CLIENT-NAME WEB APPLICATION AND INFRASTRUCTURE VULNERABILITY ASSESSMENT REPORT

| Report for | Sample Penetration Test Report |
|----------------|---|
| Client contact | +1 (xxx) xxx xxx |
| Report title | Client Name: Web application security and Infrastructure vulnerability report |
| Date | 2019 |
| Version | 1.0 |

Contents

| Executive Summary |
|---|
| Summary of Results6 |
| Conclusion6 |
| Web Application & Infrastructure Vulnerability Assessment7 |
| Target Information7 |
| Methodology7 |
| Testing Setup7 |
| List of vulnerabilities |
| 1. An administrative user for the blog with weak credentials was identified9 |
| Mapping to OWASP/CWE9 |
| Severity9 |
| Technical Description9 |
| Proof of Concept10 |
| So what? |
| Mitigation11 |
| References |
| 2. Multiple SoftwareName Admin users have weak passwords12 |
| Mapping to OWASP/CWE12 |
| Severity12 |
| Technical Description12 |
| Proof of Concept12 |
| So what? |
| Mitigation13 |
| References |
| 3. MySQL password found in cleartext in a world readable file on the server14 |
| Mapping to OWASP/CWE14 |
| <u>Severity14</u> |
| Technical Description14 |
| Proof of Concept14 |
| <u>So what?14</u> |

| Mitigation15 |
|--|
| References:1 |
| 4. Application is vulnerable to Clickjacking attacks16 |
| Mapping to OWASP/CWE |
| Severity16 |
| Technical Description16 |
| Proof of Concept16 |
| So what? |
| Mitigation18 |
| References |
| 5. Sensitive system information leaked via log files exposed over the Internet |
| Mapping to OWASP/CWE |
| Severity19 |
| Technical Description19 |
| Proof of Concept |
| <u>So what?20</u> |
| Mitigation21 |
| References |
| 6. Web Application and Database backup found as world readable files on the server |
| Mapping to OWASP/CWE |
| Severity |
| Technical Description |
| Proof of Concept22 |
| <u>So what?23</u> |
| Mitigation23 |
| References |
| 7. WordPress blog username enumeration possible24 |
| Mapping to OWASP/CWE |
| <u>Severity</u> 24 |
| Technical Description24 |
| Proof of Concept |

| So what?25 | <u>;</u> |
|---|----------|
| Mitigation25 | <u>;</u> |
| References | <u>6</u> |
| 8. WordPress version older than current stable (readme.html found)2 | Z |
| Mapping to OWASP/CWE | Z |
| Severity2 | <u>7</u> |
| Technical Description2 | <u>7</u> |
| Proof of Concept2 | <u>7</u> |
| So what?2 | <u>7</u> |
| Mitigation2 | <u>8</u> |
| References | <u>8</u> |
| 9. SoftwareName version older than current stable | <u>9</u> |
| Mapping to OWASP/CWE | <u>)</u> |
| Severity |) |
| Technical Description |) |
| Proof of Concept | <u>)</u> |
| So what? |) |
| Mitigation |) |
| References |) |
| 10. Missing HSTS header from HTTPS Server | |
| Mapping to OWASP/CWE | |
| Severity | |
| Technical Description | |
| Proof of Concept | |
| | |
| | _ |
| Reference | 2 |
| 11. World writable directories discovered with weak permissions | 3 |
| Mapping to OWASP/CWE | } |
| Severity | - |
| Technical Description | } |
| | _ |

| Proof of Concept |
|---|
| So what? |
| Mitigation |
| References |
| 12. Web server exposes its version number via response headers & body |
| Mapping to OWASP/CWE |
| Severity |
| Technical Description |
| Proof of Concept |
| So what? |
| Mitigation |
| References |
| 13. Older versions of JavaScript libraries being used |
| Mapping to OWASP/CWE |
| Severity |
| Technical Description |
| Proof of Concept |
| <u>So what?</u> |
| Mitigation |
| References |
| Conclusion |
| Annexure 1: Artefacts created on the client-name server and application |
| Annexure 2: Password list used to brute force WordPress login |

1. Executive Summary

We were contracted by client-name to conduct Web Application and Infrastructure Vulnerability Assessment to determine if there were security weaknesses in the client-name website that can render the application insecure and allow an attacker to gain access to any data that is accessible via them or gain access to the underlying operating system.

The hosting infrastructure of the client-name website was also subjected to an External Vulnerability Assessment to simulate a real-world attacker trying to find vulnerabilities and flaws.

The assessment was carried out between date 1 and date 2 on the production web application at <u>www.samplereport.co</u>

OWASP Top 10 2013 was the reference frame work to evaluate and categorise the security issues.

Summary of Results

- The application was found to use weak credentials for its administrative accounts that resulted in a complete compromise of the application.
- There are a few security controls missing in places, especially around the implementation of security HTTP response headers.
- The current configuration of the application has multiple directories with world writable permissions on the server; i.e. any user on the server can write, delete and modify files in the web application.
- Multiple log files containing sensitive information were found on the server, that can be accessed by anyone with a link to the log files, revealing the internal workings of the website.

Conclusion

The web application present doesn't have the security controls to ensure protection against dictionary attacks and brute forcing of the login details. Additionally, weak password policy allowed full access to the application. Once we could login we were able to take over the application and run operating system commands on the server as a limited user.

Using the ability to run commands on the server, we could find additional vulnerabilities affecting the system, including directories that could be modified by any user on the system, weak passwords for the admin console of the SoftwareName app and backup files that contain a lot of sensitive information that could be accessed by any system user.

All these vulnerabilities have resulted in a system that is vulnerable to attack and full server compromise. These issues need to be fixed as a priority. Fixing these issues will give assurance to the users of this application.

2. Web Application & Infrastructure Vulnerability Assessment Target Information

The purpose of this Web Application Security Testing and Vulnerability Assessment was to discover weaknesses, identify threats and vulnerabilities and security issues on the in-scope server.

| Host/Product | Domain/IP Address | Platform | Software | Type of Testing |
|---------------------|---------------------------------------|-----------------------|--|---|
| client-name Site | <u>www.samplerepor</u> <u>t.co</u> | Debian GNU/Linux 8 | Nginx X.X.X SoftwareName X.X.X WordPress 4.X.X | Blackbox Web Application and Infrastructure Vulnerability Assessment |

The scope included the following hosts and types of testing:

| Host/Product | Domain/IP Address | Platform | Open Ports | Type of Testing |
|---|-------------------------|-----------------------|------------|---|
| client-name Site Hosting Infrastructure | www.samplerepor t.co | Debian GNU/Linux 8 | 80, 443 | Blackbox Web Application and Infrastructure Vulnerability Assessment |

Methodology

Testing Setup

We setup an attacker machine in the UK so that our attack traffic was close to the target. The attack traffic originated from the UK IP - XX.XX.XX. This IP was provided to client-name before we began testing. The chain of traffic to the target was as below:

Browser > Burp Suite Pro > Secure SOCKS Tunnel > XX.XX.XX.XX > www.samplereport.co

Our Nessus scanner machine was connected to our UK IP via VPN using a OpenVPN connection to ensure all scanner traffic would go through the IP and end to end encryption was maintained.

