



ECR Security
Assessment Report
For:

SAMPLE

SAMPLE Security Assessment Report

Revision History

Date	Version	Description	Author
06/10/2019	1	Final report	Brian Milliron

SAMPLE Security Assessment Report

Table of Contents

Revision History	pg. 2
Executive Summary	pg. 4
Objective	pg. 6
Assessment Scope	pg. 6
Assessment Tools	pg. 6
Target Systems	pg. 7
Results Summary	pg. 7
Severity 5 (Critical) Findings	pg. 8
Finding 1 Raw API Exposed	pg. 8
Finding 2 Password Hashes Exposed	pg. 11
Finding 3 Weak Password Hashing Algorithm	pg. 12
Severity 4 (High) Findings	pg. 13
Finding 4 Cleartext Authentication	pg. 13
Finding 5 Cross-Site Scripting	pg. 15
Finding 6 Auth Cookie Missing Http Only Flag	pg. 16
Severity 2 (Low) Findings	pg. 17
Finding 7 Debug Console Enabled	pg. 17
Vulnerability Classifications	pg. 18

Executive Summary

Between 6/3/19 and 6/7/19, Brian Milliron conducted a security assessment of the Conglomo web application. Several serious vulnerabilities were identified which could compromise the confidentiality, availability, and integrity of the application, as well as user accounts and the personal data of users.

Summary of Findings

Finding 1:	Raw API Exposed
Severity Level:	5
Disposition:	Open
Impact to Business:	Non-privileged users can perform functions which should be restricted to admins

Finding 2:	Password Hashes Exposed
Severity Level:	5
Disposition:	Open
Impact to Business:	Password hashes are exposed to all users

Finding 3:	Weak Password Hashing Algorithm
Severity Level:	5
Disposition:	Open
Impact to Business:	Weak password hashes enable recovery of original password and unauthorized access

Finding 4:	Cleartext Authentication
Severity Level:	4
Disposition:	Open
Impact to Business:	Allows an attacker on the same local network to capture passwords

SAMPLE Security Assessment Report

Finding 5:	Cross-Site Scripting
Severity Level:	4
Disposition:	Open
Impact to Business:	Allows an attacker to inject malicious code into the session of another user

Finding 6:	Auth Cookie Missing Http Only Flag
Severity Level:	4
Disposition:	Open
Impact to Business:	Allows an attacker to steal the authentication cookie and login session of another user

Finding 7:	Debug Console Enabled
Severity Level:	2
Disposition:	Open
Impact to Business:	Aids an attacker in gaining unauthorized access.

Vulnerability Severity Levels

	5	4	3	2	1
Number of Findings	3	3	0	1	0

SAMPLE Security Assessment Report

Objective

The objective of the security assessment is to provide an assessment of the security posture of the Conglomo web application. This report helps by gauging issues found during the assessment against industry standards, corporate policy, and the knowledge of the assessors.

Assessment Scope

The security assessment was focused on the Conglomo web application hosted on the server at X.X.X.X. No testing was done on the server itself or supporting infrastructure. The results from this test are not intended to be an assessment of all applications, or entire infrastructure, and pertain only of those targets identified within this assessment's scope. While changes to the infrastructure, application code, configurations and architectures may always be in progress, the assessment provided in this report only presents those issues which existed during the assessment period. Findings listed in this report are a snapshot of the issues discovered, which existed during the assessment period, and may not be current. Findings discussed in this document are representative of issues in general and may not list all instances of a specific issue. The assessment also did not perform any denial of service (DoS) attacks against the network, its subsystems, devices or applications in order to minimize the potential of interrupting operations.

Assessment Tools

A variety of automated and manual tools are used to increase the thoroughness of the analysis as well as to increase efficiency and promote the re-usability and standardization of components. The following list of tools are the most common that are used, but may not be all inclusive.

- Standard web browser
- Burp web proxy

SAMPLE Security Assessment Report

Target Systems

This Assessment was conducted in the following environments:

- Test

The following IP Address(es) and/or URL's were assessed:

- http:// X.X.X.X

Security Assessment Results

While some effort has been made to restrict the application functionality to only authorized users, the security controls currently in place are not sufficient to protect the application data and functionality. Several vulnerabilities allow unprivileged users to gain unauthorized access to privileged accounts and/or bypass access controls. In many cases these vulnerabilities can be chained together to increase the impact and level of exposure.

