking procedure of code can mitigate this vulnerable.	A05 Security Misconfiguration - OWASP Top 10:2021

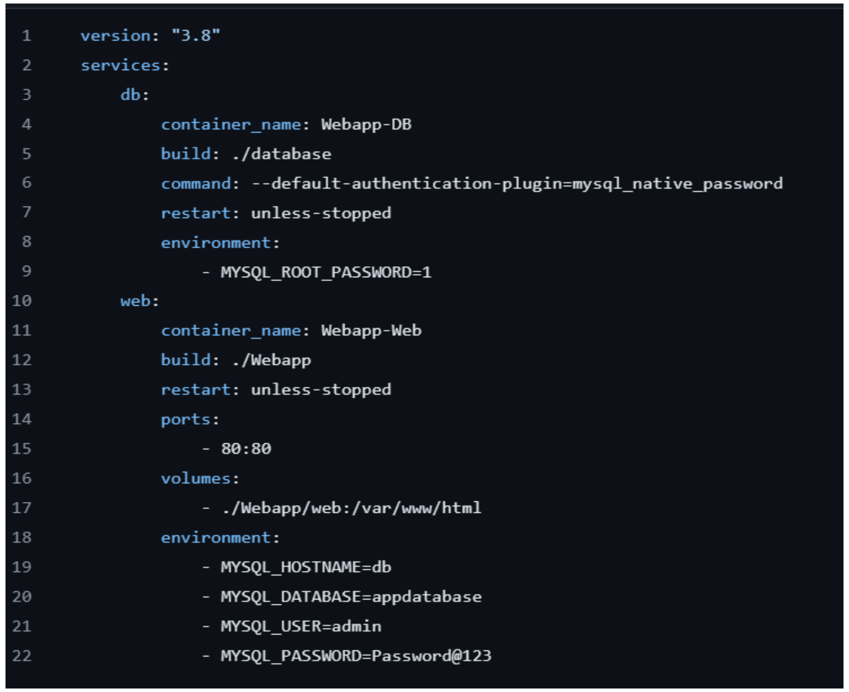
Security solutions to defend web application

- **Security misconfiguration**



```yaml
version: "3.8"
services:
    db:
        container_name: Webapp-DB
        build: ./database
        command: --default-authentication-plugin=mysql_native_password
        restart: unless-stopped
        ports:
            - 3306:3306
        environment:
            - MYSQL_ROOT_PASSWORD=1
    web:
        container_name: Webapp-Web
        build: ./Webapp
        restart: unless-stopped
        ports:
            - 80:80
        volumes:
            - ./Webapp/web:/var/www/html
        environment:
            - MYSQL_HOSTNAME=db
            - MYSQL_DATABASE=appdatabase
            - MYSQL_USER=admin
            - MYSQL_PASSWORD=Password@123
```

Before: Unnecessary port of database was exposed. This port can be detect give NMAP giving threat actor insight to the system

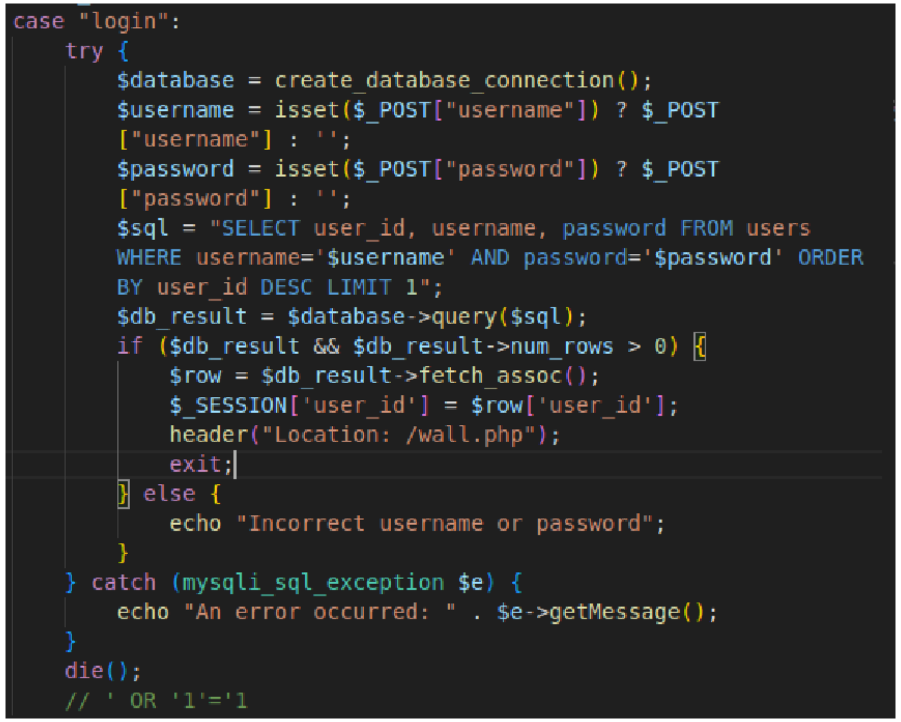


```yaml
version: "3.8"
services:
    db:
        container_name: Webapp-DB
        build: ./database
        command: --default-authentication-plugin=mysql_native_password
        restart: unless-stopped
        environment:
            - MYSQL_ROOT_PASSWORD=1
    web:
        container_name: Webapp-Web
        build: ./Webapp
        restart: unless-stopped
        ports:
            - 80:80
        volumes:
            - ./Webapp/web:/var/www/html
        environment:
            - MYSQL_HOSTNAME=db
            - MYSQL_DATABASE=appdatabase
            - MYSQL_USER=admin
            - MYSQL_PASSWORD=Password@123
```