3. List of vulnerabilities

- 1. An administrative user with weak credentials was identified
- 2. Multiple SoftwareName Admin users have weak password
- 3. MySQL password found in cleartext in a world readable file on the server
- 4. Application is vulnerable to Clickjacking attacks
- 5. Sensitive system information leaked via log files exposed over the Internet

- 6. Web Application and Database backup found as world readable files on the server
- 7. WordPress blog username enumeration possible
- 8. WordPress version older than current stable (readme.html found)
- 9. SoftwareName version older than current stable
- 10. Missing HSTS header from HTTPS Server
- 11. World writable directories discovered with weak permissions
- 12. Web server exposes its version number via response headers and body 13. Older versions of JavaScript libraries being used



| Severity | Number |
|----------|--------|
| High | 2 |
| Medium | 5 |
| Low | 6 |

4. An administrative user for the blog with weak credentials was identified

We could login into the client-name WordPress blog using a weak set of credentials. These credentials were easy to guess allowing us access to the application, data, functionality available within and to the underlying Operating System as well.

Mapping to OWASP/CWE

| OWASP Top 10 2013 - A2 - Broken | CWE-521: Weak Password Requirements |
|--|---|
| Authentication and Session Management | https://cwe.mitre.org/data/definitions/521.ht |
| https://www.owasp.org/index.php/Top_10_201 | <u>ml</u> |
| <u>3-A2-</u> | |
| Broken Authentication and Session Managem | |
| ent_ | |
| | |

Severity

High

Technical Description

The WordPress blog of client-name hosted at www.samplereport.co/blog/ is vulnerable to a weak password attack. The admin panel was accessible using the username and password combination of admin and DevBrandXX respectively.

The attack approach was as follows:

- At the bottom of the home page at www.samplereport.co we found the text "Powered by DevCompany"
- We did a Google search for the text and found the www.DevCompany.com website.
- This website had the text DevBrand as an image on the home page.
- Based on the two words, DevCompany and DevBrand we created a password list. This password list had various numbers appended to these two words (provided in Annexure 2).
- We then used wpscan¹ to run a brute force on www.samplereport.co/blog using the default username admin and the password list that was generated.
- We obtained a successful login with the username and password combination of admin and DevBrandXX
- Once we were logged in with WordPress admin privileges, we added a PHP file that could execute operating system commands. This file at /wp-content/plugins/xxxxx/index.php was overwritten by a possible file integrity checker program
- We created another file that was capable of running operating system commands at /var/www/samplereport.co/htdocs/upload/filename.php and added some protection to prevent

¹ wpscan is a tool that can be used to find vulnerabilities in WordPress. It also supports username enumeration and brute forcing of logins



other users from abusing this file.

- To execute a command, use this

URL: www.samplereport.co/upload/filename.php?hash=5d62dbXXa25e1e86XXe36275ea1XX55fec0 50aef1XXa914e25a1c816cXXfd31d&passxxxxxxxxx=ifconfig

- This was all possible because of the weak credentials of the admin account.

Proof of Concept

| -WordPr × + | | | | |
|--|---|-------------------------------|------------|--|
| (Intersection of the section of the | | | | |
| ØDisable▼ ▲Cookies▼ 20 | SS- 🔁 Forms- 🔟 Images- 🌐 Information- 🧧 Miscellaneous- 🦯 Outline- 🧪 Resize- 💥 Tools- 🔳 View Source- | <u>∎</u> Options ▼ | | |
| | P 3 + | | | |
| 🚳 Escritorio | | | | |
| 🖈 Entradas 🛛 🧹 | | | | |
| Todas las entradas Añadir nueva | | | | |
| Todos (7) Publicadas (4) Privadas (3) | | | | |
| | Acciones en lote 👻 Aplicar Todas las fechas 👻 Todas las categorías 👻 Filtrar | | | |
| | Título | Autor | Categorías | |
| 93 Medios | | admin | | |
| 📕 Páginas | Editar Edición rápida Papelera Ver | | | |
| 👎 Comentarios 🔕 | | admin | | |
| 🔊 Apariencia | n | admin | | |

Figure 1: WordPress admin panel login



| 5 | × + |
|----------|---|
| (| https://www |
| ODisable | e* 🕹 Cookies* 🧨 CSS* 📴 Forms* 🗐 Images* 🔞 Information* 🧧 Miscellaneous* 🦯 Outline* 🦨 Resize* 💥 Tools* 🔳 View Source* [|
| eth0 | |
| | UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1 RX packets:174982179 errors:0 dropped:250 overruns:0 frame:0 TX packets:279922573 errors:0 dropped:0 overruns:0 carrier:0 collisions:0 txqueuelen:1000 RX bytes:36667988741 (34.1 GiB) TX bytes:375819964829 (350.0 GiB) Interrupt:41 |
| eth2 | |
| | UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1 RX packets:8839813 errors:0 dropped:5577 overruns:0 frame:0 TX packets:7905 errors:0 dropped:0 overruns:0 carrier:0 collisions:0 txqueuelen:1000 RX bytes:674166255 (642.9 MiB) TX bytes:2059114 (1.9 MiB) Interrupt:40 |
| eth3 | |
| | UP BROADCAST RUNNING MULTICAST MTU:1500 Metric:1 RX packets:23596379 errors:0 dropped:5577 overruns:0 frame:0 TX packets:86468485 errors:0 dropped:0 overruns:0 carrier:0 collisions:0 txqueuelen:1000 RX bytes:2659941437 (2.4 GiB) TX bytes:128568529449 (119.7 GiB) Interrupt:44 |
| 10 | |
| | UP LOOPBACK RUNNING MTU:65536 Metric:1 RX packets:21885590 errors:0 dropped:0 overruns:0 frame:0 TX packets:21885590 errors:0 dropped:0 overruns:0 carrier:0 collisions:0 txqueuelen:0 RX bytes:31197606100 (29.0 GiB) TX bytes:31197606100 (29.0 GiB) |

Figure 2: Code execution on the server after logging in as WordPress admin

So what?

An attacker may be able to use a weak password to gain access to the admin panel of the blog. From there, it is very easy to create PHP files that execute the attacker's code on the server thus allowing the attacker to run operating system commands on the server. This will result in a total server compromise.

