COMPANY NAME: REDACTED

Logo Redacted

XYZ Portal

Penetration Testing Report

Prepared By

Muhammad Taha

Cyber Security Team

Prepared On
1/29/2019

NOTE: This document contains information which is intended for sole and Private use of XYZ. By reading this document you agree to keep the contents in confidence and do not copy, disclose, or distribute this outside XYZ

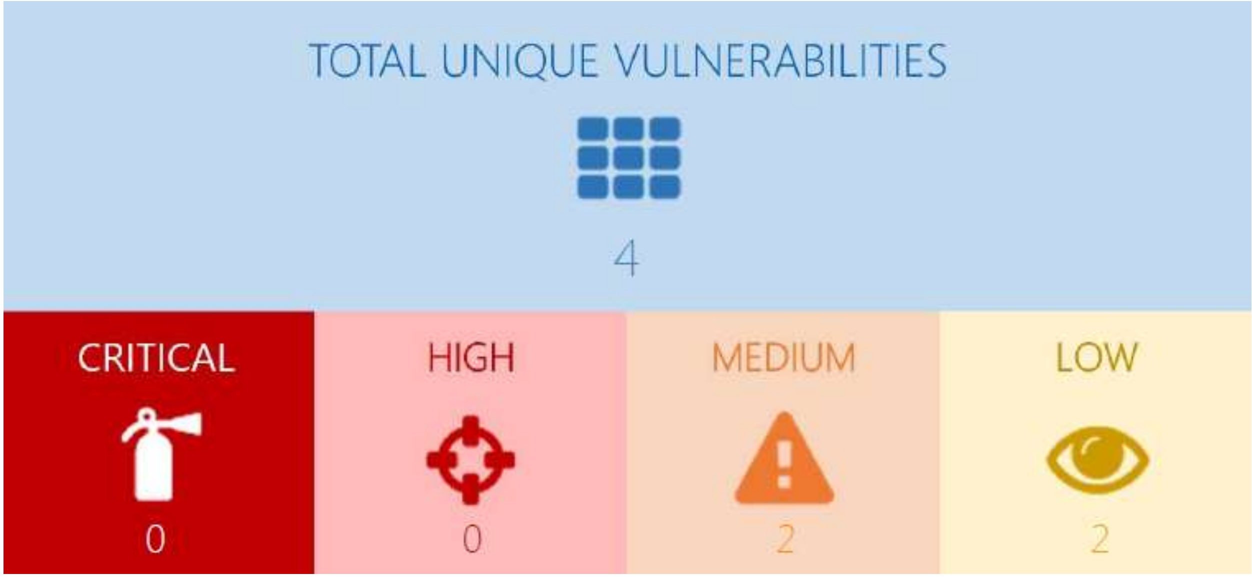
Contents

1. Executive Summary

This report is the result review of during the penetration test, the security assessment of “**XYZ Portal**” Application. As a result of the penetration testing exercise it was possible to confirm that the application does not implement necessary security controls to prevent issues such as Transport layer security protection and x-frame headers to protect users.

The results of Penetration Testing show that there are several security loopholes and weaknesses which can be exploited. These weaknesses require immediate attention and should be given high priority for solution implementation. Detail of vulnerability scanning of systems and network devices, is also included in this report corresponding to “High” ,“Medium” and “Low” level vulnerabilities those require immediate actions.

1.1 Security Testing Overview

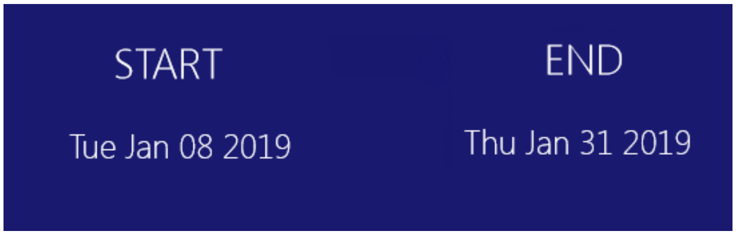


TOTAL UNIQUE VULNERABILITIES

4

CRITICAL	HIGH	MEDIUM	LOW
0	0	2	2

2. Testing Summary



START	END
Tue Jan 08 2019	Thu Jan 31 2019

3. Vulnerabilities

Critical

No Critical Vulnerabilities

High

No High Vulnerabilities

Medium

E1- Clickjacking Attack

E2- Insecure Transport Layer Protection

Low

E3- Anonymous Cipher Suite

E4- Insufficient Framing Protection

E1. Clickjacking Attack

Affected hosts

- ➔ https://xyz.com

Description

In a clickjacking attack the victim is tricked into unknowingly initiating some action in one system while interacting with the UI from seemingly completely different system. While being logged in to some target system, the victim visits the attackers' malicious site which displays a UI that the victim wishes to interact with. In reality, the click-jacked page has a transparent layer above the visible UI with action controls that the attacker wishes the victim to execute. The victim clicks on buttons or other UI elements they see on the page which actually triggers the action controls in the transparent overlaying layer. Depending on what that action control is, the attacker may have just tricked the victim into executing some potentially privileged (and most certainly undesired) functionality in the target system to which the victim is authenticated. The basic problem here is that there is a dichotomy between what the victim thinks he's clicking on versus what he or she is actually clicking on.

