

# [REDACTED] PEN TEST

PREPARED FOR

[REDACTED]

*August 4th, 2023*

Contents

Contents \_\_\_\_\_ 2

Executive Summary \_\_\_\_\_ 3

Detailed Finding Breakdown\_\_\_\_\_ 4

OWASP Category Descriptions \_\_\_\_\_ 10

## Executive Summary

### Background

In the summer of 2023, Select Medical's Vulnerability Management team performed a penetration test of the [redacted] website.

This test was designed to help [redacted] identify security gaps in their web application that could be exploited by threat actors.

Testing occurred on 8/2/23. This report contains the full results of the test, including detailed observations, OWASP Category, and recommendations for remediations.

OBSERVATION COUNT	
0	Critical Risk Observations
0	High Risk Observations
10	Medium Risk Observations
4	Low Risk Observations
14	Total Observations

TIMELINE	
07/31/2023	Kickoff
08/02/2023	Vulnerability Scan
08/04/2023	Report Delivery

### SCOPE

Testing occurred on [redacted] website and was conducted from Qualys and Burp Suite

[https://\[redacted\].com/](https://[redacted].com/)

### FINDING

ID	TITLE	RISK	NUMBER	PROPORTION
OB-1	Clickjacking	Medium	10	71.5%
OB-2	Cookie "HTTPOnly" Attribute	Low	2	14.25%
OB-3	Cookie "Secure" Attribute	Low	2	14.25%

## Detailed Finding Breakdown

### OB1 – Clickjacking

OWASP CATEGORY	SEVERITY	NUMBER
A5 Security Misconfiguration	Medium	10

#### THREAT

The web page can be framed. This means that clickjacking attacks against users are possible.

#### IMPACT

With clickjacking, an attacker can trick a victim user into clicking an invisible frame on the web page, thereby causing the victim to take an action they did not intend to take.

#### SOLUTION

Solution Clickjacking prevention mechanisms include:

- X-Frame-Options: This HTTP response header can be used to prevent framing of web pages.
- Content-Security-Policy: The 'frame-ancestors' directive can be used to prevent framing of web pages.
- Framekiller JavaScript code designed to prevent a malicious user from framing the page. This method is not recommended due to its unreliability.

#### OCCURANCES

- [redacted]
- [redacted]
- [redacted]
- [redacted]
- [redacted]
- [redacted]
- [redacted]
- [redacted]
- [redacted]
- [redacted]

# [REDACTED] PEN TEST

## ADDITIONAL RESOURCES

150124 Clickjacking - Framable Page			
URL: https://			
Finding #	2	*(S )	Web Application
Unique #	c0		Authentication Not Used
Patch #	-		
Group	Information Disclosure	Detection Date	02 Aug 2023
CWE	CWE-451	CVSS V3 Attack Vector	NETWORK
OWASP	A5 Security Misconfiguration		
WASC	WASC-15 APPLICATION MISCONFIGURATION		
CVSS V3 Base	5.8	CVSS V3 Temporal	5.5

Figure 1 – Qualys clickjacking findings ([redacted] page)