We achieved this; we created a PHP file that would execute operating system commands on the server resulting in the compromise of user data and system resources without detection.

Mitigation

- 1. Change the passwords of the admin and other accounts.
- 2. Enforce usage of strong passwords. A password strength policy should be created using the recommendations as provided by the OWASP Authentication cheat sheet.



References

A great reference provided by OWASP is the PassFault website. This website allows you to analyse different types of passwords to get an idea on what is secure password. At the very least usernames or variations of that shouldn't be allowed to be set as passwords in any application.

<u>https://www.owasp.org/index.php/Authentication Cheat Sheet#Password Complexity</u> • <u>http://www.passfault.com/</u>

5. Multiple SoftwareName Admin users have weak passwords

We could identify several SoftwareName admin accounts with weak passwords.

Mapping to OWASP/CWE

| OWASP Top 10 2013 - A2 - Broken | CWE-521: Weak Password Requirements |
|--|---|
| Authentication and Session Management | https://cwe.mitre.org/data/definitions/521.ht |
| https://www.owasp.org/index.php/Top_10_201 | <u>ml</u> |
| <u>3-A2-</u> | |
| Broken_Authentication_and_Session_Managem | |
| <u>ent</u> | |

Severity

High

Technical Description

Multiple admin users were discovered who had their password set to DevBrandXX. An attacker can use an admin account with this password to become a SoftwareName admin and gain complete control over the application.

The password is stored in the DB in the form of md5(salt+password). The salt is 473AI8Pho0rMxTWKFXXqfE6DFHYWhopXXgp26TFHobzD8NXX6UyguABi as obtained from the settings.inc.php file. The md5 value of the salt and 'DevBrandXX' combination is 97bb0a177523XXfbf6633a713caXX953. This value was searched through the xx_xxxxxx table in the database and multiple entries were found.



Proof of Concept

| mysql> select email,passwd <swd '<br="" =="">ERROR 2006 (HY000): MySQL s No connection. Trying to re Connection id: Current database:</swd> | from server has gone econnect | where passwd = LIMIT 100; away | T) | ' LIMIT 100; |
|--|-------------------------------------|--------------------------------------|----|--------------|
| + email | + passwd | | + | |
| | | | | |
| | i i | | Ì | |
| | | | | |
| | | | ł | |
| | | | | |
| + 12 rows in set (0.35 sec) | + | | + | |

Figure 3: The SoftwareName admin users with the password 'DevBrandXX'

So what?

An attacker may be able to use a weak password and gain access to the admin panel of the shop. From there, it is very easy to view and edit customer data, transactions, shopping history, the look and feel of the site, the UI and the menus etc. Gaining access to a set of admin credentials will result in a total application compromise.

Mitigation

- 1. Change the passwords of the admin and other accounts.
- 2. Enforce usage of strong passwords. A password strength policy should be created using the recommendations as provided by the OWASP Authentication cheat sheet.

References

A great reference provided by OWASP is the PassFault website. This website allows you to analyse different types of passwords to get an idea on what is secure password. At the very least usernames or variations of that shouldn't be allowed to be set as passwords in any application.

•



6. MySQL password found in cleartext in a world readable file on the

server

A file containing MySQL credentials was discovered on the web server in a world readable file. Any user who has access to the server can read the files to obtain the database credentials.

Mapping to OWASP/CWE

| OWASP Top 10 2013 - A6-Sensitive Data CV | CWE-285: Improper Authorization |
|---|---|
| Exposure htt | https://cwe.mitre.org/data/definitions/285.ht |
| https://www.owasp.org/index.php/Top_10_201 ml | <u>nl</u> |
| 3-A6-Sensitive Data Exposure | |

Severity

Medium

Technical Description

The following files have MySQL credentials in plaintext. These files have world readable permissions:

/home/ samplereport /xxxx/yyyyyy_aabb_zzzzfiles.sh /home/ samplereport /xxxx/yyyyyy_zzzzfiles.sh

The MySQL credentials discovered were: usr_web:Xy123X1xyz4X

Proof of Concept

| www-data@ | :/home | | | | | | |
|--------------------|----------|-------------|--------------|-------------------|-----------------|-----------------|--------|
| ls -ltra | | | | | | | |
| total 6653868 | | | | | | | |
| -rw-rr 1 | | 401744 | 12:28 | | | | |
| -rw-rr 1 root | root | 78381798 | 12:45 | | | | |
| -rw-rr 1 root | root | 443 | 09:45 | | | | |
| -rw-rr 1 root | root | 88490933 | 20:28 | | | | |
| -rw-rr 1 root | root | 3302563840 | 10:47 | | | | |
| -rw-rr 1 root | root | 3343687680 | 12:51 | | | | |
| -rw-rr 1 | | 300 | 13:05 | | | | |
| drwxr-xr-x 2 | • | 4096 | 13:05 | | | | |
| drwxr-xr-x 6 | | 4096 | 09:04 | | | | |
| www-data@ | :/home | / / | cat sinc* | | | | |
| cat sinc* | | | | | | | |
| #rsvnc -avzPexc | lude 'de | plov'exclu | ude 'config' | exclude 'cach | e'exclude 'blog | /wp-config.php' | root |
| sts/ | 1 | / /var/www/ | /h | tdocs/ | | ,, <u>r</u> | |
| mysgl -uusr web | | | < | /var/www/ | /htdocs/ | | |
| mysgl -uusr web | | | | /var/www/ | /htdocs/ | | |
| #chown -R www-data | | a /var/www/ | ÷ /ht | docs | | | |
| rsvncexcl | ude 'dep | lov'exclud | de '*.log' | exclude 'config | 'exclude 'img' | exclude 'inde | x.php' |
| e 'blog/wp-config. | php' roo | te | :/var/www | / · · · / · · · · | | /var/www/ | |
| chown -R www-data. | www-data | /var/www/ | /htc | ocs | | | |
| www-data@ | :/home | / . / . | | | | | |

Figure 4: MySQL credentials discovered in a plaintext file on the server



So what?

A user with limited privileges can read these files and obtain the username and password to connect to the MySQL database. Once connected, the user will have the ability to see and access all data regarding the client-name blog and shop. This data includes user information, shopping history and payment information if any.

Mitigation

Change the permission on these files so that other users cannot read them. Also, use the mysql_config_editor² to generate a MySQL configuration file that contains the encrypted value of the password. The permissions on this file should also be restricted to prevent access by other users.

