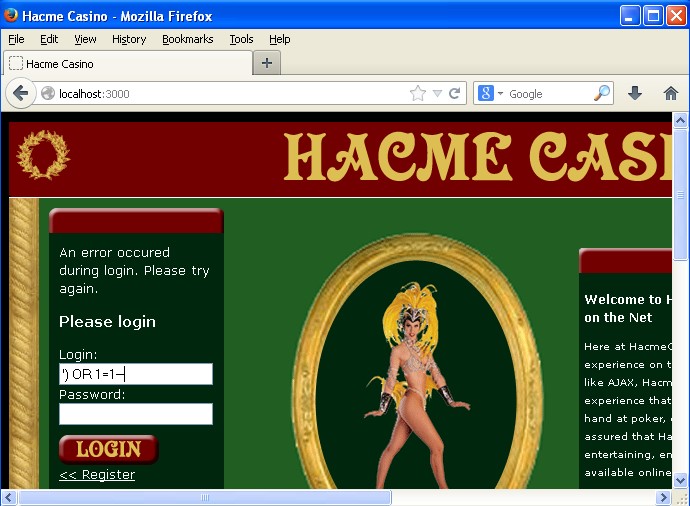
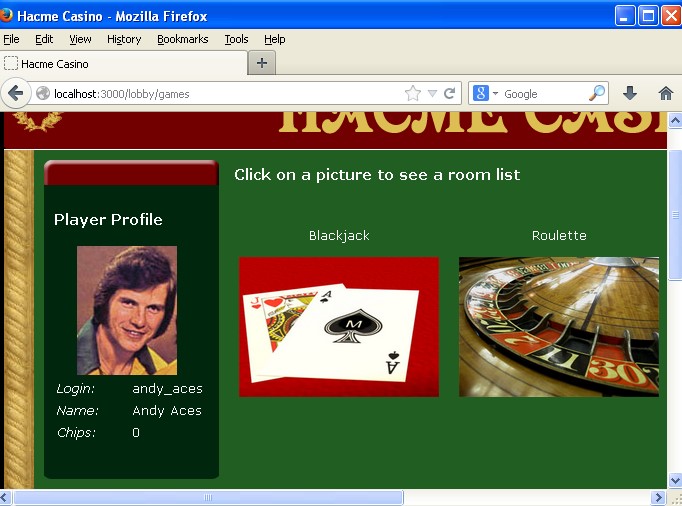
Alec Austin - J00501671

Secure Software

19 October 2017

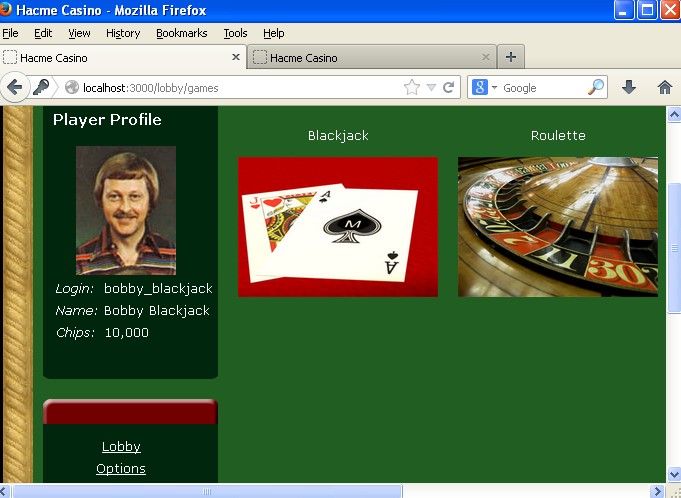
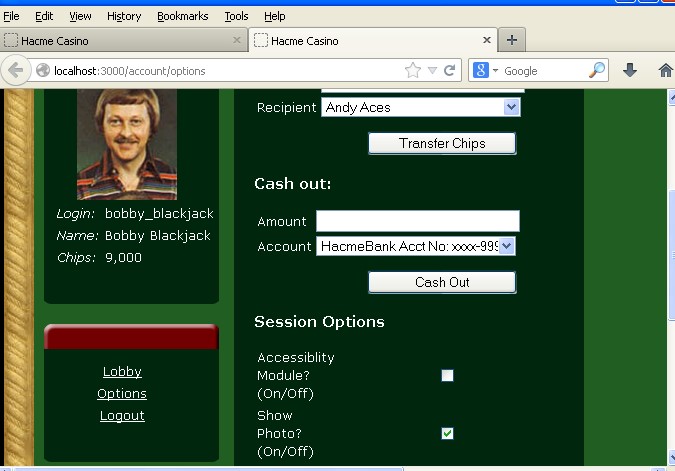
Lab 5 Report