Findings 1,2 and 3 are part of a vulnerability cluster where each vulnerability increases the severity and impact of the others. Likewise findings 5 and 6 are also part of a vulnerability cluster. Remediating any single vulnerability which is part of a cluster reduces the severity and impact of the others.

SAMPLE Security Assessment Report

Severity 5 (Critical) Findings

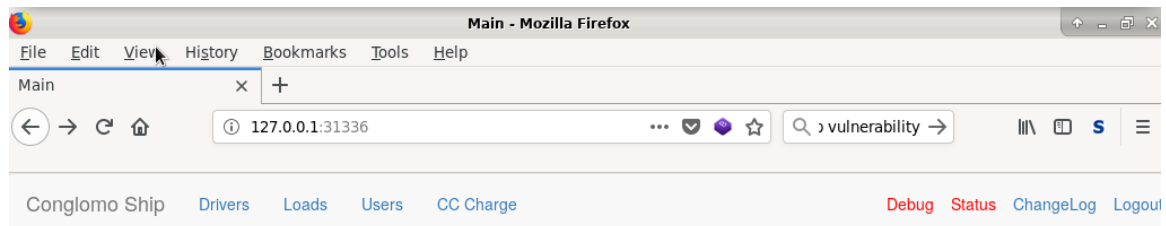
Finding 1: Raw API Exposed

Asset(s) Affected:

<http://X.X.X.X/api/raw/>

Issue: The raw API is exposed to unprivileged users

Description: The API has no access control to prevent access by unauthorized users. It should not be accessible directly from the web interface and should only be accessed by the application itself after authentication checks have been performed. It has in fact been disabled in the regular configuration for this reason, but it can easily be re-enabled by sending a GET request to the `/api/enableddebug` url, at which point the Debug menu option shows up on the navigation menu and the `/api/raw` url becomes available.

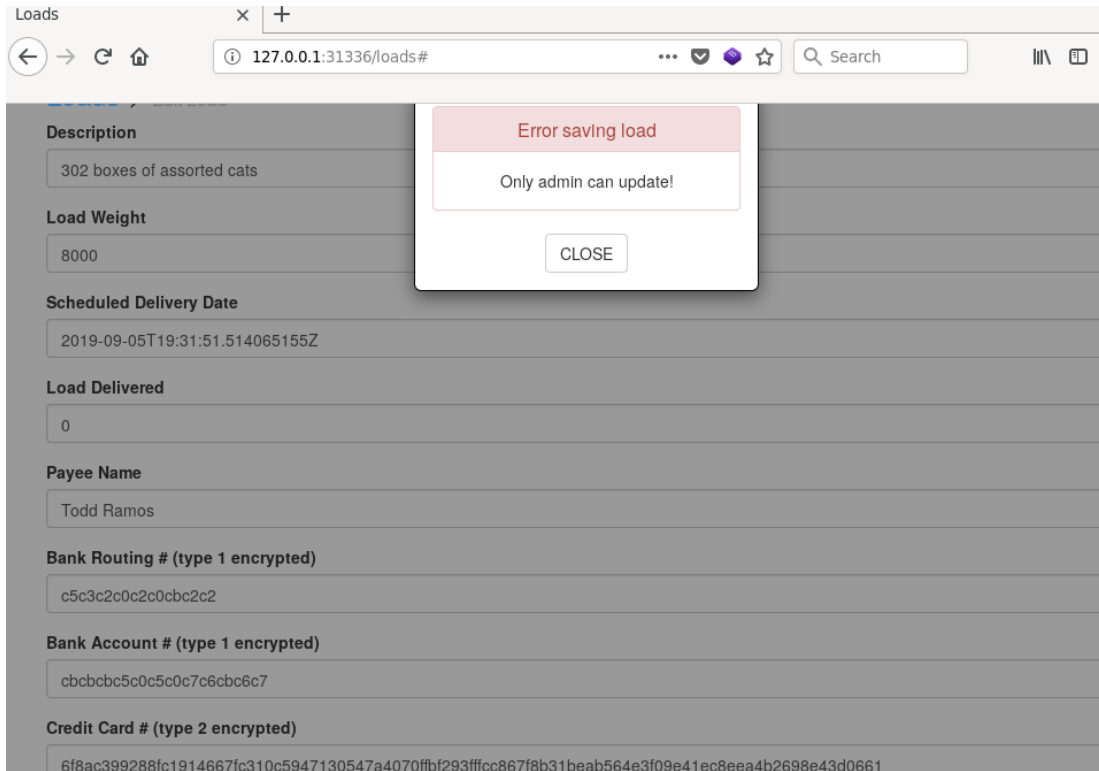


Welcome to Conglomo Ship

Choose from an activity above.