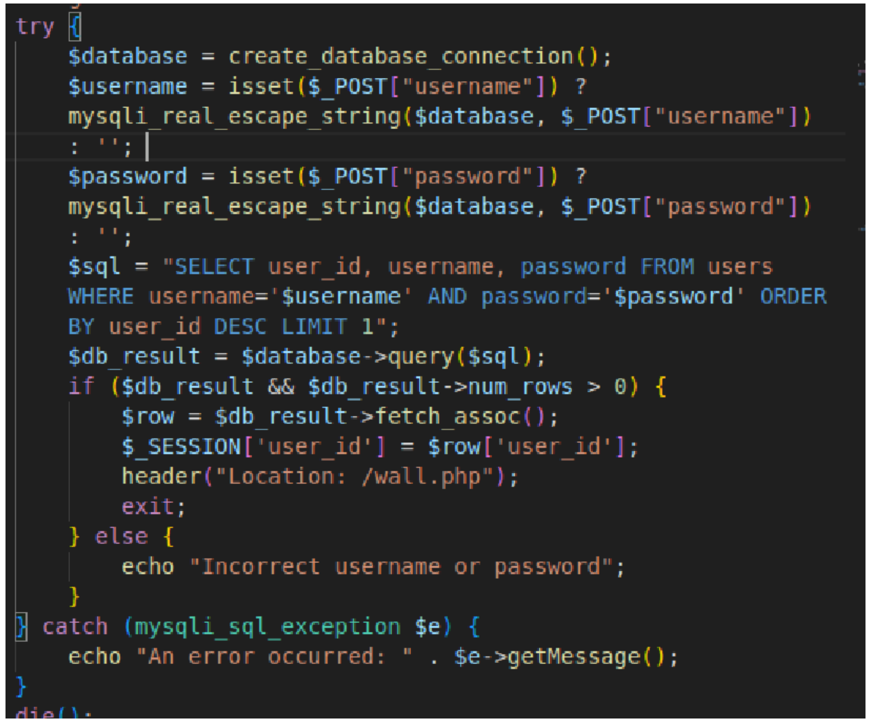
After: Remove unnecessary port

Security solutions to defend web application

- **SQL injection**



```
case "login":
    try {
        $database = create_database_connection();
        $username = isset($_POST["username"]) ? $_POST
        ["username"] : '';
        $password = isset($_POST["password"]) ? $_POST
        ["password"] : '';
        $sql = "SELECT user_id, username, password FROM users
        WHERE username='$username' AND password='$password' ORDER
        BY user_id DESC LIMIT 1";
        $db_result = $database->query($sql);
        if ($db_result && $db_result->num_rows > 0) {
            $row = $db_result->fetch_assoc();
            $_SESSION['user_id'] = $row['user_id'];
            header("Location: /wall.php");
            exit;
        } else {
            echo "Incorrect username or password";
        }
    } catch (mysqli_sql_exception $e) {
        echo "An error occurred: " . $e->getMessage();
    }
    die();
    // ' OR '1'='1
```



