Task4 Ans

Overview 1

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, don't go begging to someone else for the answer, thats one way to get you hated/made fun of. Enter the password and you can continue.

Solution

If we look at the source code of the page there is a comment saying
`<!-- the first few levels are extremely easy: password is a13fe647 -->`
hence upon checking this we find that **a13fe647** is the password indeed.

Overview 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...

Solution

The script matches our password with unencrypted text file of password that sam uploads. But since Sam didn't upload the file hence the script would match our password with empty value.So if we don't enter password and submit so our password also becomes empty hence it matches.

So submit without Entering anything

Overview 3

This time Network Security Sam remembered to upload the password file, but there were deeper problems than that.

Solution

Upon Inspecting the Page's source code we see that the password file is mentioned in the form of hidden input field which as seen below

`<input type="hidden" name="file" value="password.php">`

So upon entering password.php in url i got
990514aa which turns out to be teh password.

Overview 4

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:

Solution

We see that there is a hidden input field in the form where email of SAM is mentioned .

We can assume that the reminder is being sent to this mail address so if we change it to our own address then we get the mail with the passworde which is
Sam,
Here is the password: 7b92d260

Overview 5

Sam has gotten wise to all the people who wrote their own forms to get the password. Rather than actually learn the password, he decided to make his email program a little more secure.

Solution

Here Also as the email program was made more secure but the hidden input still remains so the method of above level also works here.
Password is
Sam,
Here is the password: 'e4643a2d'.

Overview 6

Network Security Sam has encrypted his password. The encryption system is publically available and can be accessed with this form:

Solution

We have a input and button which takes a string and gives us its encrypted version by applysing the same aencryption algorith as of the password.
We also have this mentioned : **You have recovered his encrypted password. It is: b27d676<**

Now on giving abcd and 1234 respectively and pressing encrypt we got aceg and 1357 respectively. Also trying out other strings we can conclude that the encryption algorithm takes the string and adds (n to the nth index of the string assuming 0 based indexing in thier ASCII numeric value and then show the corresponding character to that number)

so by reversing this algo we find the password is :
b15a2205

Overview 7

his 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:

Solution

Here we see that UNIX command cal is being used so we can try to add even mmore command assuming its using cal < year > we could use **cal < year > ; ls** .
And we see that it works and we see a directory named : **k1kh31b1n55h.php** .On navigating to this directory liek
`https://www.hackthissite.org/missions/basic/7/k1kh31b1n55h.php`
we get the password as : **3624eafe**

Overview 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.

Solution

Upon researching I found out that this is vulnerable to SSI(Server Side Includes) Injection.
Since Sam's daughter did't think much about security so SSI is being used here as it feed an HTML page with dynamic contents.
Now we can easily find its exploit on the internet and the ones i used were :

- `<!--#exec cmd="ls ../" -->` to list the directories and files and i found (au12ha39vc.php index.php level8.php tmp) files out o which au12ha39vc.php seems interesting so i tried to see it.
- `<!--#exec cmd="cat ../au12ha39vc.php" -->` from this i saw the content and got the password.
  The file contained **Hi, 27d6aed9! Your name contains 8 characters.**
  So Password is **27d6aed9**

Overview 9

Network Security Sam is going down with the ship - he's determined to keep obscuring the password file, no matter how many times people manage to recover it. This time the file is saved in /var/www/hackthissite.org/html/missions/basic/9/.

In the last level, however, in my attempt to limit people to using server side includes to display the directory listing to level 8 only, I have mistakenly screwed up somewhere.. there is a way to get the obscured level 9 password. See if you can figure out how...

This level seems a lot trickier then it actually is, and it helps to have an understanding of how the script validates the user's input. The script finds the first occurrence of '<--', and looks to see what follows directly after it.

Solution

We see that vulnerability from Mission 8 is still there but there is no input field to test that so we upon thinking i found out that since it mention * and its vulnerability is still in play so why don't we try accessing the password file for 9 from 8 itself and i did that using the command

- `<!--#exec cmd="ls ../../9/" -->` from this i saw files named index.php p91e283zc3.php and **p91e283zc3.php** seemed valuable so i tried to see its content
- for viewing p91e283zc3.php i used `<!--#exec cmd="cat ../../9/p91e283zc3.php" -->` and i got the password.
  The file contained Hi, **a9e3428d**! Your name contains 8 characters.
  So Password is **a9e3428d**

Overview 10

This time Sam used a more temporary and "hidden" approach to authenticating users, but he didn't think about whether or not those users knew their way around javascript...

Solution

As JavaScript is mentioned we enter the password and see the response it says unaunthenticated. On Inspecting the page i found out that in their are 2 cookies (can be sen from Inspect->Application -> Cookies)

- HackThisSite
- level10_authorized
  and value of **level10_authorized is no so if we change it to yes** then enter password again or reload for unauthenticated page we see that we are logged in.
  So change cookie **level10_authorized from no to yes**

Overview 11

Sam decided to make a music site. Unfortunately he does not understand Apache. This mission is a bit harder than the other basics.

Solution

On starting we see a ginglewhite page with only below line written in it
I love my music! "Ego " is the best!

If we reload then we notice that the song name has changed and we get
I love my music! "From Denver to L.A." is the best!

As we keep reloading the song changes and upon looking up the song names on internet we see that these are Elton John's Song

No Since its given that Sam don't understand apache so its safe to assume he did not disable directory listing which is enabled by default.
So if we do directory busting we see **index.php** and nothing else from common wordlist.
index.php is the page where we have to enter password we look further.
Now if wee see elton , eltonjohn ,j ohn , e, j as directory we see that **e** is indeed a directory and from there on we see directory visible till
`https://www.hackthissite.org/missions/basic/11/e/l/t/o/n/`
Now we see no directory visible but it is still possible that hidden directories are there and one of the most common hidden directory in apache is **.htaccess** and on entering this in url we get

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

So on going to DaAnswer directory we get
The answer is **short**! Just look a little harder.

from the below link
`https://www.hackthissite.org/missions/basic/11/e/l/t/o/n/DaAnswer`

Now the upon looking harder as it says we find nothing and in the end it turns out that Bold Word(done by me) **short** is the answer as it literally says **answer is short**.
Note this answer keeps changing depending on how much we reload it.

Now going to index.php and entering the password short we are redirected to page where we have a button that says **Go On**
On pressing it we have finally cleared the level.