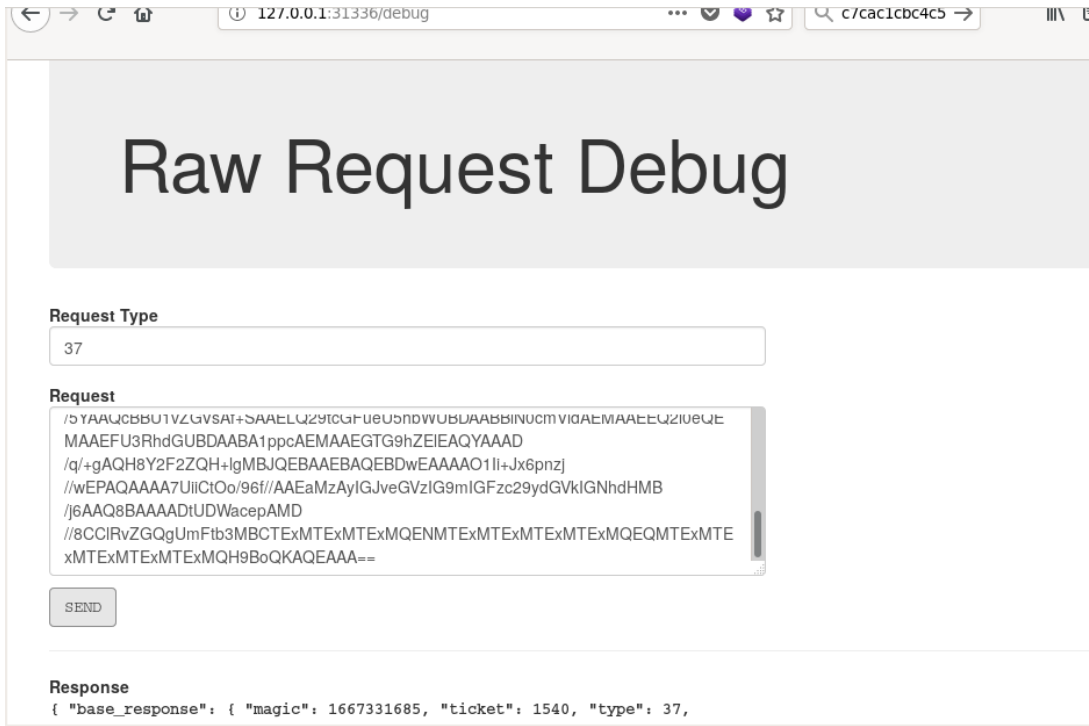
SAMPLE Security Assessment Report

The API has no authorization controls and will take actions based on any well formed request. Users can easily generate well formed requests by attempting to perform a prohibited action and saving the base64 encoded request, then pasting it into the “Request” form field of the API Debug interface. The API will then perform the prohibited action on behalf of the user.



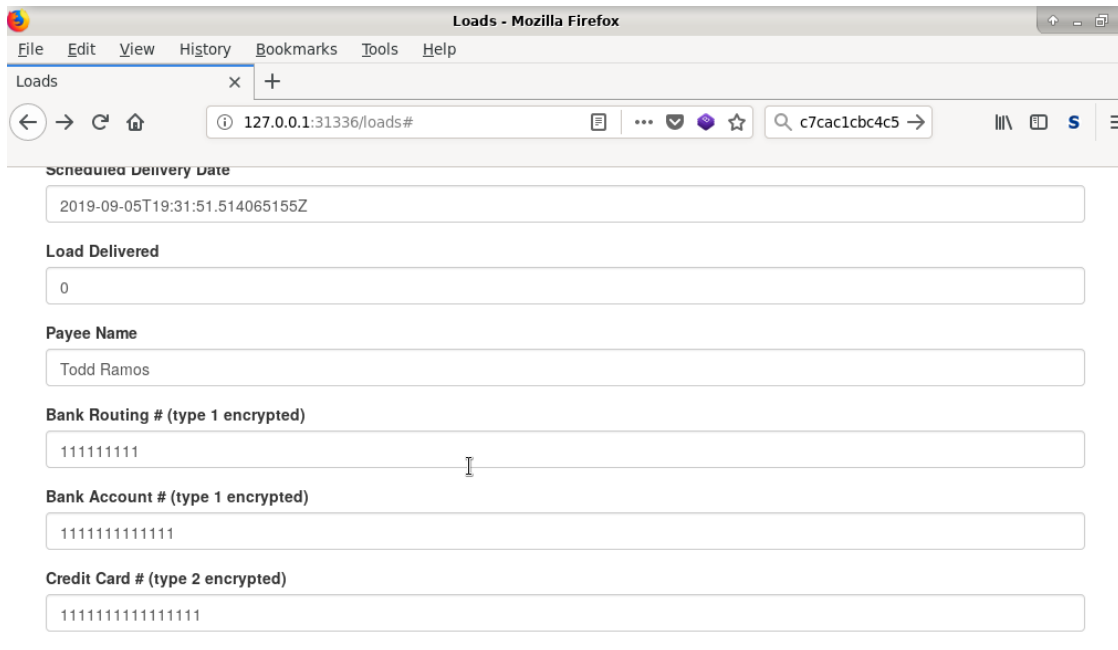
This screenshot shows the user attempting to update the payment method details, which is a prohibited action.