**Lesson 1**



The above screen shots show me logging into Hacme casino. The one on the left shows the SQL I injected to login, while the one on the right shows the account I logged into as a result. This vulnerability relates to “Chapter 1: SQL Injection” in the *24 Deadly Sins* textbook. The way to mitigate this is to create a whitelist of acceptable characters for username and password input and check that the username and password do not contain characters that are not in the whitelist before using the username and password for an SQL query. Code review would have easily spotted this weakness, as the lack of input validation would have been clear and allowed for the easy elimination for this vulnerability.

**Lesson 2**

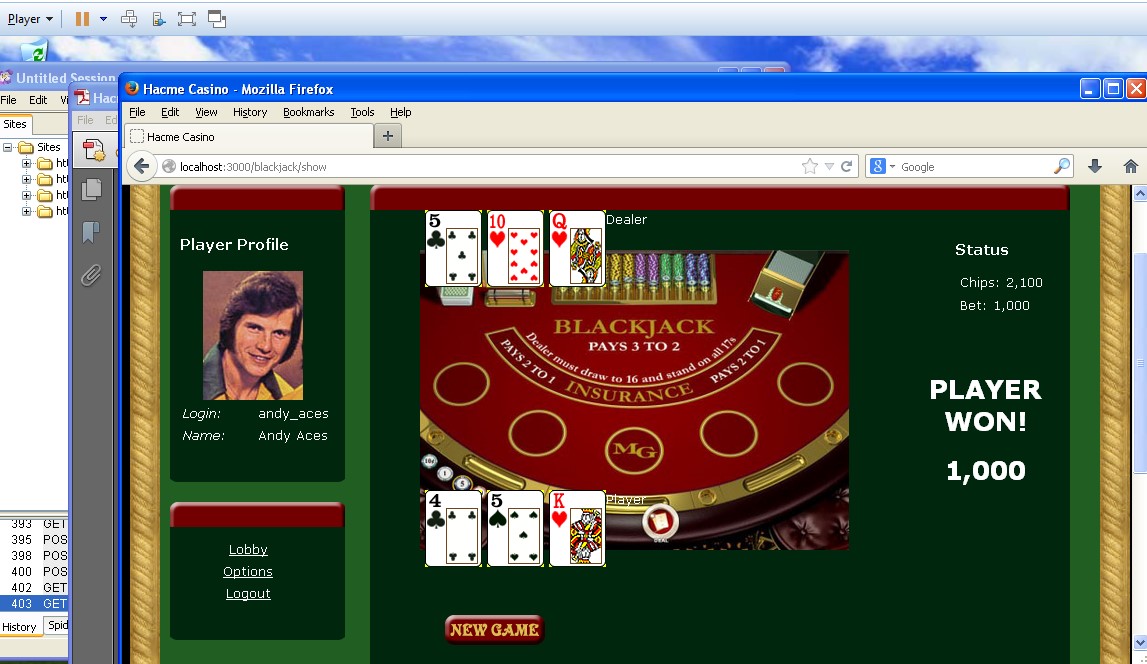
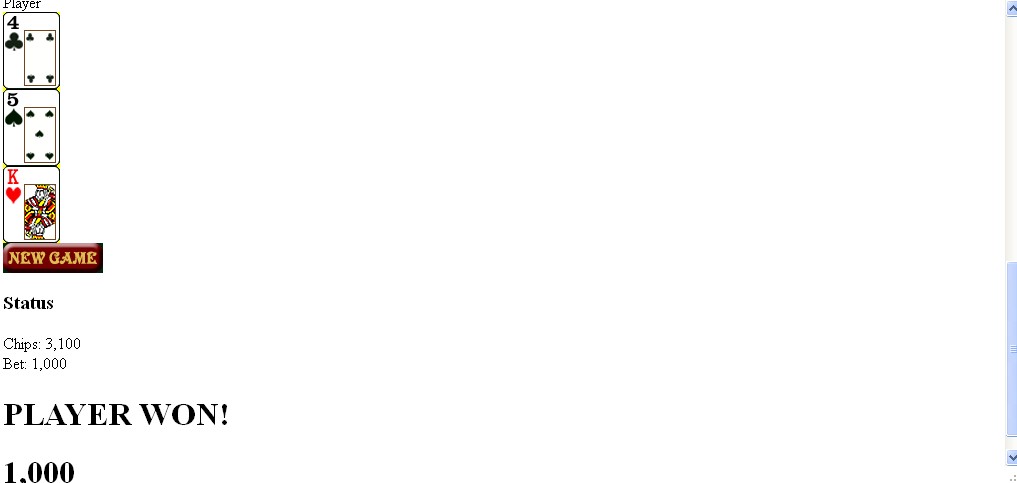