Attack Scenario

A victim has an authenticated session with a site https://xyz.com. At the same time, the victim receives an e-mail that appears to come from an online publication to which he or she subscribes with links to today's news articles. The victim clicks on one of these links and is taken to a page with the news story. There is a screen with an advertisement that appears on top of the news article with the 'skip this ad' button. Eager to read the news article, the user clicks on this button. Nothing happens. The user clicks on the button one more time and still nothing happens. In reality, the victim activated a hidden action control located in a transparent layer above the 'skip this ad' button. The ad screen blocking the news article made it likely that the victim would click on the 'skip this ad' button. Clicking on the button can actually give Hackers access to user’s sensitive information.

Recommendation

HTTP security headers provide a layer of security by helping to mitigate attacks and security vulnerabilities by telling the browser how to behave. X-frame-options (XFO), which is a header that **helps to protect your visitors against clickjacking attacks**. It is recommended that you use the x-frame-options header on pages which should not be allowed to render a page in a frame.

Steps to Reproduce

1- Create a fake website and hide website address in iframe tag.

Code

```
<html>
   <head> Clickjack test page</head>
   <body>
   <p>Click here to get offer</p>
   <iframe src="http://xyz.com" width="500" height="500"></iframe>
   </body>
</html>
```

2- Wait for users to visit this fake site and click on the hidden iframe.

3- Attacker can perform various functions including transferring sensitive information to his server etc.

E2. Insecure Transport Layer Protection

PRIORITY

Medium

Affected hosts

➔ https://xyz.com

Description

The backend web service is accessed over networks which are not controlled by end users or application developers. As a result, the data transmitted over these networks is liable to be intercepted in transit if it is not appropriately encrypted. The Hypertext Transfer Protocol (HTTP) used does not provide any mechanisms for ensuring confidentiality of data passed. If network traffic is able to be sniffed, then communications using this protocol can be obtained. Authentication and transfer of application data was found to occur over HTTP. This may allow disclosure of credentials or other sensitive information, to other parties who are able to monitor this traffic.

Attack Scenario

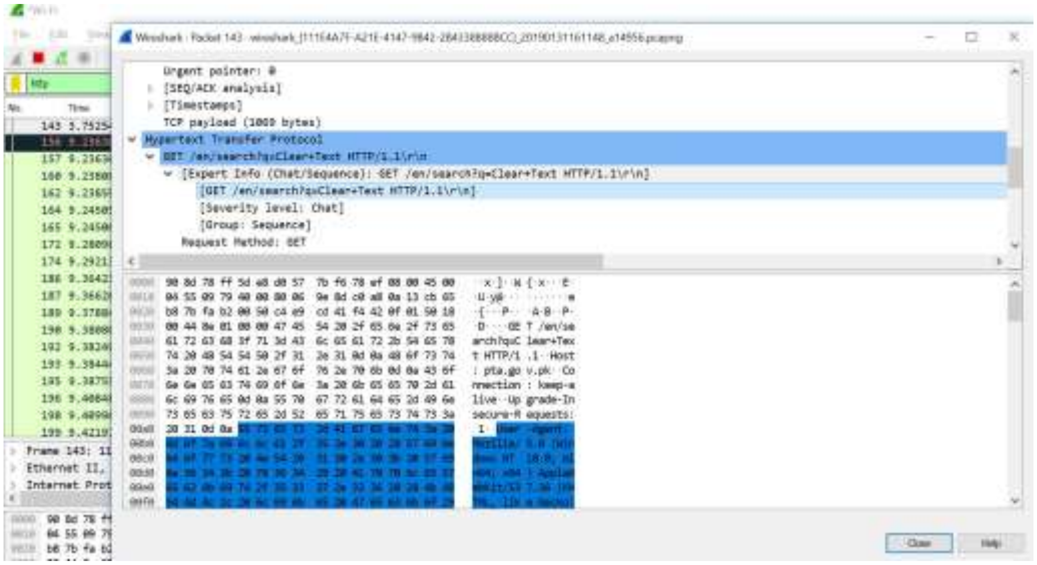
As the application has not implemented TLS properly, an attacker who is appropriately positioned, can simply monitor network traffic. This is most likely to occur when communicating over an open wireless connection or on an uncontrolled network. The attacker can capture credentials, or session tokens, and obtain access to the application data that is passed.