SAMPLE Security Assessment Report



This screenshot shows the user inputting the payment change request into the API Debug page.

This screenshot shows the final result.



SAMPLE Security Assessment Report

Recommendations: Remove the ability for the user to access the raw API.

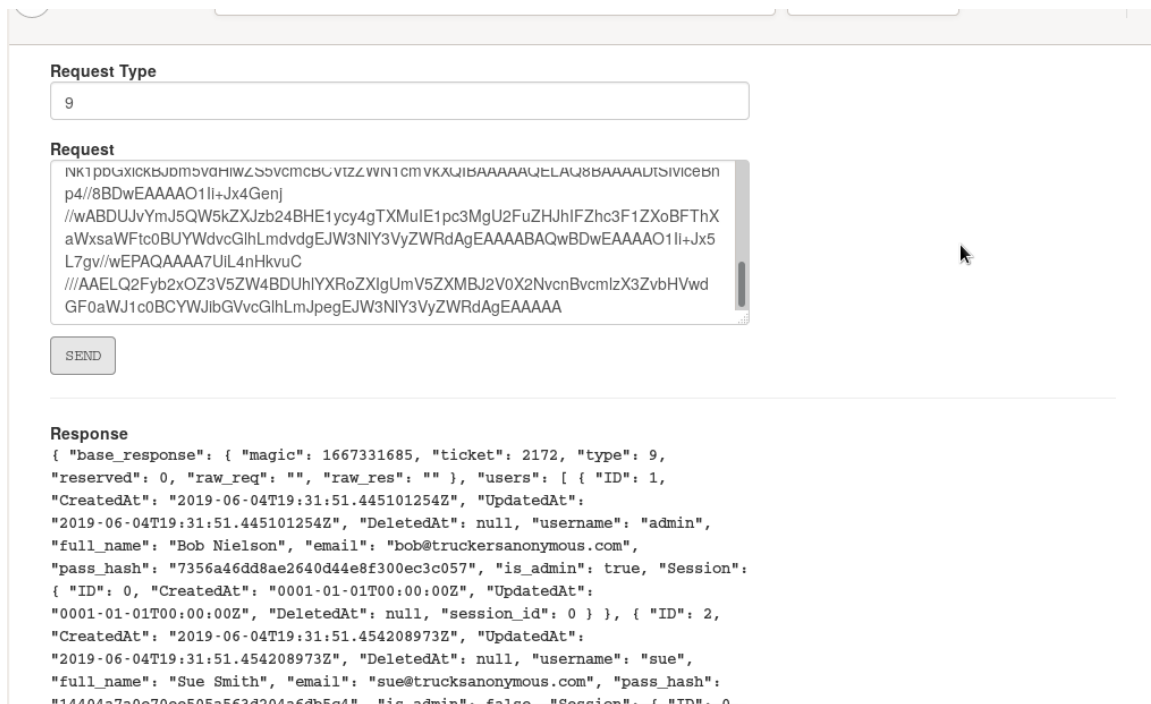
Finding 2: Password Hashes Exposed

Asset(s) Affected:

http:// X.X.X.X/api/raw

Issue: The Raw API exposes user password hashes

Description: User password hashes are very sensitive information which some attempt has been made to protect by redacting them from showing up in the Users page. However the Raw API exposes them in full unredacted form. This can enable a malicious user to “crack” the hash and reveal the original password which can then be used to login as any user, including the administrator.



