МОНГОЛ УЛСЫН ИХ СУРГУУЛЬ
МЭДЭЭЛЛИЙН ТЕХНОЛОГИ,
ЭЛЕКТРОНИКИЙН СУРГУУЛЬ
Мэдээлэл, компьютерын ухааны тэнхим

МЭДЭЭЛЛИЙН АЮУЛГҮЙ БАЙДАЛ (ICSI448)

Төсөлт ажил Web Vulnerability Scanner

Гүйцэтгэсэн:
Б.Хэрлэн
21B1NUM0931

Шалгасан:
О.Билгүүн
МКУТ

Улаанбаатар хот
2024 он

1 Оршил

Техник технологи хөгжин, өргөжихийн хэрээр мэдээллийн аюулгүй байдалд заналхийлэл үргэлж нэмэгдэж байдаг. Асуудал улам даамжрах, хяналтаас гарахаас сэргийлэхийн тулд түүнийг түрүүлээд таньж арга хэмжээ авах шаардлага тулгарна. Өдөрт хүн дор хаяж нэг удаа заавал вебсайтад хандан ямарваа нэг үйлдэл хийж байгаа. Энэ нь хувийн мэдээллээ оруулах, гүйлгээ хийх гэх зэрэг хувь хүний мэдээлэл урсгалд орж байгаа юм. Харин түүнийг алдахгүйн тулд тухайн сүлжээнд ажиллаж байгаа вебсайт мэдээллийн аюулгүй байдлыг сайтар хангасан байх шаардлага тулгарна. Энэ асуудал нь хөгжүүлэгч хүмүүсийн хувьд чухал бөгөөд энэхүү бие даалтын сэдвээрээ вебийн өртөмтгий байдлыг шалгагчийг хийхээр зорив.

2 Зорилго

Веб сайтын өртөмтгий байдлыг нь шалгахдаа, ерөнхий тохиолддог *SQL injection attack, Cross site scripting(XSS), Command Injection, Directory Traversal, Open Redirect, Sensitive Data Exposure* гэх зэрэг гаднаас аюулд өртөж болзошгүй хүчин зүйлсийг шалгаж, түүнээс сэргийлэх сануулга өгдөг бүхий шалгагчийг хэрэгжүүлнэ.

3 Онолын судалгаа

3.1 Ерөнхий тохиолдлын халдлагууд

- SQL Injection (SQL Халдлага). Халдагч нь веб аппликейшн дээр байрлах өгөгдлүүдийг ашиглан өгөгдлийн бааз руу хуурамч SQL команд илгээж, тухайн өгөгдлийг өөрчлөх, хулгайлах эсвэл устгах боломжтой болдог. YYнийг хэрэгжүүлэхдээ ихэвчлэн аливаа оролтын хэсэгт 'admin – гэх зэрэг командуудыг бичиж өгч, өгөгдлийн баазад хүсэлт илгээснээр үйлддэг. SQL Injection-д өртөх нь аюултай бөгөөд тухайн өгөгдлийн баазын мэдээллийг устгах хүртэлх эрсдэлтэй.

- Reflected XSS. Энэ халдлагын үед халдагч нь вэб хуудаснаас хариу авахдаа хэрэглэгчийн хөтчид хуурамч JavaScript код илгээж, хуурамч вэбсайт эсвэл аюултай контент илгээдэг. Энэ нь хэрэглэгчийн мэдээг хулгайлахад ашиглагддаг.
  Stored XSS (Серверт хадгалагдсан XSS). Халдагч вэбсайт дээр хэрэглэгчийн өгөгдлийг илгээж, үүнийг сервер дээр хадгалан, дараа нь бусад хэрэглэгчдэд хуурамч код дамжуулдаг. Энэ нь олон хэрэглэгчдэд нөлөөлж, тэдний хөтчүүдэд аюултай код гүйцэтгэгдэх боломжтой.

- Command Injection. Вэб аппликейшн дээр хэрэглэгчийн оруулсан мэдээллийг ашиглан халдагч нь системийн командуудыг шууд гүйцэтгэх оролдлого хийдэг. Энэ нь үйлдлийн системийн командуудыг ашиглан серверийн хандалтад нөлөөлдөг.