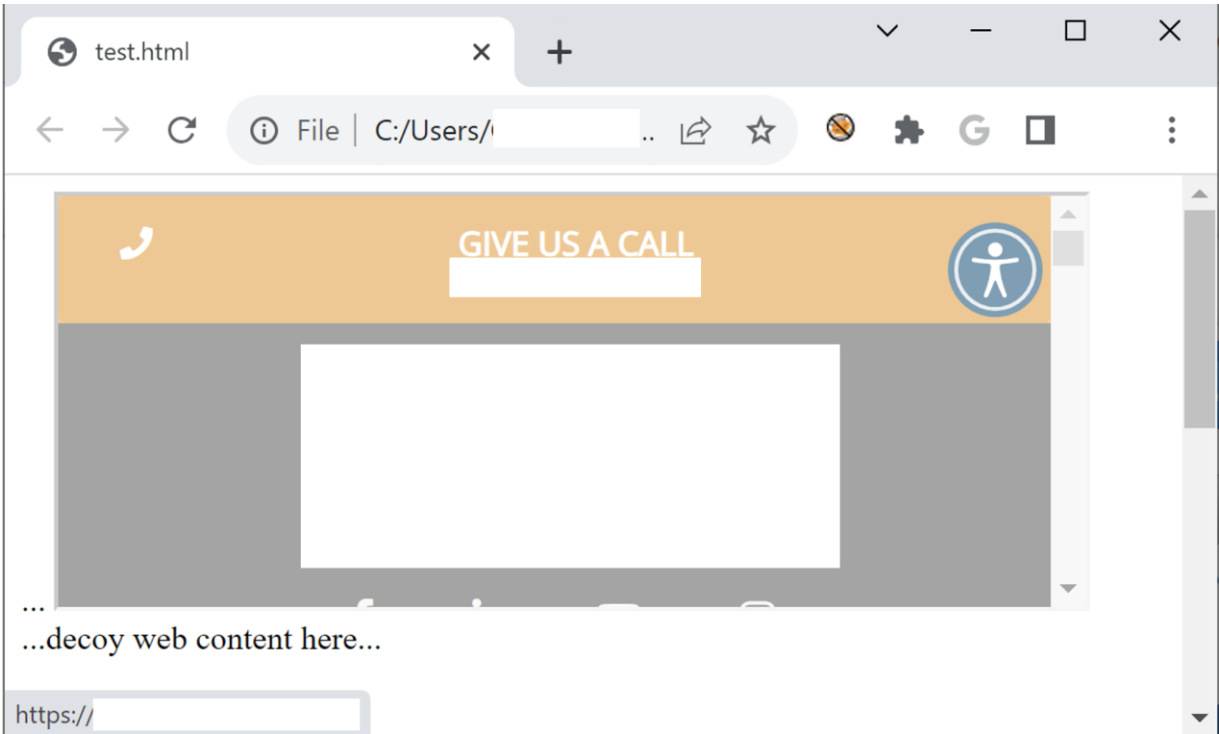


Figure 2 – Clickjacking example ([redacted] page)

# [REDACTED] PEN TEST

## OB2 – Cookie “HTTPOnly” Attribute

OWASP CATEGORY	SEVERITY	NUMBER
A5 Security Misconfiguration	Low	2

### THREAT

Threat The cookie does not contain the "HTTPOnly" attribute.

### IMPACT

Cookies without the "HTTPOnly" attribute are permitted to be accessed via JavaScript. Cross-site scripting attacks can steal cookies which could lead to user impersonation or compromise of the application account.

### SOLUTION

If the associated risk of a compromised account is high, apply the "HTTPOnly" attribute to cookies.

### OCCURANCES

- [redacted] (Cookie name: [redacted])
- [redacted] (Cookie name: [redacted])

# [REDACTED] PEN TEST

## ADDITIONAL RESOURCES

150123 Cookie Does Not Contain The "HTTPOnly" Attribute			
URL: https://			
Finding #	2	* (5)	Web Application
Unique #	69		Authentication Not Used
Patch #	–		
Group	Information Disclosure	Detection Date	02 Aug 2023
CWE	CWE-1004	CVSS V3 Attack Vector	NETWORK
OWASP	A5 Security Misconfiguration		
WASC	WASC-4 INSUFFICIENT TRANSPORT LAYER PROTECTION		
CVSS V3 Base	4.3	CVSS V3 Temporal	4.1

Figure 3 – Qualys cookie “HTTPOnly” findings (home page)

#1 Request

GET https://  
Host: .com  
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_14\_5) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.1.1 Safari/605.1.15  
Accept: \*/\*

Click this [link](#) to try to reproduce the vulnerability using above payload. Note that clicking this link may not lead to visible results, either because the vulnerability requires context to be previously set (authentication, cookies...) or because the exploitation of the vulnerability does not lead to any visible proof.

#1 Response

=deleted; expires= ; path=/; domain= : max-age=  
Cookies set via JavaScript do not have an associated HTTP response header.


 Export...

Figure 4 – Qualys cookie “HTTPOnly” payload (home page)

# [REDACTED] PEN TEST

## OB2 – Cookie “Secure” Attribute

OWASP CATEGORY	SEVERITY	NUMBER
A5 Security Misconfiguration	Low	2

### THREAT

Threat The cookie does not contain the "secure" attribute.

### IMPACT

Cookies with the "secure" attribute are only permitted to be sent via HTTPS. Cookies sent via HTTP expose an unsuspecting user to sniffing attacks that could lead to user impersonation or compromise of the application account.

### SOLUTION

If the associated risk of a compromised account is high, apply the "secure" attribute to cookies and force all sensitive requests to be sent via HTTPS.