References:

- https://cwe.mitre.org/data/definitions/521.html
- <u>https://opensourcedbms.com/dbms/passwordless-authentication-usingmysql_config_editor-with-mysql-5-6/</u>

² mysql_config_editor is a program that gets installed as part of the mysql-client suite of utilities. To install this utility, run *sudo apt-get install mysql-client-5.6*



7. Application is vulnerable to Clickjacking attacks

The X-Frame-Options header that has not been set on the application. If a page fails to set an appropriate X-Frame-Options, it can be possible for a page controlled by an attacker to load it within an iframe. This may enable a clickjacking attack, in which the attacker's page overlays the target application's interface with a different interface provided by the attacker.

Mapping to OWASP/CWE

| OWASP Top 10 2013 - A5-Security | CWE-693: Protection Mechanism Failure |
|---------------------------------------|---|
| Misconfiguration | <u>https://cwe.mitre.org/data/definitions/693.ht</u> ml |
| <u>3-A5-Security Misconfiguration</u> | CWE-451: User Interface (UI) Misrepresentation of Critical Information |
| | <u>https://cwe.mitre.org/data/definitions/451.ht</u> <u>ml</u> |

Severity Medium

Technical Description

The application server does not set the X-Frame-Options header allowing the application to be loaded in an iframe. This can lead to Clickjacking attacks where the attacker can trick a user to click on HTML elements in the application by loading it in a hidden iframe.

Proof of Concept

| Request | Response | |
|---|--|---|
| Raw Headers Hex | Raw Headers Hex HTML Render | |
| DET /es/HTTP/1.1 | HTTP/1.1 200 OK | |
| Host: www | Server: nginx/1.10.1 | |
| User-Agent: Mozilla/5.0 (Windows NT 10.0; WOW64; rv:51.0) Gecko/20100101 Firefox/51.0 | Content-Type: text/html; charset=utf-8 | |
| Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8 | Connection: close | |
| Accept-Language: en-US,en;q=0.5 | Vary: Accept-Encoding | |
| Connection: close | P3P: CP="] | " |
| Upgrade-Insecure-Requests: 1 | Powered-By: | |
| | Set-Cookie: | |
| | l3afaffd853795d21e | =2%2BM7fraEmYpQR3ehGtQTgD9Eit39m8KTuVb5eWVNEdJ |
| | JFDAr1co2MER455JrXg | |
| | ZC778Rv8gJ | GQTOug963D000115; expires=Sat, 19-Dec-2071 12:15:34 |
| | GMT; Max-Age=1727999999; path=/; doma | in=www. secure; httponly |
| | Date: GMT | |
| | X-Page-Speed: 1.11.33.2-0 | |
| | Cache-Control: max-age=0, no-cache | |
| | Content-Length: 450014 | |





| \int | | | × | + | | | | | |
|--------|---------|----------------|-----------|----------|--------|---------------|-----------|-------|------------------|
| 4 | (i) 💋 | | :8000 | | | | | | |
| ØDi | isable▼ | L Cookies - | ZCSS▼ 🛄 F | orms 🕶 🗳 | Images | (Information) | Miscellan | eous▼ | /Outline - / Res |

site can be loaded in an iFrame

| MIS DATO | S | |
|-----------------|-------|---|
| | | |
| | | |
| DATOS PERSON | IALES | |
| | | |
| * NOMBRE | | |
| Abc | | |
| * APELLIDO | | |
| а | | |
| * APELLIDO | | |
| | | |
| <u>.</u> | | - |
| DATOS DE ACC | ESO | |
| | | |
| * CORREO ELECTR | ÓNICO | |
| | | |
| * CONTRASEÑA AC | TUAL | |
| | | |
| | 5 A | |

Figure 6: The client-name user account page loaded in an iframe making clickjacking attacks possible



So what?

An attacker can load any of the pages that are available via the application and trick a user to perform mouse clicks or send keystrokes causing an unwitting action to be performed. An attacker can target any functionality of the application that a user has access to by tricking the user into performing mouse clicks or send keystrokes that can result in adding of new data or modifying/deleting existing information available in the application.

Mitigation

To effectively prevent framing attacks, the application should return a response header with the name X-Frame-Options and the value DENY to prevent framing altogether, or the value SAMEORIGIN to allow framing only by pages on the same origin as the response itself. This can be done in nginx by adding the following to the nginx.conf or the site configuration file in the sites-enabled directory, in the server section:

add_header X-Frame-Options SAMEORIGIN;

References

- <u>https://www.owasp.org/index.php/Clickjacking_Defense_Cheat_Sheet_</u>
- <u>http://blog.kotowicz.net/2009/12/5-ways-to-prevent-clickjacking-on-your.html</u>
- <u>https://geekflare.com/add-x-frame-options-nginx/</u>



8. Sensitive system information leaked via log files exposed over the

Internet

The client-name application discloses sensitive information via log files that are stored in webroot and which can be accessed via the browser.

Mapping to OWASP/CWE

| OWASP Top 10 2013 - A5-Security | CWE-200: Information Exposure |
|--|---|
| Misconfiguration | https://cwe.mitre.org/data/definitions/200.ht |
| https://www.owasp.org/index.php/Top_10_201 | <u>ml</u> |
| <u>3-A5-Security Misconfiguration</u> | |
| Misconfiguration https://www.owasp.org/index.php/Top_10_201 3-A5-Security_Misconfiguration | <u>https://cwe.mitre.org/data/definitions/200.ht</u> <u>ml</u> |

Severity

Medium

Technical Description

The www.samplereport.co/log/ directory contains log files that have easy to predict log filenames. An attacker can enumerate and download these log files since they are in a public facing directory. These log files contain a wealth of information in the form of error messages, debug messages and failed queries. Table and database names are disclosed through these log files.

The list of log filenames that we were able to enumerate are

| 12345123_exception.log | 12345123_exception.log | 12345123_exception.log |
|------------------------|------------------------|------------------------|
| 12345123_exception.log | 12345123_exception.log | 12345123_exception.log |
| | | |

©Client_Name | Penetration test report by Aristi Cybertech Private Limited



12345123_exception.log 12345123_exception.log

Proof of Concept



Figure 7: Log file in a publicly accessible directory



| GET request to | /log/exception.log | | <u>12</u> 2 | □ × |
|------------------|--------------------|----------|-------------|--------|
| Request Response | [| Previous | Next | Action |
| Raw Headers Hex | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

Figure 8: One of the log files containing sensitive data. In this case SQL queries that reveal table and column names.

So what?

An attacker can use the error messages contained within these files to perform additional attacks on the application. These files contained detailed error messages that contain database name, table and column information and other SQL errors.

Mitigation

Move the log files to a directory outside webroot. Also, make sure that the log files do not contain any sensitive information like credit card numbers, PII or credentials.