- Open Redirect. Вэб хуудас нь хэрэглэгчийг өөр нэг хуудас руу шилжүүлэх үед халдагч нь хуурамч шилжүүлэг хийх замаар хэрэглэгчийг өөр вэбсайт руу чиглүүлдэг. Энэ нь хэрэглэгчдийг фишинг халдлагад өртүүлэхэд ашиглагддаг. Өртөх магадлал өндөртэй байдаг хамгийн түгээмэл тохиолддог халдлагуудын нэг юм.

- Directory Traversal. Энэ төрлийн халдлага нь хэрэглэгчийн хүсэлтээр серверийн файлууд руу хандаж, зөвшөөрөгдөөгүй файлуудыг унших буюу хуулбарлах оролдлоготой холбоотой. Энэ нь серверийн хуурамч файлуудыг ачаалалгүйгээр хадгалах, нууцлалыг алдах эрсдэлтэй. Энэ дундаа хэрэглэгчийн passwd зэргийг хадгалсан

фолдерт нэвтрэх, config файлд нэвтрэх зэргээр аюул учруулж болзошгүй байдаг.

- Secure Web. Вэб хуудас нь хэрэглэгчийн хувийн мэдээллийг (жишээ нь, нууц үг, кредит картын дугаар гэх мэт) хангалттай хамгаалалгүйгээр ил гаргаж, халдагчдын хулгайлах боломжийг олгодог. Таньж болох хамгийн энгийн арга нь http байна уу, https байна уу гэдгийг шалгах юм.

- Clickjacking. Энэ төрлийн халдлага нь вэб хуудсанд хуурамч интерфейс буюу орон зай нэмэх замаар хэрэглэгчийн мэдээг хулгайлах эсвэл вэбсайтад хүсээгүй үйлдлүүдийг гүйцэтгэх оролдлого юм. Энэ нь ихэвчлэн хэрэглэгчийн мэдээг хулгайлах зорилготой байдаг.

- CSRF (Cross-Site Request Forgery). Халдагч нь хэрэглэгчийн вэб сайт дээрх хуурамч хүсэлтүүдийг ачааллаж, түүний нэр дээр хүсээгүй үйлдлийг гүйцэтгэхийг оролддог. Энэ нь хэрэглэгчийн аюулгүй байдлыг алдагдуулж, нууц үгийг өөрчлөх, үйлдэл хийхэд ашиглагддаг.

HTTP Headers. HTTP нь хэрэглэгчийн хөтчөөс вэб сервер рүү илгээж буй мэдээллийн хэсэг юм. Халдагч нь HTTP толгойг өөрчлөн, аюултай код эсвэл залилангийн мэдээллийг оруулах замаар халдлага хийж болно. Эдгээр шаардлагатай HTTP толгой (headers)-г хэрэглэх нь вэб аюулгүй байдлыг хангахад маш чухал. Тэдгээрийн үүрэг дараах байдлаар тайлбарлаж болно:

- Strict-Transport-Security (HSTS). Энэ толгой нь вэб хуудас нь зөвхөн HTTPS холбоосоор нэвтэрч байхаар тодорхойлж, хэрэглэгчийг HTTP холболт хийхээс хамгаалдаг. Энэ нь MITM (Man-In-The-Middle) халдлагаас сэргийлэхэд тусалдаг, учир нь халдагчид HTTP холболтыг манипуляци хийх боломжгүй болгодог.

- X-Content-Type-Options. Энэ толгой нь вэб серверт зориулсан аюулгүй байдлын хамгаалалт юм. Энэ нь хөтчид зарим төрлийн аюултай контентыг автоматаар ачаалахгүй байхаар тохируулах зорилготой.

- X-Frame-Options. Энэ толгой нь сайтын аюулгүй байдалд хамгаалалт нэмэх бөгөөд вэб хуудсыг бусад сайтуудын "iframe"оруулахыг

хориглодог. Энэ нь clickjacking халдлагаас хамгаалах ач холбогдолтой, учир нь халдагчид бусад сайтад таны сайт байрлуулан хуурамч үйлдлийг гүйцэтгэж болно.