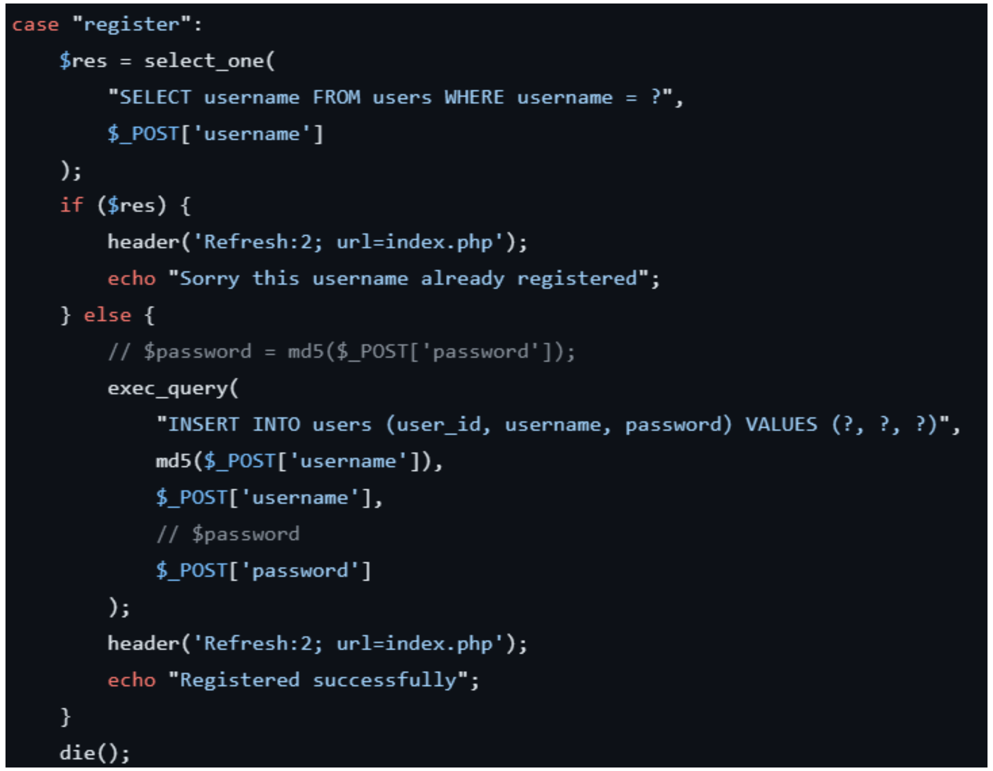
```
try {
    $database = create_database_connection();
    $username = isset($_POST["username"]) ?
    mysqli_real_escape_string($database, $_POST["username"])
    : '';
    $password = isset($_POST["password"]) ?
    mysqli_real_escape_string($database, $_POST["password"])
    : '';
    $sql = "SELECT user_id, username, password FROM users
    WHERE username='$username' AND password='$password' ORDER
    BY user_id DESC LIMIT 1";
    $db_result = $database->query($sql);
    if ($db_result && $db_result->num_rows > 0) {
        $row = $db_result->fetch_assoc();
        $_SESSION['user_id'] = $row['user_id'];
        header("Location: /wall.php");
        exit;
    } else {
        echo "Incorrect username or password";
    }
} catch (mysqli_sql_exception $e) {
    echo "An error occurred: " . $e->getMessage();
}
```

Before: No user input validation

After: Using mysql_real_escape_string to filter out all special character from the input to block SQL injection attack

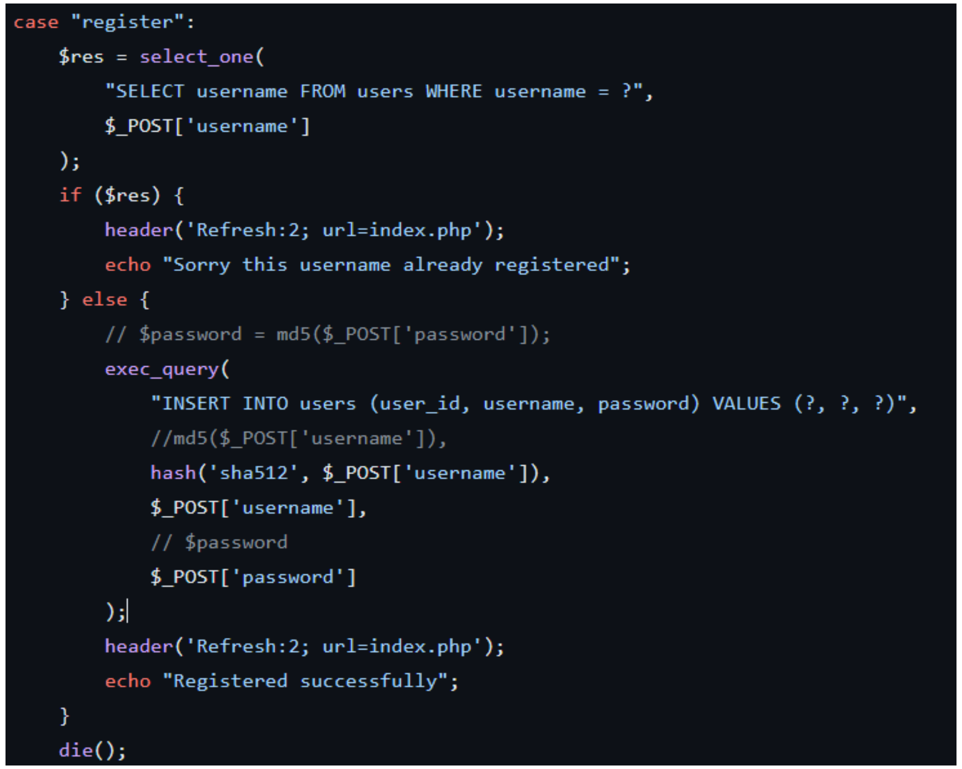
Security solutions to defend web application

- **Cryptography Failures**



```
case "register":
    $res = select_one(
        "SELECT username FROM users WHERE username = ?",
        $_POST['username']
    );
    if ($res) {
        header('Refresh:2; url=index.php');
        echo "Sorry this username already registered";
    } else {
        // $password = md5($_POST['password']);
        exec_query(
            "INSERT INTO users (user_id, username, password) VALUES (?, ?, ?)",
            md5($_POST['username']),
            $_POST['username'],
            // $password
            $_POST['password']
        );
        header('Refresh:2; url=index.php');
        echo "Registered successfully";
    }
    die();
```

