## Tests

| **Test #** | **Test procedure / description** | **Expected Outcome** | **Actual Outcome & Remarks** | **Pass / Fail** |
| --- | --- | --- | --- | --- |
| 1 | 1. on a client machine, SSH into the server machine 2. input an invalid password 3 or more times 3. check the security logs | * entries in the security log should show that there were password failed attempts | As expected | Pass |
| 2 | 1. enter "crontab -r" into a terminal 2. use the setjob.sh script to add the ips script as a cronjob with max attempts set to 3 3. check the crontab file | * the crontab file should have an entry in it for regularly running the IPS script | As expected | Pass |
| 3 | 1. enter "crontab -r" into a terminal 2. use the setjob.sh script to add the ips script as a cronjob with max attempts set to 3 3. try to log in 3 or more times from a client with invalid passwords 4. await the ips' execution 5. check the database, and iptables | * In iptables, it should have an entry to drop all packets from the client's IP address. * The database should have an entry with the client's IP address, number of failed attempts, and timestamp of the last attempt. | As expected | Pass |
| 4 | 1. enter "crontab -r" into a terminal 2. use the setjob.sh script to add the ips script as a cronjob with max attempts set to 3 3. try to log in 3 or more times from a client with invalid passwords 4. await the ips' execution 5. attempt to connect via SSh to the server again | * the SSH client should hang as it is trying to connect to the remote host | As expected | Pass |
| 5 | 1. enter "crontab -r" into a terminal 2. use the setjob.sh script to add the ips script as a cronjob with max attempts set to 3 3. try to log in 3 or more times from a client with invalid passwords 4. await the ips' execution 5. wait for the user-specified time to elapse for unbanning an ip address 6. check the database, and iptables | * the database file should no longer have an entry with the client's IP address * in iptables, there should be the original rule dropping all packets from the client's IP address, but it should be preempted by a new rule that accepts all packets from that IP address. | As expected | Pass |
| 6 | 1. enter "crontab -r" into a terminal 2. use the setjob.sh script to add the ips script as a cronjob with max attempts set to 3 3. try to log in 3 or more times from a client with invalid passwords 4. await the ips' execution 5. wait for the user-specified time to elapse for unbanning an ip address 6. attempt to connect via SSh to the server again | * the SSH client should successfully connect, and prompt the user for a password | As expected | Pass |
| 7 | 1. enter "crontab -r" into a terminal 2. use the setjob.sh script to add the ips script as a cronjob 3. try to log in 2 or times from a client with invalid passwords 4. await the ips' execution 5. check the database file and security logs | * the security logs should show some password failure attempts * the database should have an entry for the client with its IP address, 2 for attempted logins, and a timestamp of its last login * the SSH client shows that it has failed to login twice * there should be no rules in the iptables for the client | As expected | Pass |
| 8 | 1. enter "crontab -r" into a terminal 2. use the setjob.sh script to add the ips script as a cronjob 3. try to log in 2 or times from a client with invalid passwords 4. await the ips' execution 5. check the database file and security logs 6. log in successfully with the correct password 7. await the ips' execution 8. check the database file | * the security logs should show some password failure attempts as well as a successful login attempt * the database should have no entry for the client * the SSH client should connect * there should be no rules in the iptables for the client | As expected | Pass |

## Note

the tests here show only the IPS working when monitoring the ubuntu auth.log, however it has proven to work on Fedora 22's /var/log/messages as well as its /var/log/secure.