The server can also be configured to prevent .log files from being served using the following configuration inside the server section of the site configuration file or the nginx.conf file

```
location ~ \.log$ {
return 403;
}
```

References

- https://www.owasp.org/index.php/Information_Leakage_
- <u>http://stackoverflow.com/questions/8286214/how-to-block-all-file-extensions-of-certaintypes-on-nginx</u>



9. Web Application and Database backup found as world readable files on the server

A copy of the database and web application directory were found as backup files in the home directory of a system user. These files had world readable permissions allowing any user on the system to access these files and read their contents.

Mapping to OWASP/CWE

| OWASP Top 10 2013 - A5-Security | CWE-200: Information Exposure |
|--|---|
| Misconfiguration | https://cwe.mitre.org/data/definitions/200.ht |
| https://www.owasp.org/index.php/Top 10 201 | <u>ml</u> |
| <u>3-A5-Security_Misconfiguration</u> | |

Severity

Medium

Technical Description

The following files containing the MySQL database backup and the website backup were found. These files have world readable permissions:

/home/xxxx/vvvv/live_backup.sal /home/xxxx/vvvv/ samplewebsite.tar /home/xxxx/yyyy/htdocs_XX-XX-XX.tar

To find world readable files run the following command:

find /home/ -type f -perm -o=r -exec ls -ltra {} \;



Proof of Concept

| www-datag :/ // // 13 - |
|---|
| ls -ltra |
| total 6653868 |
| -rw-rr 1 401744 Jansql |
| -rw-rr 1 root root 78381798 Jan .sql |
| -rw-rr 1 root root 443 Feb .sh |
| rw-rr 1 root root 88490933 Feb Live backup.sgl |
| -rw-rr 1 root root 3302563840 Feb .tar |
| -rw-r1 root root 3343687680 Feb htdocs .tar |
| -rw-rr 1 300 Feb sincro .sh |
| drwxr-xr-x 2 4096 Feb . |
| drwxr-xr-x 6 diameter 4096 Mar and a contract of the second |
| www-data@ :/home/ / tail -n 20 live backup.sgl |
| tail -n 20 live backup.sul |
| |
| Dumping data for table we users |
| |
| |
| LOCK TARLES 'WD HEARS' WDITE: |
| AND DITTED TREE LOOP DECEMPTION OF TRANSFERRED AND TREE TREE TREE TREE TREE TREE TREE TRE |
| A 19900 ADDA FADDA WE GOT DIOADDA RAID // // Lindmini |
| inseki into wp_users valoes (, aunin , , , , , , , , , , , , , , , , , , |
| |
| / // / / / / / / / / / / / / / / / / / |
| /: 40000 ALIER TABLE WP_USERS ENABLE REIS -// |
| UNLOUG TADLES; |
| /~:40103 SET TIME_ZONE=GOLD_TIME_ZONE //; |
| |
| /*!40101 SET SQL_MODE=@OLD_SQL_MODE */; |
| /*140014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */; |
| /*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */; |
| /*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */; |
| /*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */; |
| /*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */; |
| /*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */; |
| |
| Dump completed on 2017- |
| www-data@ :/home/ / |

Figure 9: Backup files found on the server with world readable permissions

So what?

A user with limited privileges can read these files and gain access to all data that was backed up. The database backup contains information about the client-name blog and shop. The data in the backups may include website files, custom source code, user information, shopping history and payment information.

Mitigation

Change the permission on these files so that other users cannot read them. These files should not be accessible to non-root users or only to users who have been designated as system backup operators.

References

<u>https://www.owasp.org/index.php/Information_Leakage</u>



10. WordPress blog username enumeration possible

It is possible to enumerate the users of the WordPress blog inside the client-name site.

Mapping to OWASP/CWE

| OWASP Top 10 2013 - A5-Security | CWE-200: Information Exposure |
|--|---|
| Misconfiguration | https://cwe.mitre.org/data/definitions/200.ht |
| https://www.owasp.org/index.php/Top 10 201 | <u>ml</u> |
| <u>3-A5-Security_Misconfiguration</u> | |

Severity

Medium

Technical Description

The WordPress installation does not offer any protection for username enumeration. It was possible to list the users in WordPress using wpscan and by manual methods as well.

Using wpscan we can enumerate users by running the command: wpscan -u www.samplereport.co/blog/ --enumerate u

Manually, it is possible to enumerate users by following the redirection of pages when browsing to the following URLs:

www.samplereport.co/blog/?author=1 www.samplereport.co/blog/?author=2 www.samplereport.co/blog/?author=3 www.samplereport.co/blog/?author=4 www.samplereport.co/blog/?author=5 Proof of Concept



Figure 10: Username enumeration possible via HTTP redirects



| æ . | champ@toolsmachina-v1-512mb-lon1-01 | : ~/tools/wpscan | | |
|---------------------------------|--|---|---|-------------------|
| cha | amp@toolsmachina-v1-512m | mb-lon1-01:~/tool | ls/wpscan\$,/wpscan.rb -u | /blog/enumerate u |
| | | | | |
| | WordPress Security Ver Sponsored by Suc @_WPScan_, @ethicalhack | y Scanner by the ssion 2.9.1 suri - https://su c3r, @erwan_lr, p | WPScan Team Jouri.net ovdl, @_FireFart | |
| [?] [+] [+] | The remote host tried] Do you want follow the] URL:] Started: Tue | to redirect to: e redirection ? [| 'blog/es/ [Y]es [N]o [A]bort, default: [N]N | |
| [+] [+] [+] |] The WordPress] Interesting header: SH] Interesting header: X-] XML-RPC Interface avai | CRVER: nginx/1. PAGE-SPEED: 1.11 lable under: | 'blog/readme.html' file exists exposing 1.33.2-0 /blog/ .php | a version number |
| [+] | WordPress version 4. | (Released on | identified from meta generate | or, links opml |
| [+] [+] |] Enumerating plugins fr] No plugins found | com passive detec | ction | |
| [+] [+] | Enumerating usernames Identified the followi | ng 4 user/s: | + | |
| | Id Login | , Name | I | |
| | 1 admin 3 4 5 | admin | | |
| [!] |] Default first WordPres | s username 'admi | in' is still used | |
| [+] [+] [+] [+] cha | Finished: Tue Requests Done: 58 Memory used: 27.566 MF Elapsed time: 00:01:09 amp@toolsmachina-v1-512m | 3) nb-lon1-01:~/tool | Ls/wpscan\$ | |

Figure 11: Username enumeration possible via wpscan

So what?

An attacker can enumerate usernames to attempt password brute force attacks. To brute force and successfully login into the blog administration portal, two key things are required, a valid username and its password. This vulnerability exposes the username which can be used in conjunction with a large password list to brute force the account and login to the application. This will result in the compromise of the application and consequently with the help of other vulnerabilities that may exist post login, a compromise of the underlying system.

Mitigation

Block or redirect any requests to /?author using the nginx configuration on the server.