Before: User ID created by hashing username with md5 (weak crypto algorithm)

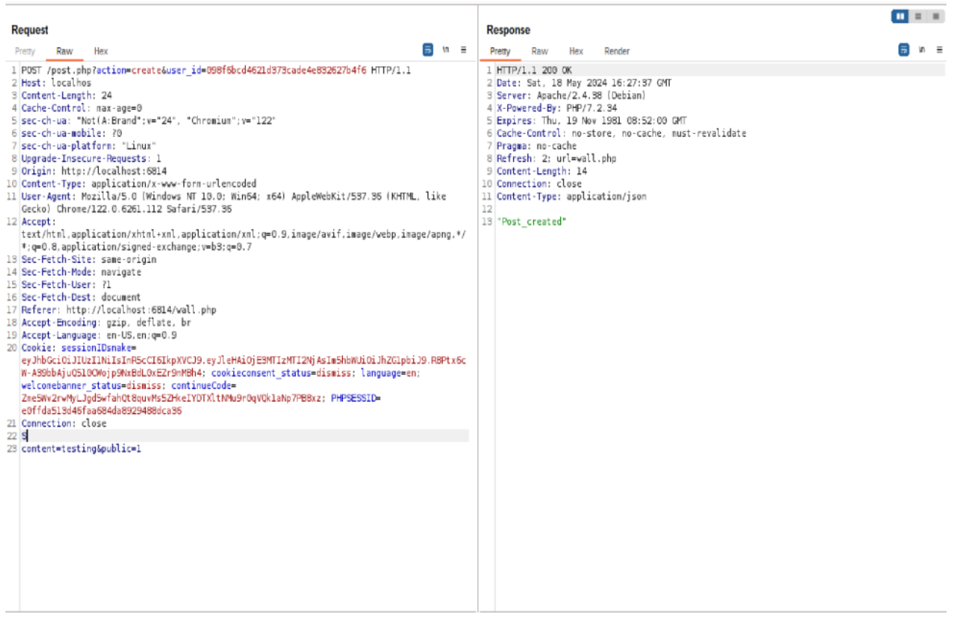


```
case "register":
    $res = select_one(
        "SELECT username FROM users WHERE username = ?",
        $_POST['username']
    );
    if ($res) {
        header('Refresh:2; url=index.php');
        echo "Sorry this username already registered";
    } else {
        // $password = md5($_POST['password']);
        exec_query(
            "INSERT INTO users (user_id, username, password) VALUES (?, ?, ?)",
            //md5($_POST['username']),
            hash('sha512', $_POST['username']),
            $_POST['username'],
            // $password
            $_POST['password']
        );
        header('Refresh:2; url=index.php');
        echo "Registered successfully";
    }
    die();
```

After: Hashing using SHA512 algorithm

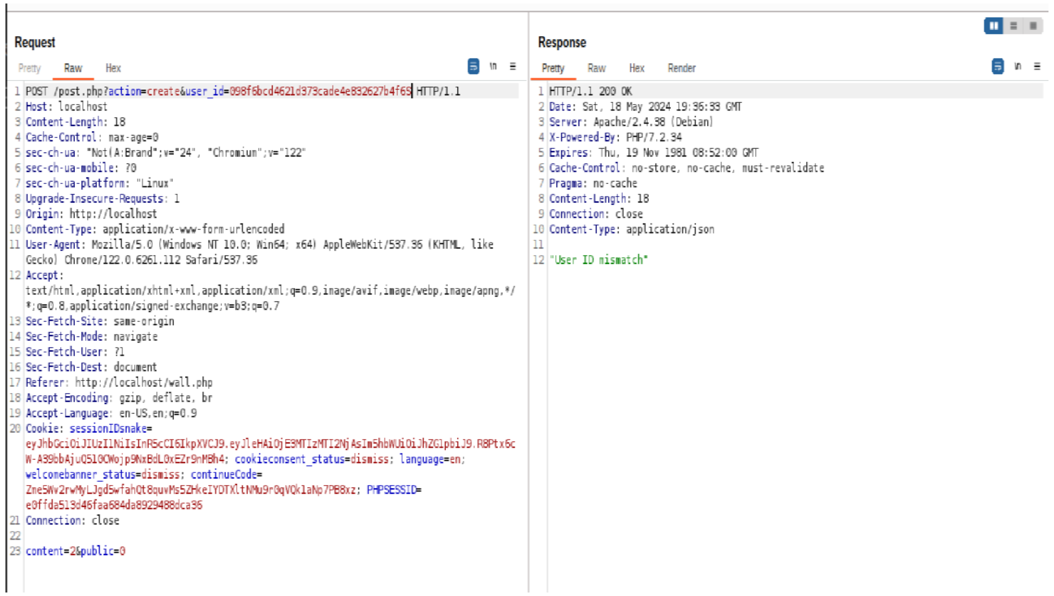
Security solutions to defend web application