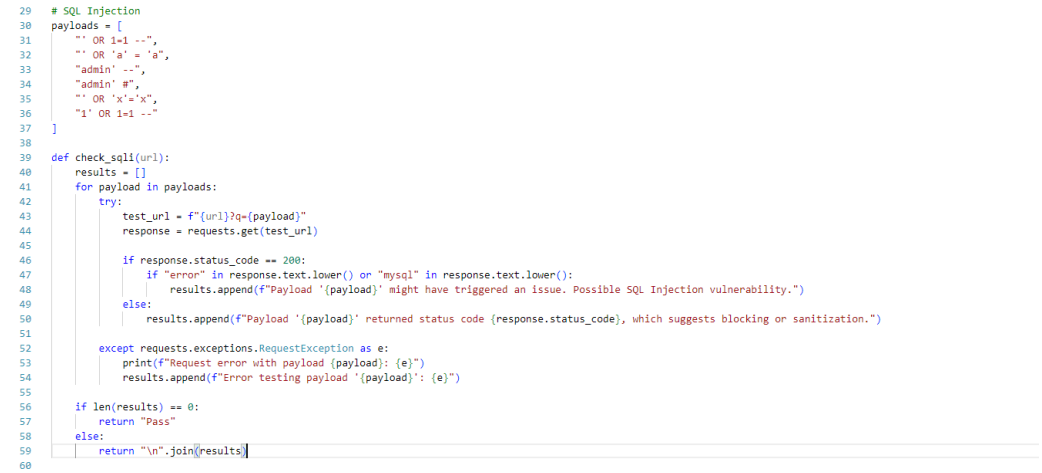
- Content-Security-Policy (CSP). CSP нь вэб хуудасны аюулгүй байдлыг нэмэгдүүлэх зорилготой. Энэ нь вэб хуудсаас malicious code (жишээ нь, JavaScript код - Command Injection ачаалахыг хязгаарлах боломжийг олгодог. CSP-ийг зөв тохируулах нь XSS (Cross-Site Scripting) халдлагаас сэргийлж, өртсөн тохиолдолд саармагжуулах боломжоор хангадаг.

- Permissions-Policy. Энэ толгой нь веб хуудас дээр хэрэглэгчийн тохиргоог хянах бөгөөд тодорхой үйлдлүүдийг (жишээ нь, камер, микрофон ашиглах) зөвхөн зөвшөөрөгдсөн сайтуудын оролцоотойгоор гүйцэтгэхийг шаарддаг.

- Referrer-Policy. Энэ толгой нь вэб хуудсанд өөр сайт руу шилжих үед "referer"мэдээллийг хэрхэн дамжуулахыг зохицуулдаг. Энэ нь хэрэглэгчийн хувийн мэдээллийг илрүүлэхгүй байхад тусалдгаас гадна хувийн мэдээлэл алдагдсан тохиолдолд дахин сэргээхэд мөн тус болдог.

4 Хэрэгжүүлэлт

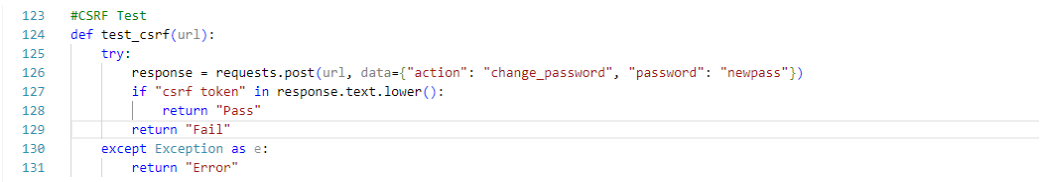
Хэрэгжүүлэлт хийхдээ, Nextjs фреймворк ашиглан frontend, python ашиглан backend хийж гүйцэтгэв.



```python
# SQL Injection
payloads = [
    "' OR 1=1 --",
    "' OR 'a' = 'a",
    "admin' --",
    "admin' #",
    "' OR 'x'='x",
    "1' OR 1=1 --"
]

def check_sqli(url):
    results = []
    for payload in payloads:
        try:
            test_url = f"{url}?q={payload}"
            response = requests.get(test_url)

            if response.status_code == 200:
                if "error" in response.text.lower() or "mysql" in response.text.lower():
                    results.append(f"Payload '{payload}' might have triggered an issue. Possible SQL Injection vulnerability.")
            else:
                results.append(f"Payload '{payload}' returned status code {response.status_code}, which suggests blocking or sanitization.")

        except requests.exceptions.RequestException as e:
            print(f"Request error with payload {payload}: {e}")
            results.append(f"Error testing payload '{payload}': {e}")

    if len(results) == 0:
        return "Pass"
    else:
        return "\n".join(results)