Request Type

9

Request

```
NK1pDcXICKBJDm5vQHIWZ55vcmCBcVIZzVWN1cmVKKWIBAAAAAQELAQ8BAAAAU5IVICEBn
p4//8BDwEAAAAO1ll+Jx4Genj
//wABDUJvYmJ5QW5kZXJzb24BHE1ycy4gTXMuE1pc3MgU2FuZHZhIFZhc3F1ZXoBFTxX
aWxsaWFtOBUYWdvcGhlLmdvdGElJW3NIY3VyZWRRdAgEAAAABAQwBDwEAAAAO1ll+Jx5
L7gv//wEPAQAAAA7Uil4nHkvuC
///AAELQ2Fyb2xvZ3V5ZW4BDUhlYXRoZXIglGUmV5ZXMBJ2V0X2NvcnBvcmlzX3ZvbHVwd
GF0aWJ1c0BCYWFibGVvcGhlLmJpegElJW3NIY3VyZWRRdAgEAAAAA
```

Response

```
{ "base_response": { "magic": 1667331685, "ticket": 2172, "type": 9,
"reserved": 0, "raw_req": "", "raw_res": "" }, "users": [ { "ID": 1,
"CreatedAt": "2019-06-04T19:31:51.445101254Z", "UpdatedAt":
"2019-06-04T19:31:51.445101254Z", "DeletedAt": null, "username": "admin",
"full_name": "Bob Nielson", "email": "bob@truckersanonymous.com",
"pass_hash": "7356a46dd8ae2640d44e8f300ec3c057", "is_admin": true, "Session":
{ "ID": 0, "CreatedAt": "0001-01-01T00:00:00Z", "UpdatedAt":
"0001-01-01T00:00:00Z", "DeletedAt": null, "session_id": 0 } }, { "ID": 2,
"CreatedAt": "2019-06-04T19:31:51.454208973Z", "UpdatedAt":
"2019-06-04T19:31:51.454208973Z", "DeletedAt": null, "username": "sue",
"full_name": "Sue Smith", "email": "sue@trucksanonymous.com", "pass_hash":
"14404a7a0e70ee505a563d204a64b5c4", "is_admin": false, "Session": { "ID": 0
```

Recommendations: Remove the ability for users to view raw password hashes.

SAMPLE Security Assessment Report

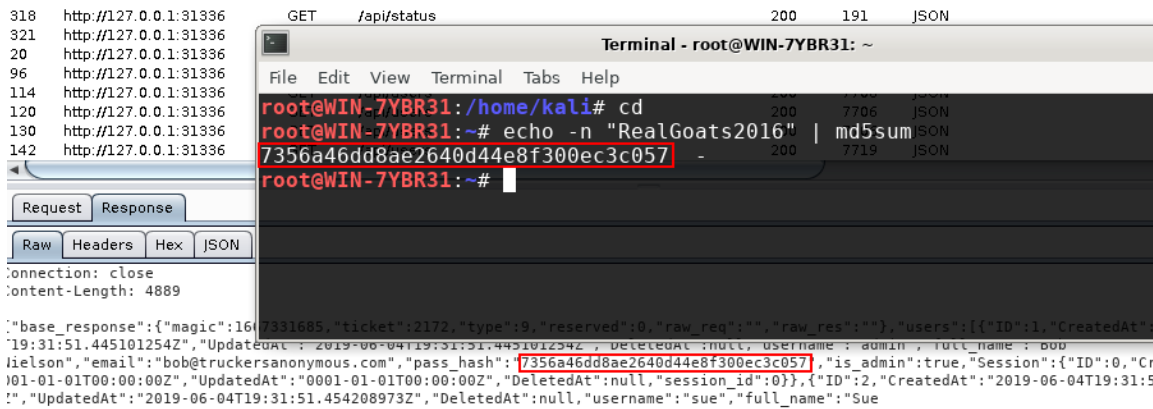
Finding 3: Weak Password Hashing Algorithm

Asset(s) Affected:

http://X.X.X.X/

Issue: User passwords are hashed using the weak MD5 hashing algorithm

Description: Passwords are hashed to provide additional protection in the event that the application is compromised by forcing the attacker to undertake the additional step of cracking the hashes prior to being able to use them to login as another user. The stronger the hashing algorithm, the longer it takes to crack the hashes. The MD5 hashing algorithm is one of the weakest. Additionally the hash function does not use a salt, which makes it even easier for an attacker who has access to MD5 rainbow tables. Practically speaking, an attacker who gains access to these password hashes can crack them in under an hour and quickly turn them into usable passwords for further attacks on the application.