- **Software and data integrity failures**



```
POST /post.php?action=create&user_id=098f6bcd4621d373cade4e832627b4f6 HTTP/1.1
Host: localhos
Content-Length: 24
Cache-Control: max-age=0
sec-ch-ua: "Not(A:Brand";v="24", "Chromium";v="122"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Linux"
Upgrade-Insecure-Requests: 1
Origin: http://localhost:6814
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/122.0.6261.112 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:6814/wall.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: sessionIDsnake=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE3MTIzMTI2NjAsIm5hbWUiOiJhZG1pbiJ9.R8Ptx6c
W-A39bbAjuQ51OCWojp9NxBdL0xEZr9nMBh4; cookieconsent_status=dismiss; language=en;
welcomebanner_status=dismiss; continueCode=
Zme5Wv2rwMyLJgd5wfahQt8quvMs5ZHkeIYDTXltNMu9r0qVQk1aNp7PB8xz; PHPSESSID=
e0ffda513d46faa684da8929488dca36
Connection: close
S

content=testing&public=1
```

```
HTTP/1.1 200 OK
Date: Sat, 18 May 2024 16:27:37 GMT
Server: Apache/2.4.38 (Debian)
X-Powered-By: PHP/7.2.34
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Refresh: 2; url=wall.php
Content-Length: 14
Connection: close
Content-Type: application/json

"Post_created"
```



```
POST /post.php?action=create&user_id=098f6bcd4621d373cade4e832627b4f65 HTTP/1.1
Host: localhost
Content-Length: 18
Cache-Control: max-age=0
sec-ch-ua: "Not(A:Brand";v="24", "Chromium";v="122"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Linux"
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/122.0.6261.112 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/wall.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: sessionIDsnake=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE3MTIzMTI2NjAsIm5hbWUiOiJhZG1pbiJ9.R8Ptx6c
W-A39bbAjuQ51OCWojp9NxBdL0xEZr9nMBh4; cookieconsent_status=dismiss; language=en;
welcomebanner_status=dismiss; continueCode=
Zme5Wv2rwMyLJgd5wfahQt8quvMs5ZHkeIYDTXltNMu9r0qVQk1aNp7PB8xz; PHPSESSID=
e0ffda513d46faa684da8929488dca36
Connection: close


content=2&public=0
```

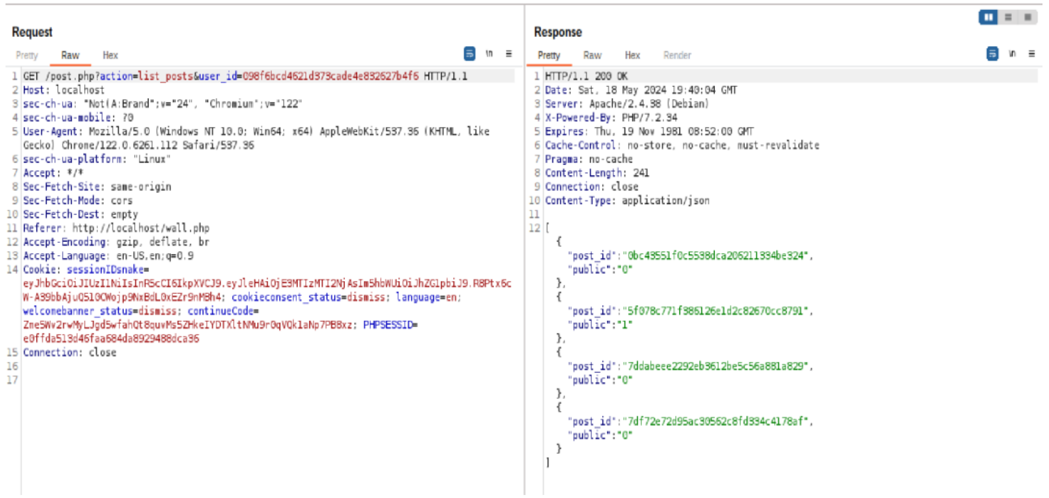
```
HTTP/1.1 200 OK
Date: Sat, 18 May 2024 19:36:33 GMT
Server: Apache/2.4.38 (Debian)
X-Powered-By: PHP/7.2.34
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Length: 18
Connection: close
Content-Type: application/json

"User ID mismatch"
```

Before: By adding other user_id in the request, its possible to create post credited to other user

After: Checking session ID along side with user_id in the request to make sure request created by the correct user