## Screenshots

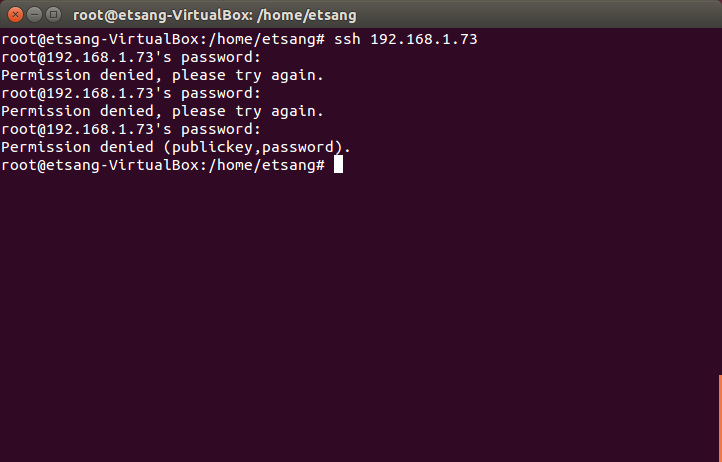


Figure Test 1, client is SSHing into the server with invalid passwords 3 times

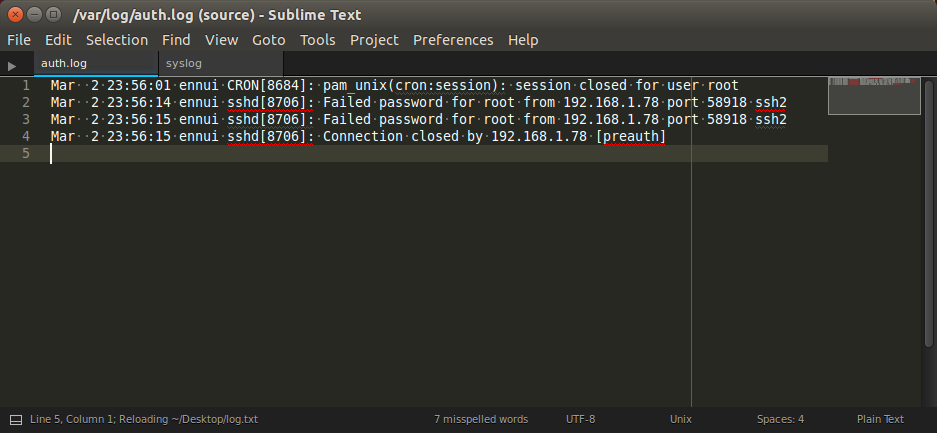


Figure Test 1, security logs show that there are failed password attempts

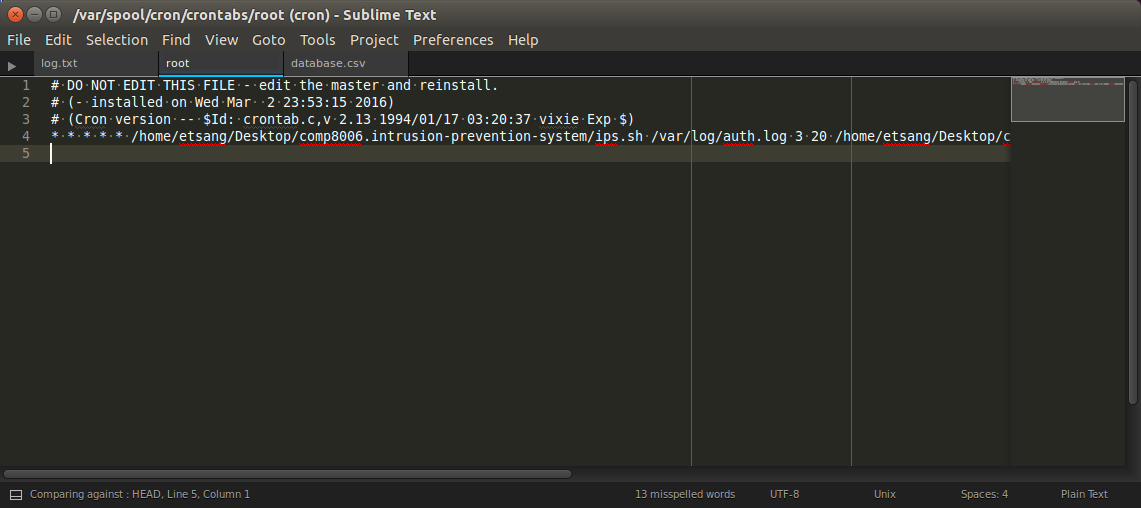


Figure Test 2, the crontab file for root has an entry in it for regularly running the IPS script