Edit the webserver's configuration to add rules to rewrite the location when an attempt is made to access the <u>/?author</u> request from the URL. This should be done inside the server section of the configuration.

An example of the nginx configuration with this enabled is shown below:



```
if ($args ~ "^/?author=([0-9]*)"){
  set $rule_0 1$rule_0;
        }
        if ($rule_0 =
"1"){
           rewrite ^/$ www.samplereport.co/blog/404 permanent;
        }
```

References

- <u>https://www.owasp.org/index.php/Information_Leakage</u>
- <u>http://www.edwiget.name/2013/10/blocking-wordpress-user-enumeration-on-nginx/</u>



11. WordPress version older than current stable (readme.html found)

The version of WordPress running on the server (4.X.X) is older than the current stable release (4.7).

Mapping to OWASP/CWE

| OWASP Top 10 2013 – A9- Using Components with Known Vulnerabilities | CWE-200: Information Exposure <u>https://cwe.mitre.org/data/definitions/200.html</u> |
|--|--|
| https://www.owasp.org/index.php/Top_10_2013- | |
| <u>A9-</u> | |
| Using Components with Known Vulnerabilities | |

Severity

Low

Technical Description

The website has a sub folder called /blog that has a WordPress instance running. This instance of WordPress is at version 4.X.X. The latest stable version is 4.7.3 as of XX March 2017.

The version number of the installed WordPress was discovered via the readme.html file located at: www.samplereport.co/blog/readme.html

Proof of Concept



Figure 12: Version of WordPress is older than current stable release of 4.7 (as of XX March 2017)



So what?

Systems running software with older versions may become vulnerable due to weakness in the running software. An attacker may use publicly disclosed vulnerabilities to gain access to the application data or user information resulting in a possible compromise of the application or the server running the application.

Mitigation

Upgrade to the latest version of WordPress. WordPress administrators receive update notifications in the admin dashboard which can be used to start the upgrade process. Independently, you can also obtain the latest version of WordPress from the vendor site at <u>https://wordpress.org/.</u>

Use the following techniques to remove the version number from the other areas of the application as well:

- 1. Delete the readme.html file from webroot as this gives away the version of WordPress.
- 2. Add the following lines of php code to the end of the functions.php file found in your /wp-content/themes/your-theme-name/ folder.

```
remove_action('wp_head',
'wp_generator'); function
remove_version() { return ''; }
add_filter('the_generator', 'remove_version');
```

References

- <u>https://www.owasp.org/index.php/Top_10_2013-A9Using_Components_with_Known_Vulnerabilities</u>
- <u>https://codex.wordpress.org/Updating_WordPress</u>



12. SoftwareName version older than current stable

The version of SoftwareName running on the server (X.X.X) is older than the current stable release (Y.Y).

Mapping to OWASP/CWE

| OWASP Top 10 2013 – A9- Using Components | CWE-200: Information Exposure |
|--|--|
| with Known Vulnerabilities | <u>https://cwe.mitre.org/data/definitions/200.html</u> |
| https://www.owasp.org/index.php/Top 10 2013- <u>A9-</u> <u>Using Components with Known Vulnerabilities</u> | |

Severity

Low

Technical Description

The website is primarily running a SoftwareName instance. This instance of SoftwareName is at is at version X.X.X. The latest stable version is Y.Y as of XX March 2017.

The version number of the installed SoftwareName was discovered by reading the contents of /var/www/samplereport.co /htdocs/config/settings.inc.php

Proof of Concept

| (\equiv) https:// | | | C 🧃 | Search |
|---|---------------|-----------------|------------|----------------|
| ⊘Disable▼ LCookies▼ ZCSS▼ DForms▼ Images▼ | ●Information▼ | Miscellaneous - | /Outline - | Resize 🛪 💥 Too |
| php</td <td></td> <td></td> <td></td> <td></td> | | | | |
| <pre>define('_DB_SERVER_', 'localhost');</pre> | | | | |
| <pre>define(' DB NAME ', ');</pre> | | | | |
| <pre>define('_DB_USER_', '');</pre> | | | | |
| <pre>define('_DB_PASSWD_', '');</pre> | | | | |
| <pre>define('_DB_PREFIX_', 'ps_');</pre> | | | | |
| <pre>define('_MYSQL_ENGINE_', 'InnoDB');</pre> | | | | |
| define('_PS_CACHING_SYSTEM_', 'CacheMemcac | he'); | | | |
| <pre>define(' PS CACHE ENABLED ', '0');</pre> | | | | |
| define('_COOKIE_KEY_', ' | | | | '); |
| <pre>define('_COOKIE_IV_', ' ');</pre> | | | | |
| <pre>define('_PS_CREATION_DATE_', ' ')</pre> | ; | | | |
| if (!defined('_PS_VERSION_')) | | | | |
| define(' PS VERSION ', ' '); | | | | |
| define('_! KEY_', ' | | '); | | |
| define('', ' | =='); | | | |

Figure 13: Version of SoftwareName is older than current stable release of Y.Y (as of XX March 2017)



So what?

Systems running software with older versions may become vulnerable due to weakness in the running software. An attacker may use publicly disclosed vulnerabilities to gain access to the application data or user information resulting in a possible compromise of the application or the server running the application.

Mitigation

Upgrade to the latest version of SoftwareName.

References

<u>https://www.owasp.org/index.php/Top_10_2013-</u>

A9Using Components with Known Vulnerabilities

• <u>http://doc.SoftwareName.com/display/PS16/Automatic+update</u>



13. Missing HSTS header from HTTPS Server

The application fails to prevent users from connecting to it over unencrypted connections.

Mapping to OWASP/CWE

| OWASP Top 10 2013 – A5-Security Misconfiguration | CWE – Not available |
|---|---------------------|
| https://www.owasp.org/index.php/Top 10 2013- A5-Security_Misconfiguration_ | |

Severity

Low

Technical Description

The HTTP "Strict-Transport-Security" header is missing in the HTTPS responses from the application server. The lack of HSTS allows downgrade attacks, SSL-stripping man-in-the-middle attacks, and weakens cookie-hijacking protections.

Proof of Concept

| Go Cancel < * > * | | | Target: https://www.lasirena.es 📝 |
|---|---|------------------|-----------------------------------|
| Raw Headers Hex DBT //er/HTTP/1.1 Mote | Response Raw Headers Hex HTML Render HTTP/1.1 200 OK Server poinx/1.10.1 | | |
| Accepts 1,* Accepts 1,* Conserv:Language: en User-Agent: MosIIII/S0 (compatible; MSIE 9.0; Windows NT 6.1; Win64; x64; Trident/S.0] Connection: close | Context-Type:tox/Atml;charset=utf-8 Connection:close Vary:AcceptBooding X:UA-Compatible:IB=edge.chrome=1 P3P:CP=" Powered-By: Set-Cookle: | | |
| | Max-Age=1728000000; path=/; domain= Date: Pri, GMT X-Page-Speed; 1.11.33.2-0 Cache-Control max-age=0, no-cache Content-Length: 348469 | secure; httponly | |

Figure 14: Missing strict transport security header in the response header

So what?