Recommendations: Use a strong hashing algorithm such as scrypt or bcrypt to resist cracking attempts. Additionally use a unique salt prior to hashing so that an attacker will be unable to use pre-computed rainbow tables to crack the hash.

References:

https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Password_Storage_Cheat_Sheet.md

SAMPLE Security Assessment Report

Severity 4 (High) Findings

Finding 4: Cleartext Authentication

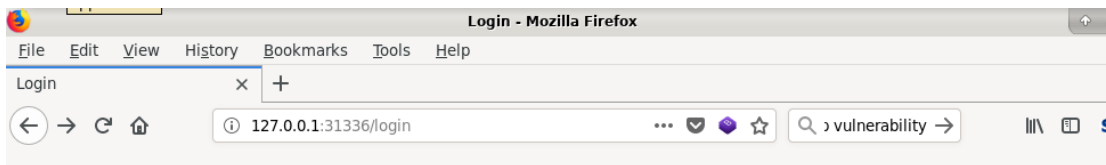
Asset(s) Affected:

http://X.X.X.X/login

Issue: The web application allows login credentials to be transmitted over cleartext

Description: The application allows users to connect to it over unencrypted connections. An attacker suitably positioned to view a legitimate user's network traffic could record and monitor the user login credentials in order to impersonate the user or gain unauthorized access to resources. Furthermore, an attacker able to modify traffic could use the application as a platform for attacks against its users and third-party websites.

To exploit this vulnerability, an attacker must be suitably positioned to eavesdrop on the victim's network traffic. This scenario typically occurs when a client communicates with the server over an insecure connection such as public Wi-Fi, or a corporate or home network that is shared with a compromised computer. Common defenses such as switched networks are not sufficient to prevent this.



Conglomo Ship Login

Username

Password

SAMPLE Security Assessment Report

```
POST /login HTTP/1.1
Host: 127.0.0.1:31336
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:60.0) Gecko/20100101 Firefox/60.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Referer: http://127.0.0.1:31336/login
Content-Type: application/x-www-form-urlencoded
Content-Length: 27
Cookie: capp=MTU1OTc1ODQ5NnxEdi1CQkFF0180SUFBUkFCRUFBUJQLUNBQUE9fKwcpXpt3Qmi7BNBBVMunNpp7awJAwzAzS-qIu6fdLSI
DNT: 1
Connection: close
Upgrade-Insecure-Requests: 1

username=test&password=test
```

Recommendations: Applications should use transport-level encryption (SSL/TLS) to protect all communications passing between the client and the server. The Strict-Transport-Security HTTP header should be used to ensure that clients refuse to access the server over an insecure connection.

References:

https://en.wikipedia.org/wiki/Transport_Layer_Security

Finding 5: Cross-Site Scripting

Asset(s) Affected:

http://X.X.X.X/users

Issue: The web application uses bootstrap v3.3.7 which is known to have cross-site scripting (XSS) vulnerabilities in the data-target, data-template, data-content, data-title, and data-viewport attributes