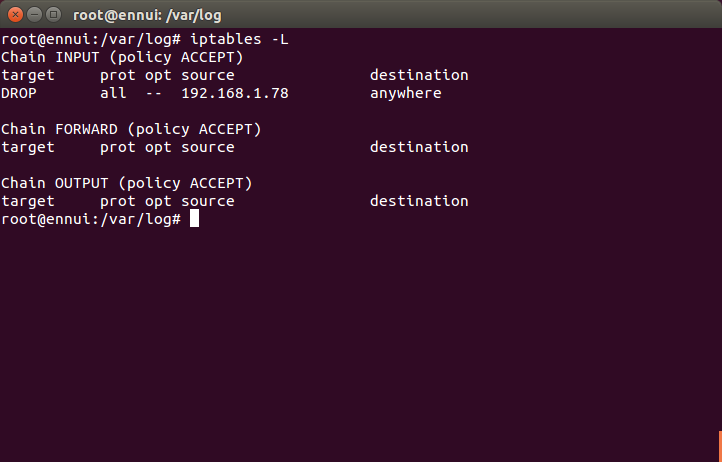


Figure Test 3, iptables shows that a new rule was appended to it, banning all traffic from the client IP address

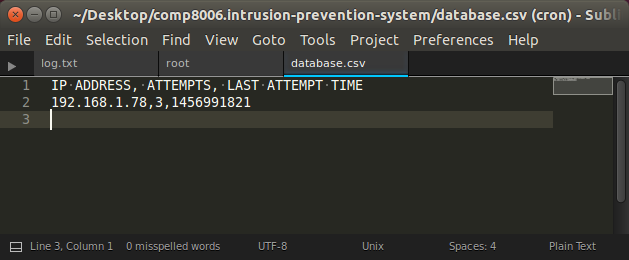


Figure Test 3, the database file shows the IP address of the client that failed to log in, how many password attempts there were, and a timestamp of their last attempt

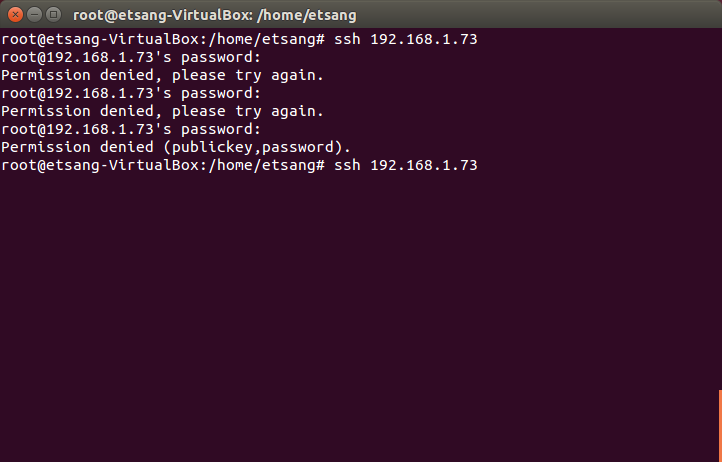


Figure Test 4, SSH client hanging while trying to connect to the server, because the server has banned its IP address

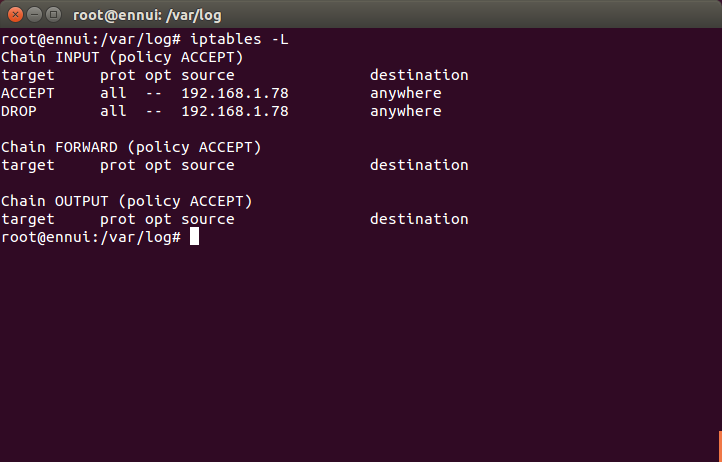


Figure Test 5, the iptables drop all packets from malicious client rule has been preempted with a rule that accepts all traffic from the client