An attacker able to modify a legitimate user's network traffic could bypass the application's use of SSL/TLS encryption, and use the application as a platform for attacks against its users. This attack is performed by rewriting HTTPS links as HTTP, so that if a targeted user follows a link to the site from an HTTP page, their browser never attempts to use an encrypted connection.

Mitigation

A Strict-Transport-Security HTTP header should be sent with each HTTPS response. The syntax Strict-Transport-Security: max-age=<seconds>[; includeSubDomains]



The parameter max-age gives the time frame for requirement of HTTPS in seconds and should be chosen quite high, e.g. several months. The flag includeSubDomains defines that the policy applies also for sub domains of the sender of the response. This will ensure that a supported browser will enforce secure connection while communicating with the application. Reference

- <u>https://www.owasp.org/index.php/HTTP_Strict_Transport_Security_Cheat_Sheet_</u>
- <u>https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Strict-Transport-Security</u>



14. World writable directories discovered with weak permissions

Several directories were discovered in the web root folder that had weak permissions set on them. These directories had world writable permissions set, allowing any user on the system to add/modify/delete content inside them.

Mapping to OWASP/CWE

| OWASP Top 10 2013 - A5-Security | CWE-693: Protection Mechanism Failure |
|--|---|
| Misconfiguration https://www.owasp.org/index.php/Top_10_201 3-A5-Security_Misconfiguration | <u>https://cwe.mitre.org/data/definitions/693.ht</u> <u>ml</u> |

Severity

Low

Technical Description

Using the shell that was uploaded using the weak WordPress credentials (see <u>vulnerability 1 - An</u> <u>administrative user for the blog with weak credentials was identified</u>), we could identify directories that have weak permissions. These permissions allow any user on the system to write to into these folders.

www.samplereport.co/upload/filename.php?hash=5d62dbXXa25e1e86XXe36275ea1XX55fec050aef 1XXa914e25a1c816cXXfd31d&passxxxxxxxxxxxxxx =ls%20-ltrad%20/var/www/ samplereport.co /htdocs/*/%20|%20grep%20drwxrwxrwx Without the encoded characters, the URL is

www.samplereport.co/upload/filename.php?hash=5d62dbXXa25e1e86XXe36275ea1XX55fec050aef 1XXa914e25a1c816cXXfd31d&passxxxxxxxxxxxxx =ls -ltrad /var/www/samplereport.co/htdocs/*/ | grep drwxrwxrwx

The following directories were found to have world read and writable permissions set

- 1. /var/www/samplereport.co/htdocs/xxxxx/
- 2. /var/www/samplereport.co/htdocs/yyyyy/
- 3. /var/www/samplereport.co/htdocs/zzzzz/
- 4. /var/www/samplereport.co/htdocs/aaaaaa/
- 5. /var/www/ samplereport.co /htdocs/bbbbb/
- 6. /var/www/ samplereport.co /htdocs/cccccc/
- 7. /var/www/ samplereport.co /htdocs/dddd/



Proof of Concept

| | | × + | | | | | | | | |
|-----------------|-------|-----------------|------------------|------------|------------|--|-------------------------|------|---|---|
| 🗲 🛈 🔒 https:// | / | | | | | | E] 14 | 0% C | | Q |
| ODisable LCooki | es• 🌶 | CSS 🕶 🔯 Forms 🕶 | 💷 Images 🔹 🕕 Inf | formation▼ | Miscellane | eous 🛛 🥖 Outline 🕶 🥒 Resize 🔹 💥 Tools 🖛 🔳 Vi | ew Source 🔹 🖸 Options 🕶 | | | |
| drwxrwxrwx | 1 | 9 www-data | www-data | 4096 | Mar | /var/www/- | 'htdocs/ | | | |
| drwxrwxrwx | 1 | 5 www-data | www-data | 4096 | Mar | /var/www/ | 'htdocs/ | | ' | 1 |
| drwxrwxrwx | | 2 www-data | www-data | 4096 | Mar | /var/www/ | 'htdocs/ | | | |
| drwxrwxrwx | 12 | 5 www-data | www-data | 4096 | Mar | /var/www/ | 'htdocs/: | | | |
| drwxrwxrwx | 2 | 4 www-data | www-data | 4096 | Mar | /var/www/ | 'htdocs/ | | | |
| drwxrwxrwx | 1 | 8 www-data | www-data | 4096 | Mar | /var/www/ | 'htdocs/ | | | |
| drwxrwxrwx | | 3 www-data | www-data | 4096 | Mar | /var/www/ | 'htdocs/ | | | |

Figure 15: Several directories in webroot were found to have weak permissions

So what?

A system user can create/edit files inside the directories. A malicious file can be created that steals client-name shop's user's passwords, tokens or payment information resulting in data and confidentiality loss for the site's users.

Mitigation

Remove world write permissions on directories inside webroot using the chmod command.

The following is an example of the chmod command to remove world write permission on a folder:

chmod 775 directory_name

So, to remove the world writable permission on /var/www/ samplereport.co /htdocs/ upload folder run the following command:

chmod 775 /var/www/ samplereport.co/htdocs/upload

References

- https://www.owasp.org/index.php/Top 10 2013-A5-Security Misconfiguration
 - https://www.owasp.org/index.php/File_System#Insecure_permissions_

15. Web server exposes its version number via response headers & body

The server software versions used by the application are revealed by the web server in the response header and the response body when access restricted folders/files.

Mapping to OWASP/CWE



| OWASP Top 10 2013 - A5-Security | CWE-200: Information Exposure |
|--|---|
| Misconfiguration | https://cwe.mitre.org/data/definitions/200.ht |
| https://www.owasp.org/index.php/Top_10_201 | <u>ml</u> |
| <u>3-A5-Security_Misconfiguration</u> | |

Severity

Low

Technical Description

The webserver is configured to send the version of nginx to the client. The version number is exposed via the response header and in the body of the page when accessing a restricted directory. The version number of nginx is X.X.X as obtained from this misconfiguration.

A HTTP response to any page on the site will have the nginx version in the response header as shown in the screenshots.

A HTTP response to a restricted directory like /backoffice will have the nginx version in the response body as shown in the screenshots.