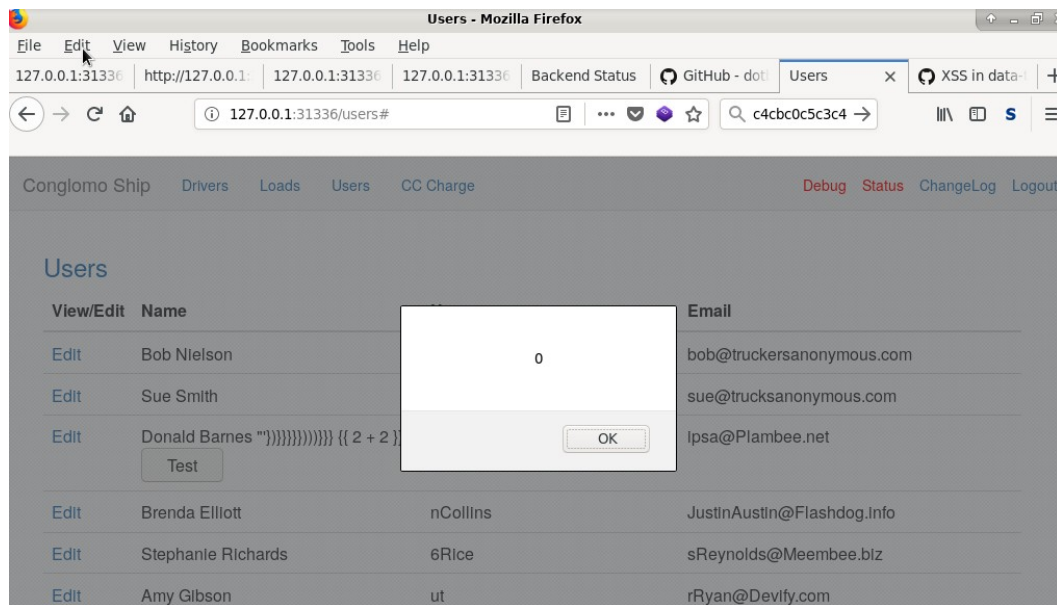
Description: The Conglomo web application is vulnerable to a stored XSS attack where the attacker can be any user with write access to any form field. In this example the users table containing contact and other personal information for users was poisoned with a malicious javascript. Even though html tags are being properly escaped, javascript can still be saved in the table and will run whenever that user data is viewed by anyone.

XSS allows attackers to inject malicious code into an otherwise benign website. These scripts acquire the permissions of scripts generated by the target website and can therefore compromise the confidentiality and integrity of data transfers

SAMPLE Security Assessment Report

between the website and client. Websites are vulnerable if they display user supplied data from requests or forms without sanitizing the data so that it is not executable. Stored attacks are those where the injected script is permanently stored on the target servers, such as in a database, in a message forum, visitor log, comment field, etc.

The attacker-supplied code can perform a wide variety of actions, such as stealing the victim's session token or login credentials, performing arbitrary actions on the victim's behalf, and logging their keystrokes.



Recommendations: Upgrade to the latest version of bootstrap

References:

- [https://www.owasp.org/index.php/Cross-site_Scripting_\(XSS\)](https://www.owasp.org/index.php/Cross-site_Scripting_(XSS))
- https://github.com/OWASP/CheatSheetSeries/blob/master/cheatsheets/Cross_Site_Scripting_Prevention_Cheat_Sheet.md
- <https://snyk.io/test/npm/bootstrap/3.3.7>

SAMPLE Security Assessment Report

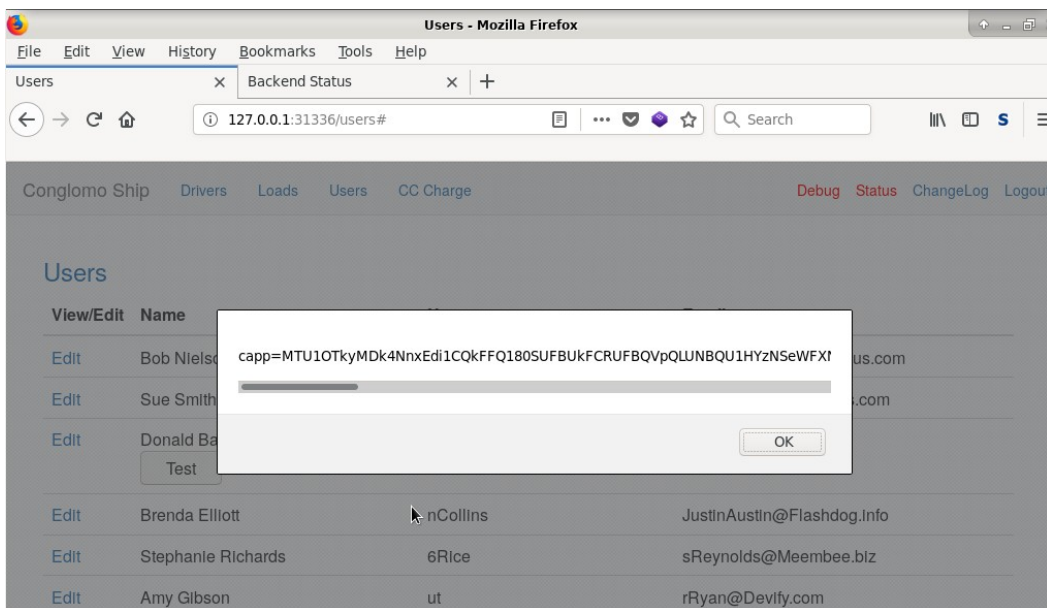
Finding 6: Auth Cookie Missing Http Only Flag

Asset(s) Affected:

http://X.X.X.X/

Issue: The capp cookie does not have the Http Only flag set

Description: The web application uses the capp cookie to control the session state. Because it doesn't have the Http-Only flag, this means a malicious javascript can read the cookie value. This greatly increases the severity of the previous XSS vulnerability because now an attacker can copy the session cookie of a logged in user, including the administrator, and use it in his own session, gaining access to the compromised account.