```

Зураг 1: SQL Injection

Тайлбар. SQL Injection-д өртөж болзошгүй эсэхийг шалгахдаа тухайн веб аппликейнд түгээмэл тохиолддог payload-г ашиглан хүсэлт илгээн шалгаж байна. Хэрвээ буцаж ирж буй response нь өгөгдлийн баазтай холбогдсон байгаа тохиолдолд өртөмтгий байж болзошгүй гэж үзнэ.



```python
#CSRF Test
def test_csrf(url):
    try:
        response = requests.post(url, data={"action": "change_password", "password": "newpass"})
        if "csrf token" in response.text.lower():
            return "Pass"
        return "Fail"
    except Exception as e:
        return "Error"
```

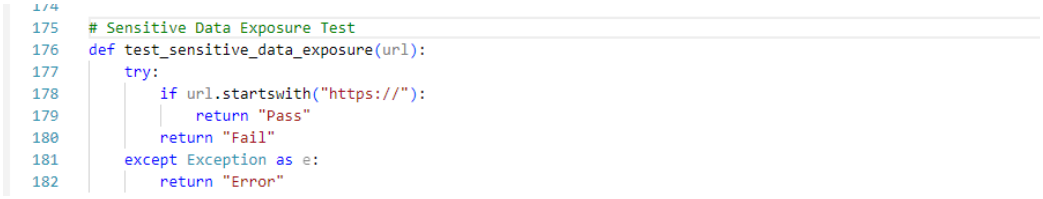
Зураг 2: CSRF

Тайлбар. URL-г ашиглан серверт хүсэлгүй үйлдэл болох жишээ нь, нууц үгийг өөрчлөх үйлдлийг POST ашиглан хийхээр оролдоход CSRF token байгаа эсэхийг хариунаас нь харж, шалгана.

```python
# Directory Traversal Test
def test_directory_traversal(url):
    try:
        payloads = [
        "../../../etc/passwd",
        "../../../../../../etc/passwd",
        "../../../../../../windows/win.ini",
        "..\\..\\..\\..\\..\\..\\..\\windows\\win.ini",
        "/etc/passwd",
        "/windows/win.ini",
        "..%2f..%2f..%2f..%2f..%2fetc%2fpasswd",
        "..%5c..%5c..%5c..%5c..%5cetc%5cpasswd",
        "..%252f..%252f..%252fetc%252fpasswd",
        "..%00../etc/passwd",
        "../../../../../../boot.ini",
        "..%2f..%2f..%2fwindows%2fwin.ini",
        "..%5c..%5c..%5cwindows%5cwin.ini",
        "/proc/self/environ"
        ]

        for payload in payloads:
            response = requests.get(url, params={"file": payload})

            # Check for sensitive content in the response
            if "root:" in response.text or "[extensions]" in response.text:
                return f"Fail: Vulnerable to directory traversal with payload '{payload}'"

        return "Pass: No directory traversal vulnerability detected"
    except Exception as e:
        return f"Error: {str(e)}"
```

Зураг 3: Directory Traversal

Тайлбар. Түгээмэл тохиолддог payload-г бүх боломжоор нь тухайн вебийн серверт хандах гэж үзнэ. Хэрвээ буцаан ирсэн response нь ямар нэгэн байдлаар sensitive мэдээллийг агуулсан байвал өртөж болзошгүй гэж үзнэ.



```python
# Sensitive Data Exposure Test
def test_sensitive_data_exposure(url):
    try:
        if url.startswith("https://"):
            return "Pass"
        return "Fail"
    except Exception as e:
        return "Error"
```