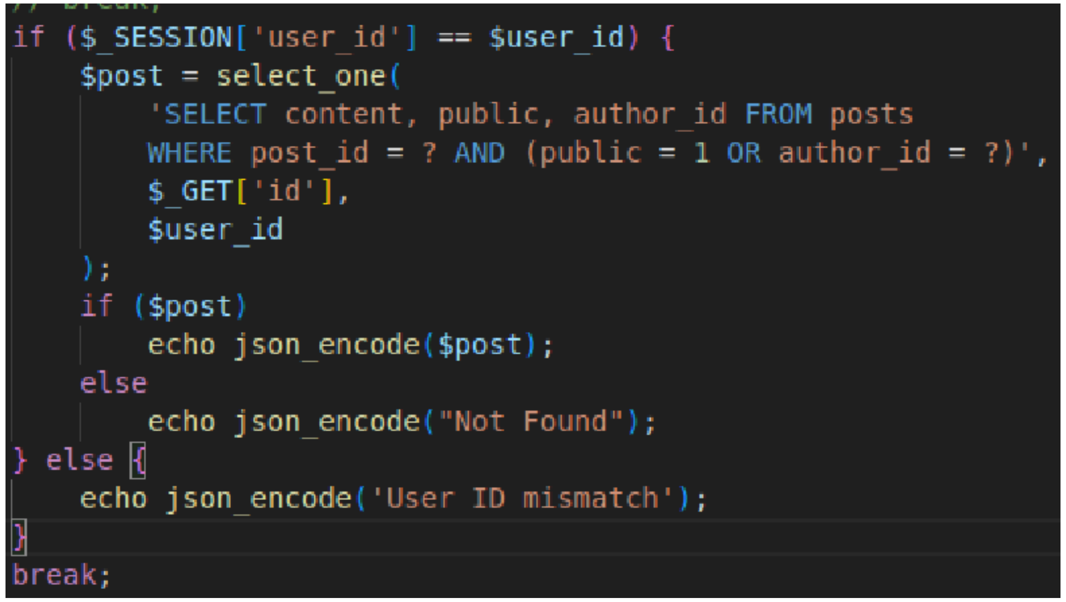
Security solutions to defend web application

- **IDOR**



```
GET /post.php?action=list_posts&user_id=098f6bcd4621d373cade4e832627b4f6 HTTP/1.1
Host: localhost
sec-ch-ua: "Not(A:Brand";v="24", "Chromium";v="122"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/122.0.6261.112 Safari/537.36
sec-ch-ua-platform: "Linux"
Accept: */*
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: cors
Sec-Fetch-Dest: empty
Referer: http://localhost/wall.php
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: sessionIDsnake=
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJleHAiOjE3MTIzMTI2NjAsIm5hbWUiOiJhZG1pbiJ9.R8Ptx6c
W-A39bbAjuQ510CWojp9NxBdL0xEZr9nMBh4; cookieconsent_status=dismiss; language=en;
welcomebanner_status=dismiss; continueCode=
Zne5Wv2rwMyLJgd5wfahQt8quvMs5ZHkeIYDTXltNMu9r0qVQk1aNp7PB8xz; PHPSESSID=
e0ffda513d46faa684da8929488dca36
Connection: close
```

```
HTTP/1.1 200 OK
Date: Sat, 18 May 2024 19:40:04 GMT
Server: Apache/2.4.38 (Debian)
X-Powered-By: PHP/7.2.34
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate
Pragma: no-cache
Content-Length: 241
Connection: close
Content-Type: application/json

[
  {
    "post_id":"0bc43551f0c5538dca206211334be324",
    "public":"0"
  },
  {
    "post_id":"5f078c771f386126e1d2c82670cc8791",
    "public":"1"
  },
  {
    "post_id":"7ddabeee2292eb3612be5c56a881a829",
    "public":"0"
  },
  {
    "post_id":"7df72e72d95ac30562c8fd334c4178af",
    "public":"0"
  }
]
```

