📄 INCIDENT REPORT — Plaintext Credential Exposure Over HTTP

Prepared By: ***Nobal Antony.M***

Date: ***3.12.2025***

Classification: Security Awareness / Network Analysis Lab

Environment: Controlled Testing Environment (Non-Production)

1. Executive Summary

During a controlled cybersecurity analysis exercise, unencrypted login credentials were successfully intercepted using **Wireshark** from two publicly available, intentionally insecure testing websites:

1. **testphp.vulnweb.com** (Designed for security research)

2. http://httpbin.org/forms/post (Safe HTTP POST testing endpoint)

The objective was to validate the risks associated with transmitting sensitive data over **unencrypted HTTP channels** and to observe how attackers may capture credentials during transit.

This lab demonstrated how **HTTP POST requests carry sensitive information in clear text**, making them vulnerable to interception by any party on the same network or through a man-in-the-middle (MITM) scenario.

This report outlines the findings, methodology, technical evidence, and recommendations.

2. Scope of Analysis

2.1 Objective

- To capture and analyze HTTP POST requests containing user credentials using Wireshark.
- To demonstrate the vulnerability of HTTP-based authentication.
- To simulate credential interception in a legal, controlled lab environment.

2.2 In-Scope Assets

- `http://testphp.vulnweb.com/login.php`
- `http://httpbin.org/forms/post` → `http://httpbin.org/post`

2.3 Tools Used

- **Wireshark 4.x** (Packet Capture and Network Protocol Analysis)
- **Web Browser (Chrome/Firefox)**
- **Local Wi-Fi Network Interface**

3. Methodology

3.1 Packet Capture Setup

- Wireshark was launched on the active network interface (Wi-Fi).
- Capture was started before interacting with target websites.
- Display filters were applied to isolate relevant traffic.

3.2 Interaction with Target Websites

A) testphp.vulnweb.com (Login Page)

- A test login form was accessed.
- Fake credentials (e.g., `username=test`, `password=password123`) were submitted.
- Traffic was captured in real time.

B) http://httpbin.org/forms/post

- The form page was opened.
- Multiple test inputs (simulated usernames, passwords, and fields) were sent.
- This endpoint returned full form data in the response body, aiding visibility.

3.3 Wireshark Filters Used

To isolate HTTP login traffic:

```
http.request.method == "POST"
```

For searching sensitive fields inside packets:

```
frame contains "username"
frame contains "password"
```

To reconstruct and inspect the raw payload:

- **Follow → HTTP Stream**

3.4 Credential Spoofing & Observation

- Multiple fake credentials were submitted intentionally.
- Each submission generated visible HTTP POST payloads in clear text.
- These were captured and reconstructed using Wireshark.

4. Technical Findings & Evidence

4.1 Observed HTTP POST Packet (testphp.vulnweb.com)

Captured Request:

```
POST /userinfo.php HTTP/1.1
Host: testphp.vulnweb.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 45

uname=test&pass=password123&submit=Submit
```

4.2 Observed HTTP POST Packet (httpbin.org)

Captured Request:

```
POST /post HTTP/1.1
Host: httpbin.org
Content-Type: application/x-www-form-urlencoded

custname=admin&custtel=spoofedPass123&custemail=test@
test.com
```

4.3 HTTP Stream Reconstruction

Using **Follow → HTTP Stream**, full request/response sequences were recovered, showing:

- User-supplied username
- User-supplied password
- Form metadata (fields, names, content-type)

All appeared **in clear text** due to lack of encryption.

4.4 No Encryption Detected

The sessions used:

- **HTTP/1.1 (unencrypted)**
- **No TLS handshake**
- **No certificate exchange**

This confirms complete visibility of credentials in transit.

5. Impact Assessment

If this vulnerability existed in a production or corporate environment, the risk would be **critical**.

5.1 Potential Attack Scenarios

- **Eavesdropping:** Any attacker on the same network captures credentials.
- **MITM Attacks:** Public Wi-Fi hotspots become highly dangerous.
- **Session Hijacking:** Cookies, tokens, or session data may be exposed.
- **Credential Stuffing Risk:** Users reusing passwords across platforms increase the threat.

5.2 Severity

High (in a real-world environment)
The lack of encryption directly exposes sensitive data.

6. Recommendations

6.1 Immediate

- Enforce **HTTPS/TLS 1.2 or 1.3** for all login endpoints.
- Redirect all HTTP requests to HTTPS via 301 redirect.
- Configure HSTS (HTTP Strict Transport Security).

6.2 Short-Term

- Implement secure coding practices for handling credentials.

- Ensure form submissions use encrypted channels.
- Conduct regular network-level security audits.

6.3 Long-Term

- Adopt tokenized authentication (OAuth2, JWT).
- Use encrypted session cookies with secure + HttpOnly flags.
- Educate development teams on secure transport protocols.

7. Conclusion

This controlled analysis confirmed that:

- **HTTP transmits credentials in clear text**, making them easily interceptable.
- **Wireshark can capture and reconstruct sensitive data** during transit with minimal effort.
- **TLS encryption is mandatory** for protecting authentication and user data.

The insights from this exercise reinforce the importance of performing **regular security assessments**, understanding **network-level vulnerabilities**, and implementing **industry-standard encryption protocols**.