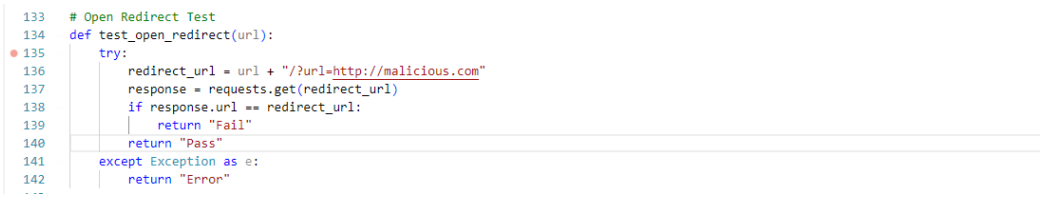
Зураг 4: Secure Web

Тайлбар. Энгийн тест болох тухайн өгөгдсөн URL өргөтгөл нь HTTPS протокол дээр явагдаж байгаа эсэхийг шалгана.

```python
#HTTPS Security headers
def test_http_headers(url):
    response = requests.get(url)
    headers = response.headers
    missing_headers = []
    required_headers = ['Strict-Transport-Security', 'X-Content-Type-Options', 'X-Frame-Options', 'Content-Security-Policy', 'Permissions-Policy', 'Referrer Policy']

    for header in required_headers:
        if header not in headers:
            missing_headers.append(header)

    if missing_headers:
        return f"Fail: Missing headers: {', '.join(missing_headers)}"
    return "Pass"
```

Зураг 5: HTTP Secure header

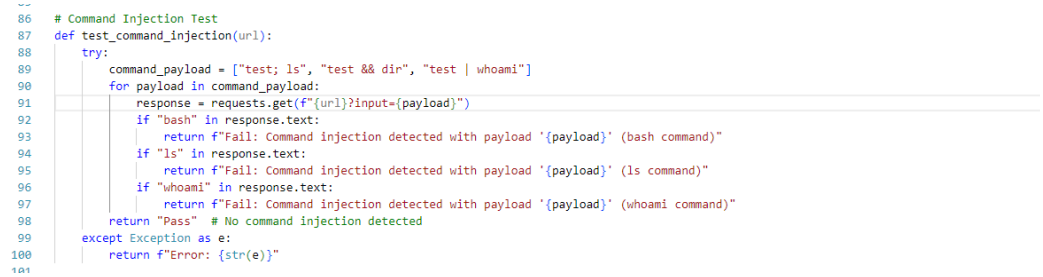
Тайлбар. Шаардлагатай HTTPS HEADERS агуулж байгаа эсэхийг шалгаад, дутуу байгаа толгой-нуудын нэрийг буцаана. Ингэснээрээ хөгжүүлэгч цаашид энэ тал дээр анхаарах боломж үүсгэж байна.



```python
# Open Redirect Test
def test_open_redirect(url):
    try:
        redirect_url = url + "/?url=http://malicious.com"
        response = requests.get(redirect_url)
        if response.url == redirect_url:
            return "Fail"
        return "Pass"
    except Exception as e:
        return "Error"
```

Зураг 6: Open Redirect

Тайлбар. URL-г ашиглан, ямар нэгэн хортой линк-лүү хандаж болж байгаа эсэхийг шалгана. Ямар нэгэн байдлаар болж байгаа тохиолдолд түүнийг өртөж болзошгүй гэж үзнэ.



```python
# Command Injection Test
def test_command_injection(url):
    try:
        command_payload = ["test; ls", "test && dir", "test | whoami"]
        for payload in command_payload:
            response = requests.get(f"{url}?input={payload}")
            if "bash" in response.text:
                return f"Fail: Command injection detected with payload '{payload}' (bash command)"
            if "ls" in response.text:
                return f"Fail: Command injection detected with payload '{payload}' (ls command)"
            if "whoami" in response.text:
                return f"Fail: Command injection detected with payload '{payload}' (whoami command)"
        return "Pass"  # No command injection detected
    except Exception as e:
        return f"Error: {str(e)}"
```

Зураг 7: Command Injection

Тайлбар. Түгээмэл ашиглагддаг, үйлдлийн системээс хамаарсан командуудыг оруулж үзснээр шалгана. Ямарваа нэгэн байдлаар, response-д sensitive мэдээлэл байвал түүнийг өртөж болзошгүй гэж үзнэ.

```python
@app.route('/scan', methods=['POST'])
def scan_website():
    data = request.json
    url = data.get('url')

    if not url:
        return jsonify({"error": "URL is required"}), 400

    results = {}
    results["URL"] = url
    results["SQL Injection"] = check_sqli(url)
    results["IP address"] = get_ip_address(url)
    results["Reflected XSS"] = test_reflected_xss(url)
    results["Stored XSS"] = test_stored_xss(url)
    results["Command Injection"] = test_command_injection(url)
    results["Open Redirect"] = test_open_redirect(url)
    results["Directory Traversal"] = test_directory_traversal(url)
    results["Sensitive Data Exposure"] = test_sensitive_data_exposure(url)
    results["Domain"] = get_domain_name(url)
    results["HTTP Headers"] = test_http_headers(url)
    results["Clickjacking"] = test_clickjacking(url)
    results["CSRF"] = test_csrf(url)
    return jsonify(results)

if __name__ == '__main__':
    app.run(debug=True, port=5000)
```