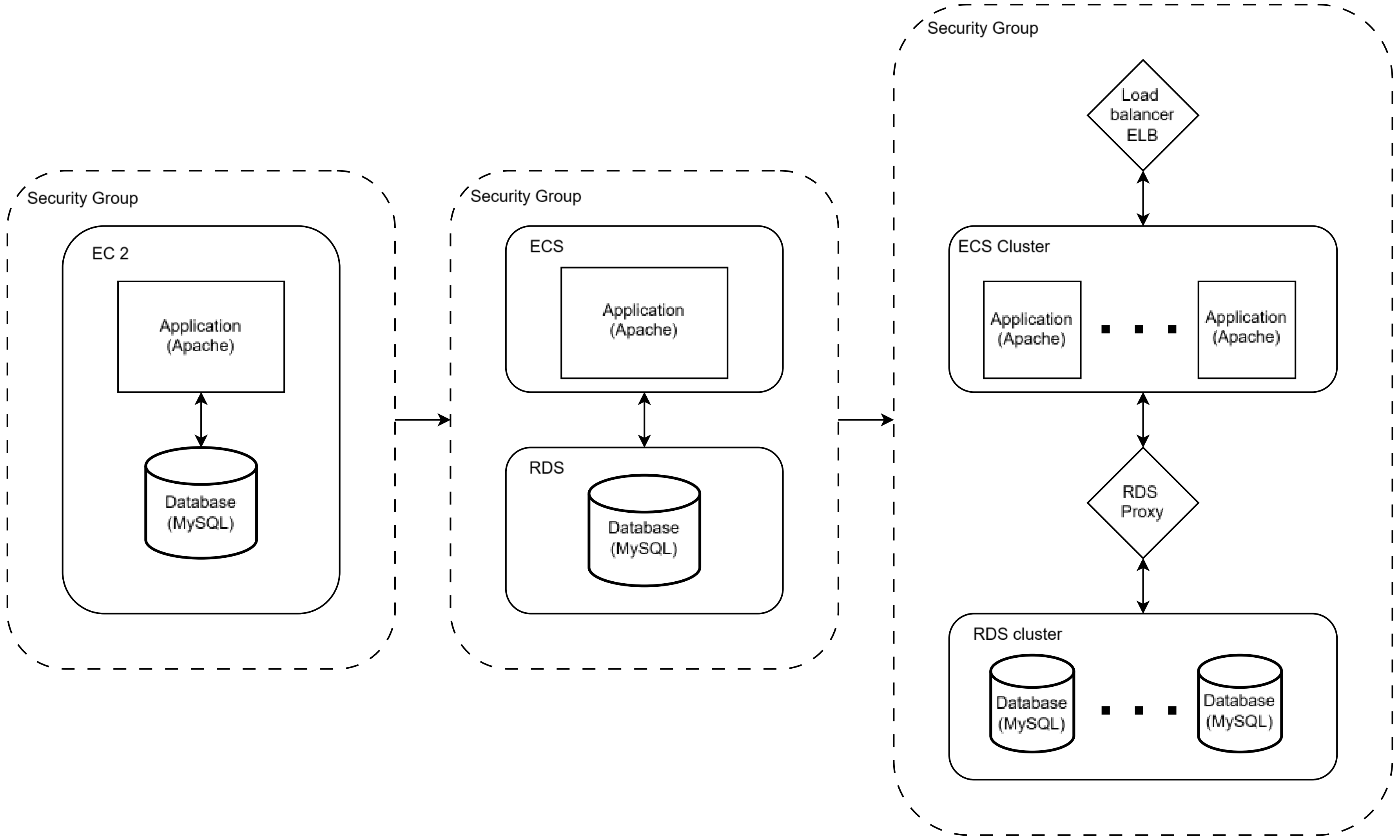


```php
if ($_SESSION['user_id'] == $user_id) {
    $post = select_one(
        'SELECT content, public, author_id FROM posts
        WHERE post_id = ? AND (public = 1 OR author_id = ?)',
        $_GET['id'],
        $user_id
    );
    if ($post)
        echo json_encode($post);
    else
        echo json_encode("Not Found");
} else {
    echo json_encode('User ID mismatch');
}
break;
```

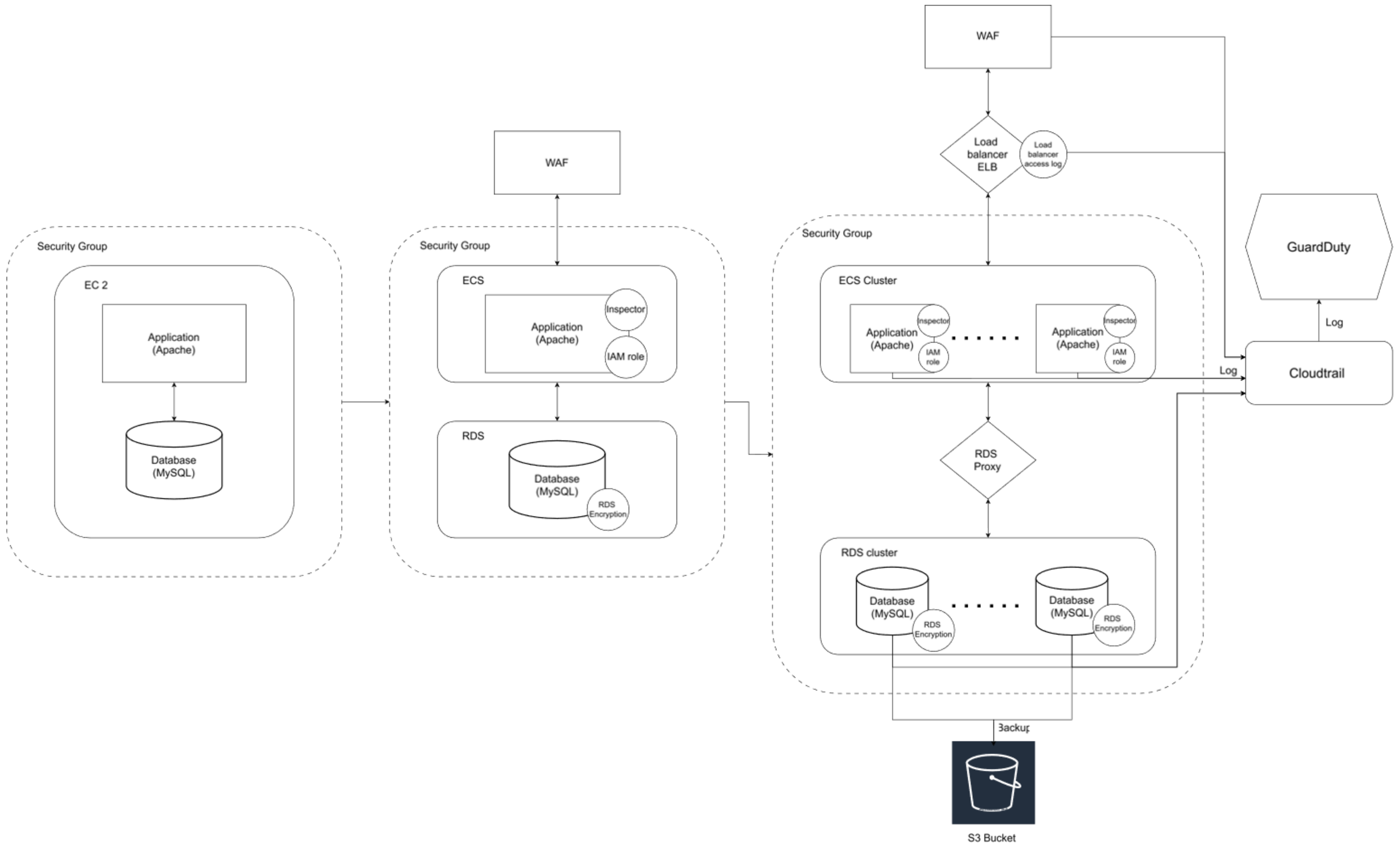
Before: By adding other user_id in the request, its possible to view id of all post created by other user