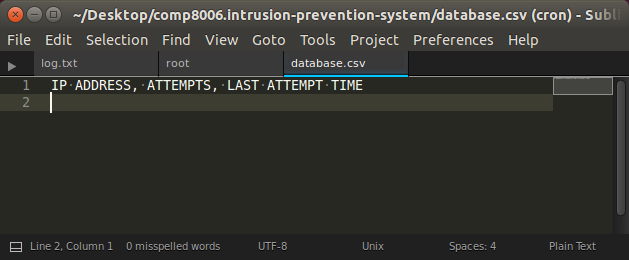


Figure Test 5, there is no more entry for banning the previously banned client because the ban time has elapsed

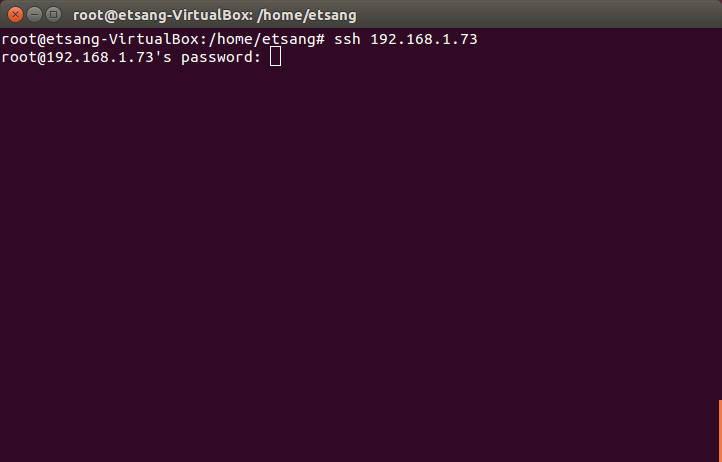


Figure Test 6, the SSH client can now connect to the server after being unbanned

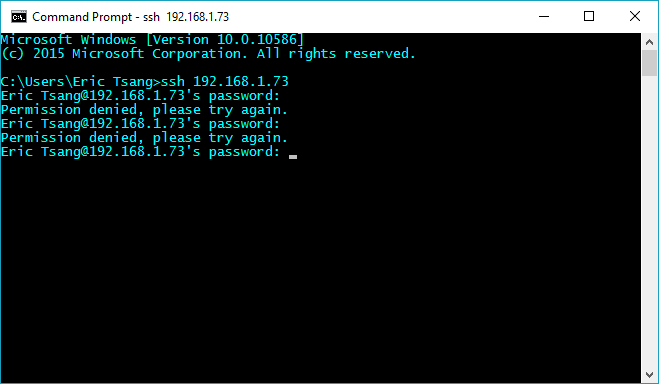


Figure Test 7, SSH client fails to input valid password twice