The above screen shots show Bobby Blackjack’s credit amounts before and after clicking on a link that I created that causes those that click on it to give me 1000 credits. This vulnerability relates to chapter 4 of the *24 Deadly Sins*, namely Magic URLs and hidden form fields. The way to mitigate this is to have a confirmation box pop up asking the user if he/she is sure that he/she wants to give away his/her chips, so clicking on the link won’t simply give the chips away immediately. Penetration testing would have allowed this vulnerability to be spotted and handled fairly easily, as it would have tested making a fake link like this one.

**Lesson 3**

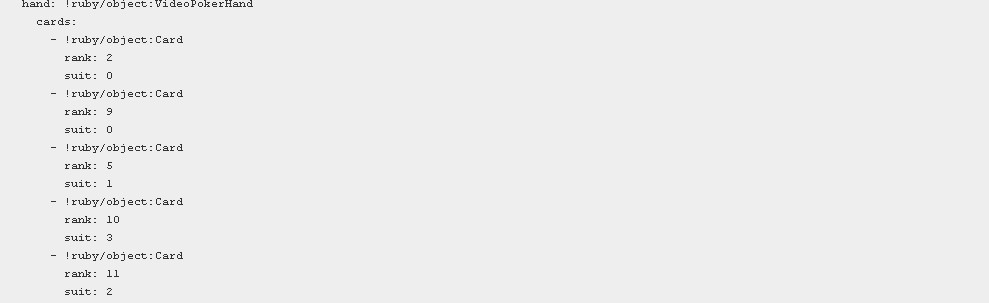
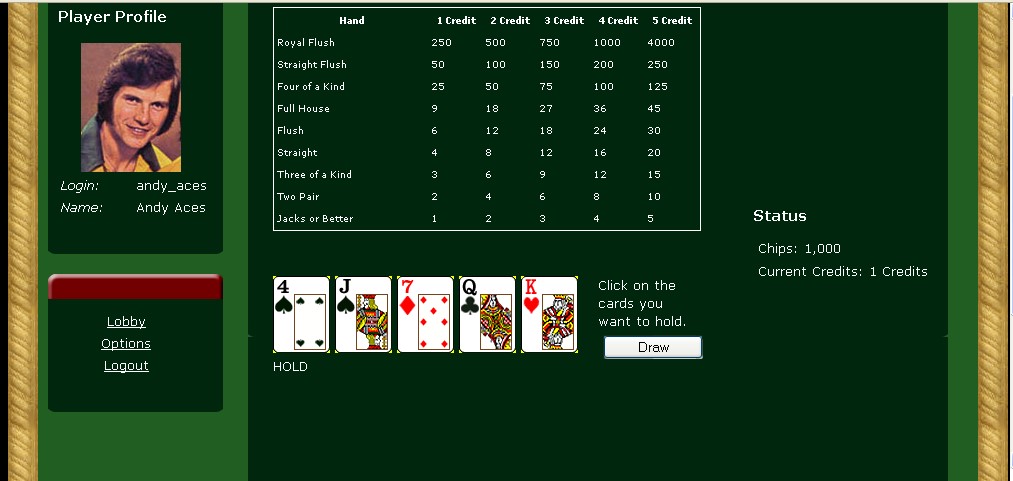
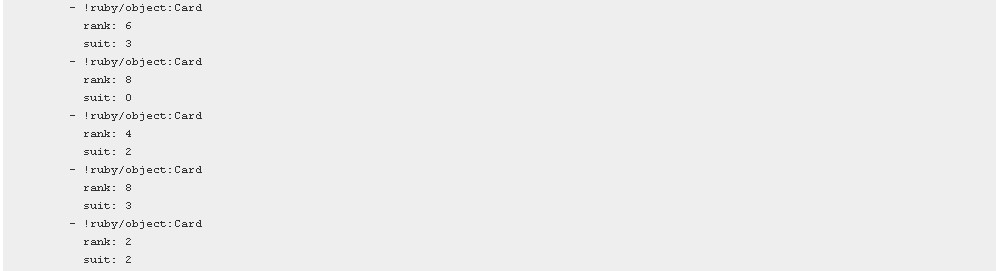
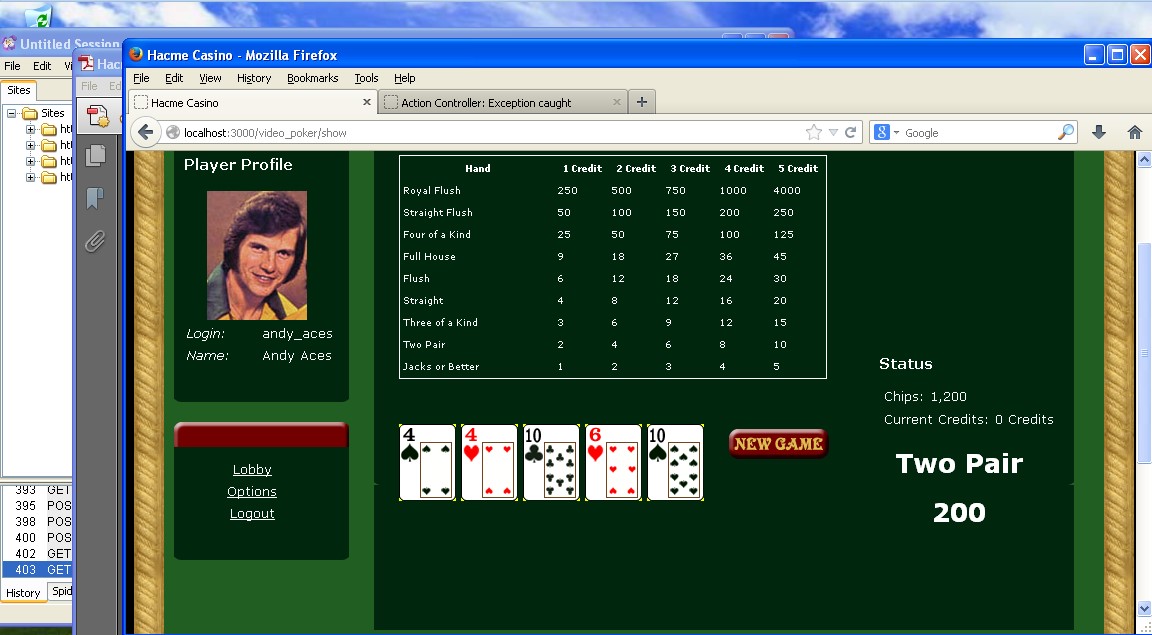
The above images show me with an unfavorable hand of blackjack followed by me returning to the lobby after leaving in the middle of that game. Clearly, I still have the same number of chips (1,100), even though I forfeited the game. This is related to chapter 11 of the *24 Deadly Sins*, “Improper Error Handling.” The way to mitigate this is to make the player lose the bet credits for leaving the game. This could have very easily been found by penetration testing, which would have tested leaving mid-game.

**Lesson 4**



The screenshots above show me winning a game of blackjack, followed by winning the same game of blackjack again by repeating the AJAX request that resulted in the game victory going off. This sin relates to chapter 3 of *24 Deadly Sins*, “Web Client-Related Vulnerabilities (XSS).” This could be mitigated by clearing the game information after game ends so that the AJAX “stay” request will do nothing after the first one. This is another vulnerability that could have easily been found with code review or penetration testing, as code review would have noticed the game state being kept even after the game ends, while penetration testing would have tested the repeat stay request call.

**Lesson 5**

The images above show my hand, the next 5 cards in the deck, the card I decided to hold (which would normally make no sense), and the end result of a win that I had because of that knowledge. This is covered in chapter 11 of *24 Deadly Sins*, “Failure to Handle Errors Correctly.” This could be mitigated by not giving so much information from an error. This would have been detected by penetration testing, which would have tried calling all of the test methods.

**The End**