After: Checking session ID along side with user_id in the request to make sure user can only view their own post or other people public post.

Secure web Application Architecture



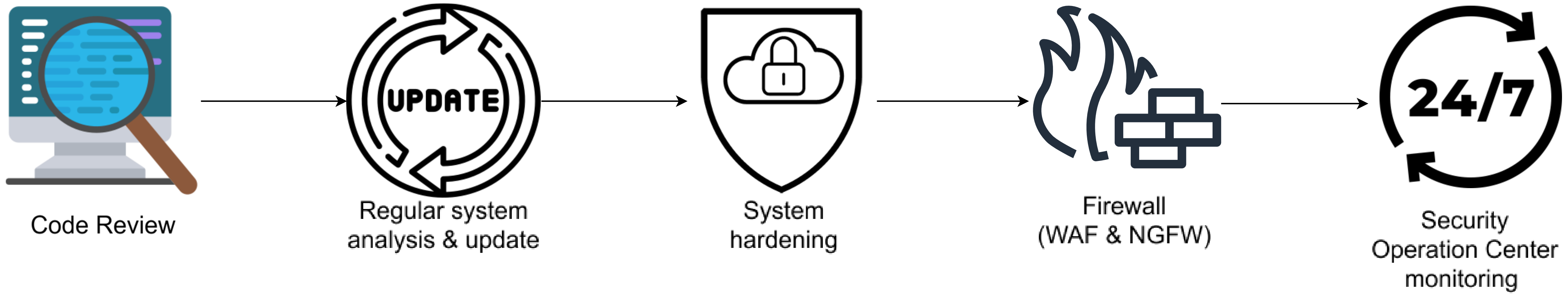
Secure web Application Architecture



IaC Solution

Application deploy using Ansible which streamline the deployment process and increase the consistency when a application is deploy on large number of systems.

Security solutions to defend web application with a reasonable cost


UPDATE
24/7
Code Review
Regular system analysis & update
System hardening
Firewall (WAF & NGFW)
Security Operation Center monitoring

Security solutions to defend web application with a reasonable cost

- There is no absolute security solution to protect against all the cybersecurity threats. There will always be vulnerabilities in any system so people is moving from vulnerabilities prevention to vulnerabilities resistance. So instead of trying to find and patch **ALL** the vulnerabilities, now its it about quickly patched up any newly discorvered vulnerabilities and finding way to reduce and mitigate risk to the system.
- Further more, security is also heavily dependant on criticality and budget to the application so **there is no one size fit all** security solution. Each application, company with there own characters require different thing in security.

Demo

Appendix: Example of Paid/Opensource solutions

Solutions	AWS native	Paid	Opensource
Monitoring	- AWS Control Tower - Cloudtrail - X-ray	- Splunk - IBM Qradar	- Wazuh - Security Onion - Utmstack
WAF	- AWS WAF	- Cloudflare WAF - Imperva cloud WAF	- Mod Security - Open-appsec
FW	- AWS Network Firewall - Security Group	- Palo Alto NGFW - Fortinet NGFW	- OPNsense - pfSense
KMS	- AWS KMS	- IBM Security Guardium - Doppler Secrets Management Platform	- Hashicorp Vault - Openstack KeyManager

Thank you for listening

Lê Quang Anh – Team 132

lequanganh97@gmail.com