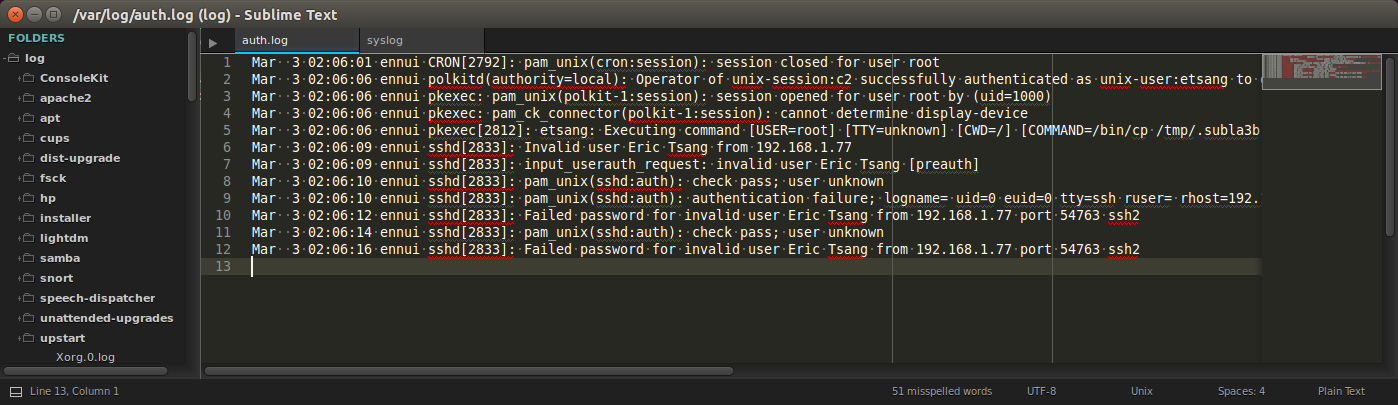


Figure Test 7, authentication logs from the server showing 2 invalid login attempts

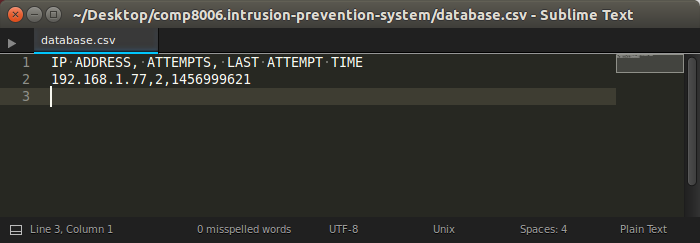


Figure Test 7, ips notes that the client has tried to log on twice, but failed

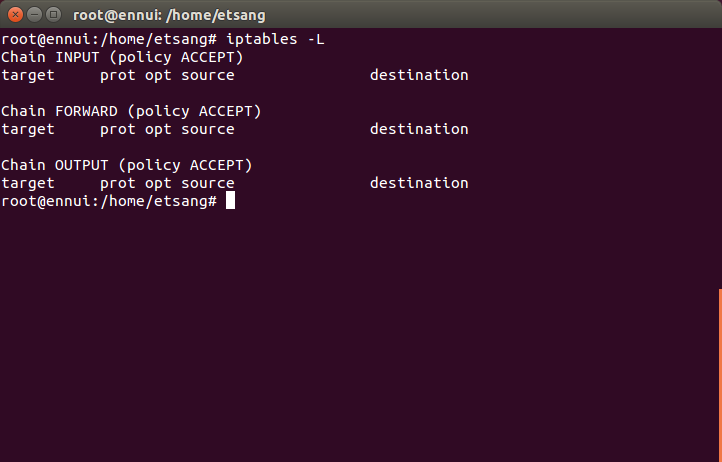


Figure Test 7, no iptables entry is added to ban the client yet, because 2 attempts is still below the threshold