### OCCURANCES

- [redacted] (Cookie name: [redacted])
- [redacted] (Cookie name: [redacted])



# [REDACTED] PEN TEST

150122 Cookie Does Not Contain The "secure" Attribute			
URL: https://			
Finding #	2	* (9 )	Web Application
Unique #	2c		Authentication Not Used
Patch #	—		
Group	Information Disclosure	Detection Date	02 Aug 2023
CWE	CWE-614	CVSS V3 Attack Vector	NETWORK
OWASP	A5 Security Misconfiguration		
WASC	WASC-4 INSUFFICIENT TRANSPORT LAYER PROTECTION		
CVSS V3 Base	4.3	CVSS V3 Temporal	4.1

Figure 5 – Qualys cookie “secure” findings (home page)

#1 Request

GET https://  
Host:  
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10\_14\_5) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/12.1.1 Safari/605.1.15  
Accept: \*/\*

Click this [link](#) to try to reproduce the vulnerability using above payload. Note that clicking this link may not lead to visible results, either because the vulnerability requires context to be previously set (authentication, cookies...) or because the exploitation of the vulnerability does not lead to any visible proof.

#1 Response

=deleted; expires= ; path=/; domain= ; max-age=  
Cookies set via JavaScript do not have an associated HTTP response header.

Export...

Figure 6 – Qualys cookie “secure” payload (home page)

## OWASP Category Descriptions

The following table summarizes the OWASP categories and descriptions:

OWASP CATEGORY	DESCRIPTION
<b>Broken Access Control</b>	A01:2021 - Access control enforces policy such that users cannot act outside of their intended permissions. Failures typically lead to unauthorized information disclosure, modification, or destruction of all data or performing a business function outside the user's limits.
<b>Cryptographic Failures</b>	A02:2021 - The first thing is to determine the protection needs of data in transit and at rest. For example, passwords, credit card numbers, health records, personal information, and business secrets require extra protection, mainly if that data falls under privacy laws, e.g., EU's General Data Protection Regulation (GDPR), or regulations, e.g., financial data protection such as PCI Data Security Standard (PCI DSS).
<b>Injection</b>	A03:2021 - Injection flaws, such as SQL, NoSQL, OS, and LDAP injection, occur when untrusted data is sent to an interpreter as part of a command or query. The attacker's hostile data can trick the interpreter into executing unintended commands or accessing data without proper authorization.
<b>Insecure Design</b>	A04:2021 - Insecure design is a broad category representing different weaknesses, expressed as "missing or ineffective control design." Insecure design is not the source for all other Top 10 risk categories. There is a difference between insecure design and insecure implementation. We differentiate between design flaws and implementation defects for a reason, they have different root causes and remediation. A secure design can still have implementation defects leading to vulnerabilities that may be exploited. An insecure design cannot be fixed by a perfect implementation as by definition, needed security controls were never created to defend against specific attacks.
<b>Security Misconfiguration</b>	A05:2021 - Commonly a result of insecure default configurations, incomplete or ad hoc configurations, open cloud storage, misconfigured HTTP headers, and verbose error messages containing sensitive information.
<b>Vulnerable and Outdated Components</b>	A06:2021 - Components, such as libraries, frameworks, and other software modules, run with the same privileges as the application. If a vulnerable component is exploited, such an attack can facilitate serious data loss or server takeover. Applications and APIs using components with known vulnerabilities may undermine application defenses and enable various attacks and impacts.
<b>Identification and Authentication Failures</b>	A07:2021 - Application functions related to authentication and session management are often implemented incorrectly, allowing attackers to compromise passwords, keys, or session tokens, or to exploit other implementation flaws to assume other users' identities temporarily or permanently.
<b>Software and Data Integrity Failures</b>	A08:2021 - Software and data integrity failures relate to code and infrastructure that does not protect against integrity violations. An example of this is where an application relies upon plugins, libraries, or modules from untrusted sources, repositories, and content delivery networks (CDNs). An insecure CI/CD pipeline can introduce the potential for unauthorized access, malicious code, or system

	compromise. Lastly, many applications now include auto-update functionality, where updates are downloaded without sufficient integrity verification and applied to the previously trusted application.
<b>Security Logging and Monitoring Failures</b>	A09:2021 - Insufficient logging and monitoring, coupled with missing or ineffective integration with incident response, allows attackers to further attack systems, maintain persistence, pivot to more systems, and tamper, extract, or destroy data.
<b>Server-Side Request Forgery (SSRF)</b>	A10:2021 - SSRF flaws occur whenever a web application is fetching a remote resource without validating the user-supplied URL. It allows an attacker to coerce the application to send a crafted request to an unexpected destination, even when protected by a firewall, VPN, or another type of network access control list (ACL). As modern web applications provide end-users with convenient features, fetching a URL becomes a common scenario. As a result, the incidence of SSRF is increasing. Also, the severity of SSRF is becoming higher due to cloud services and the complexity of architectures.