Recommendations: Set the Http Only flag on the capp cookie. If the HttpOnly attribute is set on a cookie, then the cookie's value cannot be read or set by client-side JavaScript. This measure makes certain client-side attacks, such as cross-site scripting, harder to exploit by preventing them from trivially capturing the cookie's value via an injected script.

References:

<https://www.owasp.org/index.php/HttpOnly>

SAMPLE Security Assessment Report

Severity 2 (Low) Findings

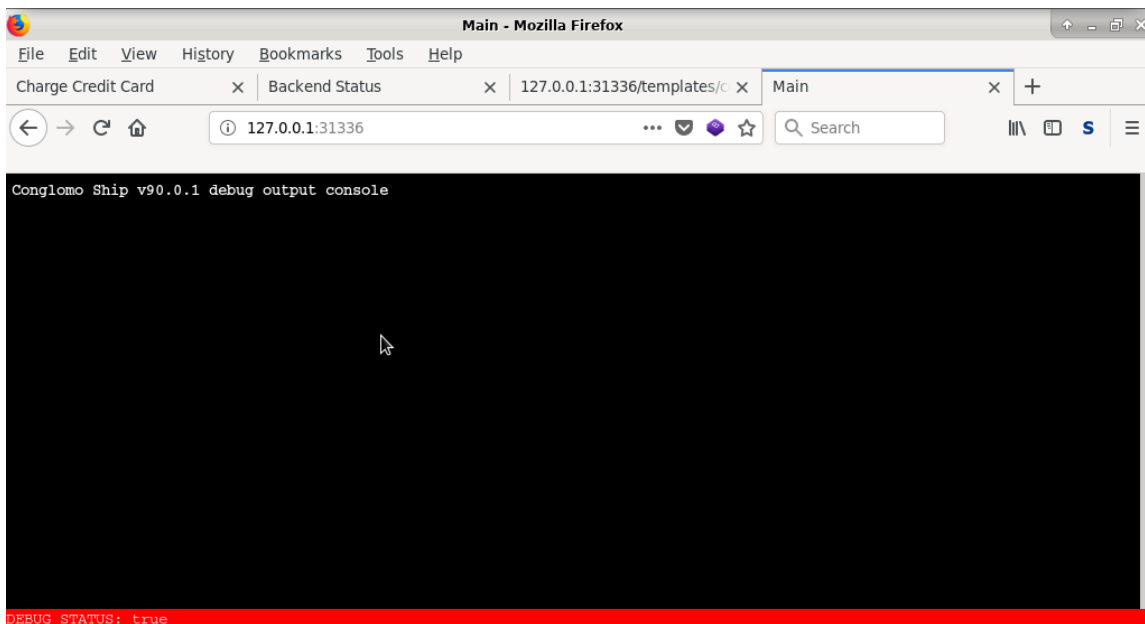
Finding 7: Debug Console Enabled

Asset(s) Affected:

<http://X.X.X.X/>

Issue: The debug console is enabled and accessible to all users

Description: The debug console displays detailed information about the web application which would be useful in tailoring an attack. It can be accessed by any user simply by pressing the ~ key. It will show the api requests and responses, which can be fed into the Raw API to facilitate attacks.



Recommendations: Disable debug console

Vulnerability Classifications

Table Vulnerability Severity Scoring

Severity of Issue	Severity Level	Criteria	Mitigation Plan Date	Mitigate by Date
Critical	5	Serious and immediate threat to enterprise; confidentiality, integrity or availability of a critical resource could be compromised	n/a	Mitigation should commence immediately
High	4	Serious threat to application or critical resource	optional	0 – 4 weeks
Medium	3	Moderate threat to application or critical resource	2 weeks	0 – 8 weeks
Low	2	Minor threat to application or critical resource	4 weeks	4 – 24 weeks
Informational	1	General security information	n/a	n/a