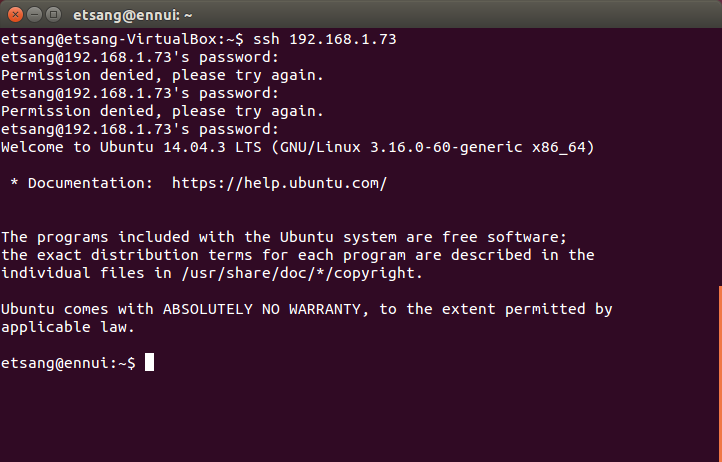


Figure Test 8, SSH client logs in successfully after 2 invalid passwords

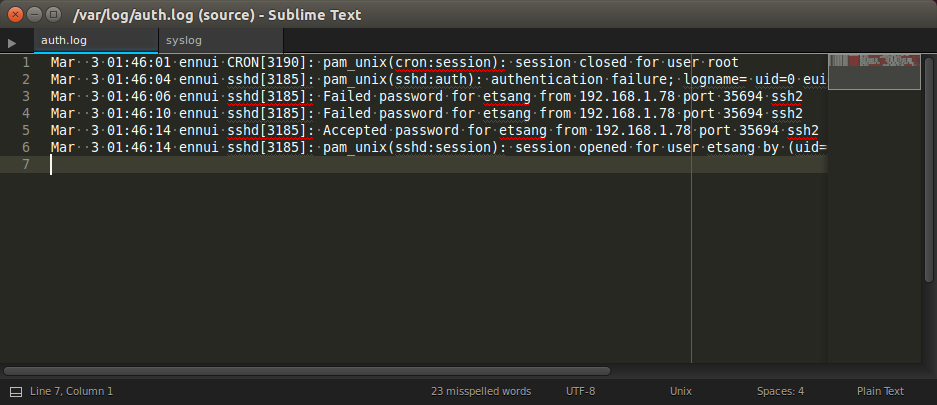


Figure Test 8, security logs show that there were two failed login attempts followed by a successful one

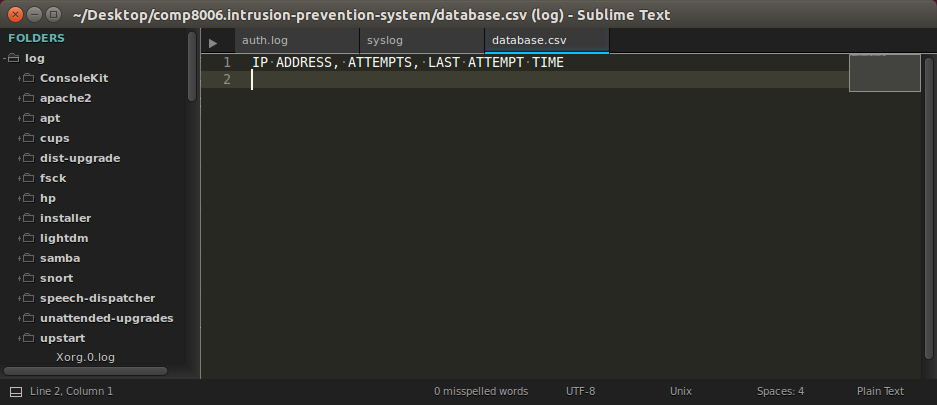


Figure Test 8, there is no recorded in the database for the client because it logged in successfully

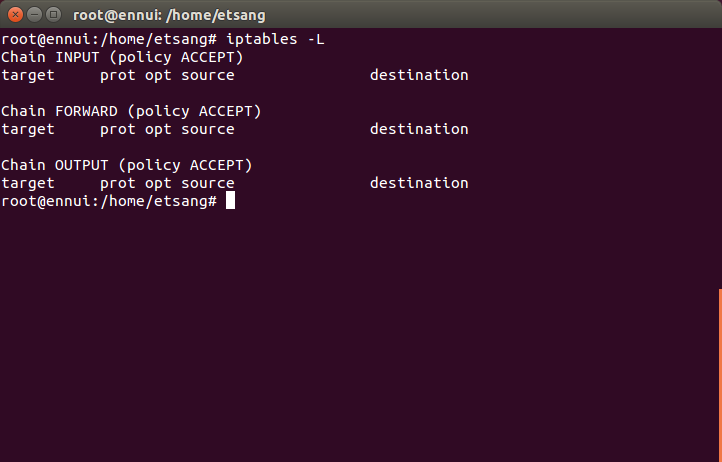


Figure Test 8, there is no entry in the iptables for the client because it is still below the threshold for being banned