Зураг 8: Back-end ажилуулж буй

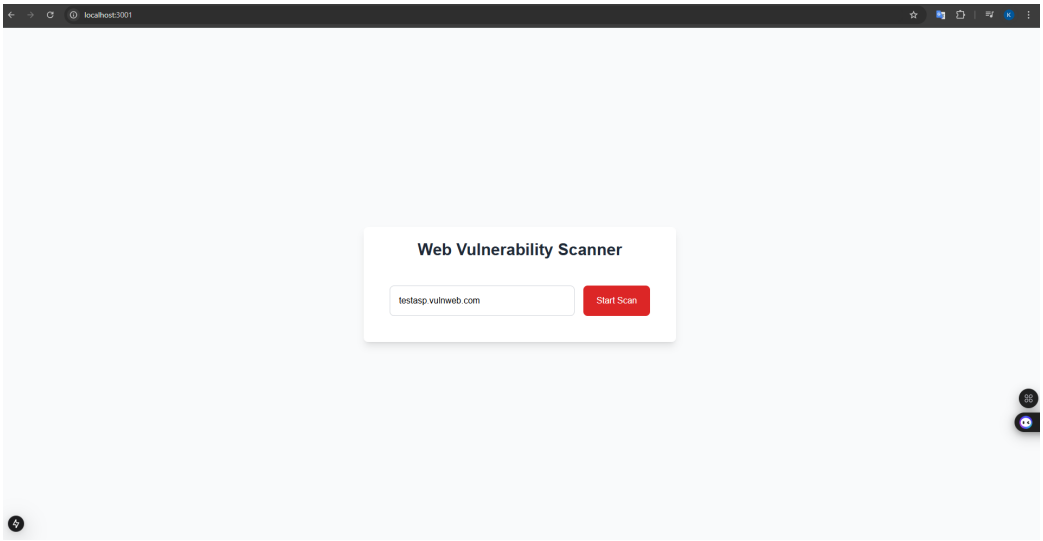
Тайлбар. Эдгээр ердийн тохиолддог аюулгүй байдлыг хэмжихэд чухал шаардлагатай хүчин зүйлүүдийг шалган, тэнцсэн тэнцээгүй мэдээллийг буцаадаг байдлаар backend ажиллана.