Proof of Concept



Figure 16: The HTTP response headers contain version number of the nginx server



| 4 | 03 Fo | rbidden | | | × \ + | | | | | |
|----|-------|---------|---------|------|-----------|------------------|-----|---------|-----|------------------------|
| + |) > | () | https:/ | // | | | C | - | | 🔍 Search |
| ØD | isabl | e• 🕹 C | ookies▼ | CSS- | 🔁 Forms 🕶 | Information* | Mis | cellane | ous | 🖉 Outline 🕶 🥒 Resize 🕶 |

403 Forbidden

nginx/

Figure 17: The HTTP response body contains the version number of the nginx server

So what?

Systems running software with older versions may become vulnerable due to weakness in the running software. An attacker may use publicly disclosed vulnerabilities to gain access to the application data or user information resulting in a possible compromise of the application or the server running the application.

Mitigation

To hide the version of nginx in the response header and body, set the server_tokens value in the nginx configuration file to Off. This will prevent the server from disclosing the version information. The nginx configuration file is present at /etc/nginx/nginx.conf

References

- 1. <u>https://www.owasp.org/index.php/Information_Leakage_</u>
- 2. <u>https://www.digitalocean.com/community/tutorials/how-to-secure-nginx-on-ubuntu-14-04</u>

16. Older versions of JavaScript libraries being used

The client-name SoftwareName application as well as the WordPress blog, both use older versions of JavaScript libraries that are vulnerable to attacks.

The JavaScript file jquery-X.X.X.min.js and the JavaScript file jquery-migrate-X.X.X.min.js include a vulnerable version of the library 'jquery'.

Mapping to OWASP/CWE



| OWASP Top 10 2013 – A9- Using Components | CWE-200: Information Exposure |
|---|--|
| with Known Vulnerabilities | <u>https://cwe.mitre.org/data/definitions/200.html</u> |
| https://www.owasp.org/index.php/Top_10_2013- A9- Using Components with Known Vulnerabilities_ | |

Severity

Low

Technical Description

The application uses older versions of the jQuery library. The versions found to be in use are jQuery - v X.X.X and jQuery migrate - v X.X.X. The version numbers can be obtained by accessing the files at: www.samplereport.co/js/jquery/jquery-migrate-X.X.X.min.js www.samplereport.co/js/jquery/jquery-X.X.X.min.js

Proof of Concept

| X | + | |
|--|---|---|
| 🗲 🛈 🔒 https:// | .min.js | |
| ØDisable▼ 🕹 Cookies▼ 🗡 CSS▼ 🔯 For | ms 🕶 💷 Images 🔹 🕕 Information 🕶 | 📕 Miscellaneous 🛛 🥖 Outline 🔹 🧳 Resize 👻 |
| <pre>/*! jQuery v !function(a,b){"ob {if(!a.document)th window?window:this {},i=h.toString,j= \xA0]+ [\s\uFEFF\x {jquery:m,construct</pre> | <pre> (c) oject"==typeof m row new Error(" ,function(a,b){ h.hasOwnPropert A0]+\$/g,p=/^-ms ctor:n,selector:</pre> | jQuery Foundation odule&&"object"==typ jQuery requires a wi var c=[],d=c.slice,e y,k="".trim,l={},m=" -/,q=/-([\da-z])/gi, "",length:0,toArray: |

Figure 18: jQuery version at X.X.X





/*! jQuery Migrate | jQuery Fou jQuery.migrateMute===void 0&&(jQuery.migrateMute=!0), (i[n]=!0,e.migrateWarnings.push(n),r&&r.warn&&!e.migr "+n),e.migrateTrace&&r.trace&&r.trace()))}function a(Object.defineProperty(t,a,{configurable:!0,enumerable {r(o),i=e}}),n}catch(s){}e._definePropertyBroken=!0,t t.console.log&&t.console.log("JQMIGRATE: Logging is a (e.migrateTrace=!0),e.migrateReset=function(){i={},e. r("jQuery is not compatible with Quirks Mode");var o=

Figure 19: jQuery-migrate at v X.Y.X

So what?

It may be possible for an attacker to load JavaScript in the browser using a jQuery call. This JavaScript can be used to perform any action as a client-name user since it will be running in the context of the client-name website.

Ideally, using JQuery, applications should not be able to execute JavaScript from third party domains.

Mitigation

Update to the latest version of JQuery via the vendor site: https://jquery.com/download/. The current version, as of this report, is 3.2.0 (as of XX March 2017)

References

 <u>https://www.owasp.org/index.php/Top_10_2013-A9-</u> Using_Components_with_Known_Vulnerabilities_

17. Conclusion

The application at <u>https://www.samplereport.co</u> and its hosting infrastructure were subjected to a Web Application Security Test and a Vulnerability Assessment to identify weaknesses that can render the application insecure and allow an attacker to gain access to any data that is accessible via them or gain access to the underlying operating system.

The web application present doesn't have the security controls to ensure protection against dictionary attacks and brute forcing of the login details. Additionally, weak password policy allowed full access to the application. Once we could login we were able to take over the application and also run operating system commands on the server as a limited user.



Using the ability to run commands on the server, we were able to find additional vulnerabilities affecting the system, including directories that could be modified by any user on the system, weak passwords for the admin console of the SoftwareName app and backup files that contain a lot of sensitive information that could be accessed by any system user.

All these vulnerabilities have resulted in a system that is vulnerable to attack and full server compromise. These issues need to be fixed as a priority. Fixing these issues will give assurance to the users of this application.



18. Annexure 1: Artefacts created on the client-name server and application

The following objects were created either on the server or in the application. Please remove them once verification of this report has been completed.

1. Backdoor shell at /var/www/samplereport.co/htdocs/upload/xxx.php

This file was created to run commands on the operating system as part of the assessment. All commands that were run were non-intrusive. No other files were created on the server apart from this file.

This file can be accessed using a web browser and a secret hash. For example www.samplereport.co/upload/filename.php?hash=5d62dbXXa25e1e86XXe36275ea1XX55fec05 Oaef1XXa914e25a1c816cXXfd31d&passxxxxxxxxxx = ifconfig

2. A user with username yyyyyyy@XXXXXXX.COM was created as part of the test. This user was created on the client-name SoftwareName instance using the register new user functionality of the application.



19. Annexure 2: Password list used to brute force WordPress login

The following list of strings were put into a file called passwordlist.txt and used to brute force the login for the WordPress admin user.

DevCompany DevCompany DevCompany DevCompany DevBrand DevBrand DevCompany0 DevCompany0 DevCompany0 DevCompany0 DevBrand0 DevBrand0 DevCompany1 DevCompany1 DevCompany1 DevCompany1 DevBrand1 DevBrand1 DevCompany00 DevCompany00 DevCompany00 DevCompany00 DevBrand00 DevBrand00 DevCompany11 DevCompany11 DevCompany11 DevCompany11 DevBrand11 DevBrand11