Recommendation

It is recommended to implement transport layer security on website and site through the use of TLS with the addition of HSTS Header. The implementation of this will ensure confidentiality of communications are adequately protected and will redirect all http requests to https.

Steps to Reproduce

1- Change Https to Http manually
2- Send the link to users who will visit sensitive information on the website.
3- Capture the network packets using Wireshark or other packet sniffing tool.



Wireshark · Packet 143 · wireshark_{11164A7F-A21E-4147-9842-2B4338888CC}_20190131161148_a14556.pcapng

```
    Urgent pointer: 0
  > [SEQ/ACK analysis]
  > [Timestamps]
    TCP payload (1000 bytes)
v Hypertext Transfer Protocol
  v GET /en/search?q=Clear+Text HTTP/1.1\r\n
    v [Expert Info (Chat/Sequence): GET /en/search?q=Clear+Text HTTP/1.1\r\n]
        [GET /en/search?q=Clear+Text HTTP/1.1\r\n]
        [Severity level: Chat]
        [Group: Sequence]
      Request Method: GET
```

```
0000  98 8d 78 ff 5d a8 d8 57  7b f6 78 ef 08 00 45 00   ·x·]··W{·x···E·
0010  04 55 09 79 40 00 80 06  9e 8d c0 a8 0a 13 cb 65   ·U·y@··········e
0020  b8 7b fa b2 00 50 c4 e9  cd 41 f4 42 0f 01 50 18   ·{···P···A·B··P·
0030  00 44 0a 81 00 00 47 45  54 20 2f 65 6e 2f 73 65   ·D····GET /en/se
0040  61 72 63 68 3f 71 3d 43  6c 65 61 72 2b 54 65 78   arch?q=C lear+Tex
0050  74 20 48 54 54 50 2f 31  2e 31 0d 0a 48 6f 73 74   t HTTP/1 .1··Host
0060  3a 20 70 74 61 2e 67 6f  76 2e 70 6b 0d 0a 43 6f   : pta.go v.pk··Co
0070  6e 6e 65 63 74 69 6f 6e  3a 20 6b 65 65 70 2d 61   nnection : keep-a
0080  6c 69 76 65 0d 0a 55 70  67 72 61 64 65 2d 49 6e   live··Up grade-In
0090  73 65 63 75 72 65 2d 52  65 71 75 65 73 74 73 3a   secure-R equests:
00a0  20 31 0d 0a [illegible]                              1··User-Agent:
```

E3. Anonymous Cipher Suite

PRIORITY

Low

Affected hosts

➔ https://xyz.com

Description

The default SSL cipher configuration in Apache Tomcat 4.1.28 through 4.1.31, 5.0.0 through 5.0.30, and 5.5.0 throughc5.5.17 uses certain insecure ciphers, including the anonymous cipher, which allows remote attackers to obtain sensitive information or have other, unspecified impacts.

Attack Scenario

The Secure Socket Layer (SSL) protocol allows for secure communication between a client and a server. The client usually authenticates the server using an algorithm like RSA or DSS. Some SSL ciphers allow SSL communication without authentication. Most common Web browsers like Microsoft Internet Explorer, Netscape and Mozilla do not use anonymous authentication ciphers by default. A vulnerability exists in SSL communications when clients are allowed to connect using no authentication algorithm. SSL client-server communication may use several different types of authentication: RSA, Diffie-Hellman, DSS or none. When 'none' is used, the communications are vulnerable to a man-in-the-middle attack.

Recommendation

Remove anonymous ciphers from default configurations if not changed manually before.

E4. Insufficient Framing Protection

Affected hosts

PRIORITY

Low

- ➔ https://xyz.com

Description

The act of placing one web page inside another web page is referred to as framing. This is performed through the use of HTML tag called an iframe. An example where framing is often performed is for the inclusion of content from other sources such as embedding of maps or videos. Should a trusted application be framed within an untrusted web page, click-jacking or frame sniffing attacks against application user can be performed. The application was found to be lacking appropriate X-Frame-Options headers which significantly reduces likelihood against click-jacking and frame sniffing attacks.

Attack Scenario

An attacker may be able to get an authenticated user to inadvertently perform actions by luring or enticing them to access a maliciously crafted web page. By clicking on the page, actions could be performed on the user’s behalf, through the framed page. There are few examples available on the Internet of how to build a clickjacking web page, however using such examples in a real life attack would most likely require significant customization in order to take advantage of the particular functions within the Web front end for the application.

Recommendation

To prevent framing attacks, include the X-Frame-Options HTTP header for all web application page responses.

The specified values available for X-Frame-Options are:

- **DENY:** The page cannot be displayed in a frame

- **SAMEORIGIN:** The page can only be displayed in a frame on the same origin as the page itself.

- **ALLOW-FROM URI:** The page can only be displayed in a frame on the specified origin

Typically, the **DENY** option is suitable for most web applications.