5 Ажиллах зарчим

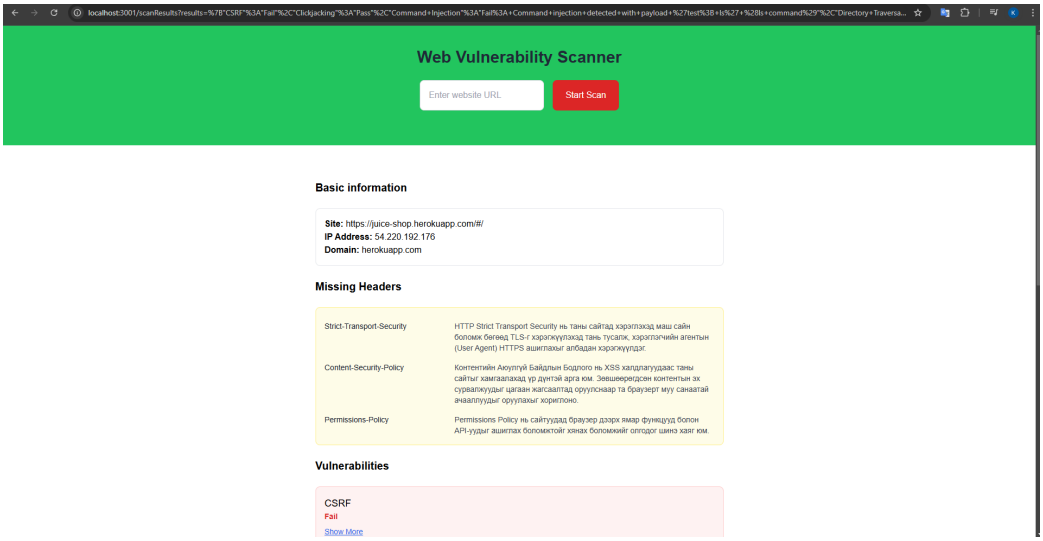
Ажиллах зарчмын хувьд,

- Шалгах веб аппликейшний линкийг оруулж, шалгана. Ингэхдээ бусад сайтад энэ мэтээр халдах нь хууль бус тул тусгай зөвшөөрөлтэйгээр шалгалт хийх шаардлагатай. Уш бие даалтын хүрээнд бэлэн, өртөмтгий байхаар бүтээгдсэн туршилтын сайтуудыг туршсан болно.

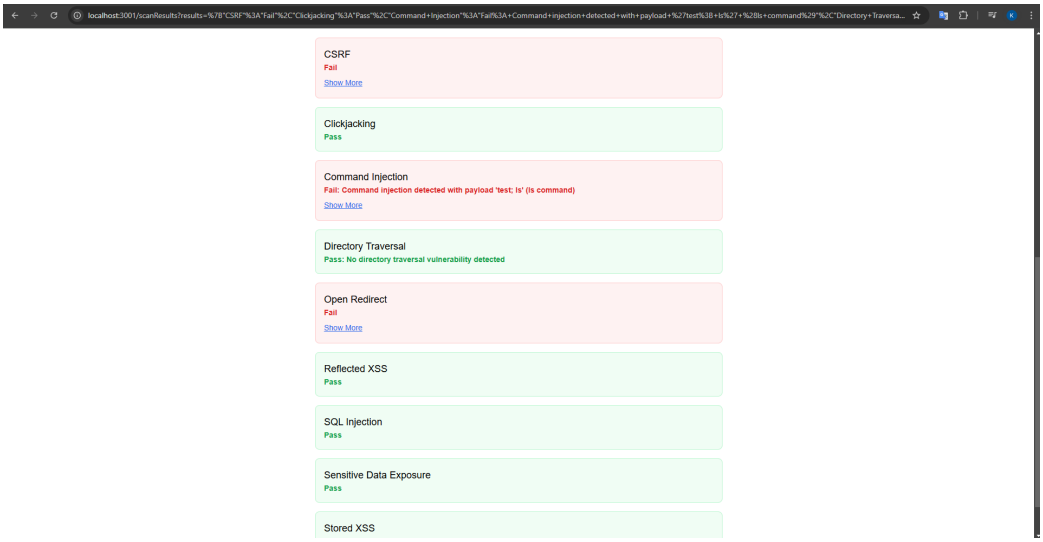
- Үр дүнг харуулахдаа дээрх хэсэг нь ногоон өнгөтэй байвал харьцангуй secure байгаа боловч дутагдалтай зүйлүүд байгааг харж болно. Байхгүй байгаа HTTP HEADER-ын мэдээлэлийг харуулна. Үүний дараагаар, аль тестийг давсан болон даваагүй байгаа аюулгүй байдлын хүчин зүйлсийн тайлбар болон хэрхэн сэргийлж болохыг харуулав.



Зураг 9: Ажиллуулж харуулсан - 1



Зураг 10: Ажиллуулж харуулсан - 2



Зураг 11: Ажиллуулж харуулсан - 3

6 Дүгнэлт

Уг төсөлт ажлын хүрээнд, хамгийн энгийн байдлаар хүчин зүйлсийг нь шалгаж, хэрэгжүүлэлт хийсэн бөгөөд жижиг хэмжээний web vulnerability scanner бүтээв. Өдөр тутам, шинэ содон халдлага үүсэж, хийгдэж байгаа энэ үед үүнийг илүү өргөжүүлж ашиглах нь чухал юм. Энэхүү ажлыг гүйцэтгэснээрээ, веб аппликейшнийг хэрхэн илүү найдвартай болгох талаар илүү их төсөөллийг олж, цаашид анхаарах зүйлсээ мэдэж авлаа. Энэхүү бүтээл нь зөвхөн өөрийн хэрэглээнд зохион бүтээгдэв. Энгийн аюулгүй байдлын хэмжүүрүүдийг хангаж байгаа эсэхийг шалгаж болохуйц scanner бүтээн ажиллав.