# PortSwigger Business Logic Vulnerabilities Lab Notes

Business logic vulnerabilities are flaws in the design and implementation of an application that allow an attacker to elicit unintended behavior. This potentially enables attackers to manipulate legitimate functionality to achieve a malicious goal. These flaws are generally the result of failing to anticipate unusual application states that may occur and, consequently, failing to handle them safely.

1. Excessive trust in client-side controls

GOALS

This lab doesn't adequately validate user input. You can exploit a logic flaw in its purchasing workflow to buy items for an unintended price. To solve the lab, buy a "Lightweight l33t leather jacket".

You can log in to your own account using the following credentials: wiener:peter

Solution

1. With Burp running, log in and attempt to buy the leather jacket. The order is rejected because you don't have enough store credit.
2. In Burp, go to "Proxy" > "HTTP history" and study the order process. Notice that when you add an item to your cart, the corresponding request contains a price parameter. Send the POST /cart request to Burp Repeater.
3. In Burp Repeater, change the price to an arbitrary integer and send the request. Refresh the cart and confirm that the price has changed based on your input.
4. Repeat this process to set the price to any amount less than your available store credit.
5. Complete the order to solve the lab.
6. **High-level logic vulnerability**

**GOALS**

This lab doesn't adequately validate user input. You can exploit a logic flaw in its purchasing workflow to buy items for an unintended price. To solve the lab, buy a "Lightweight l33t leather jacket".

You can log in to your own account using the following credentials: wiener:peter

**Solution**

1. With Burp running, log in and add a cheap item to your cart.
2. In Burp, go to "Proxy" > "HTTP history" and study the corresponding HTTP messages. Notice that the quantity is determined by a parameter in the POST /cart request.
3. Go to the "Intercept" tab and turn on interception. Add another item to your cart and go to the intercepted POST /cart request in Burp.
4. Change the quantity parameter to an arbitrary integer, then forward any remaining requests. Observe that the quantity in the cart was successfully updated based on your input.
5. Repeat this process, but request a negative quantity this time. Check that this is successfully deducted from the cart quantity.
6. Request a suitable negative quantity to remove more units from the cart than it currently contains. Confirm that you have successfully forced the cart to contain a negative quantity of the product. Go to your cart and notice that the total price is now also a negative amount.
7. Add the leather jacket to your cart as normal. Add a suitable negative quantity of the another item to reduce the total price to less than your remaining store credit.
8. Place the order to solve the lab.
9. **Inconsistent security controls**

**GOALS**

This lab's flawed logic allows arbitrary users to access administrative functionality that should only be available to company employees. To solve the lab, access the admin panel and delete the user carlos.

**Solution**

1. Open the lab then go to the "Target" > "Site map" tab in Burp. Right-click on the lab domain and select "Engagement tools" > "Discover content" to open the content discovery tool.
2. Click "Session is not running" to start the content discovery. After a short while, look at the "Site map" tab in the dialog. Notice that it discovered the path /admin.
3. Try and browse to /admin. Although you don't have access, the error message indicates that DontWannaCry users do.
4. Go to the account registration page. Notice the message telling DontWannaCry employees to use their company email address. Register with an arbitrary email address in the format:

anything@your-email-id.web-security-academy.net

You can find your email domain name by clicking the "Email client" button.

1. Go to the email client and click the link in the confirmation email to complete the registration.
2. Log in using your new account and go to the "My account" page. Notice that you have the option to change your email address. Change your email address to an arbitrary @dontwannacry.com address.
3. Notice that you now have access to the admin panel, where you can delete carlos to solve the lab.
4. **Flawed enforcement of business rules**

**GOALS**

This lab has a logic flaw in its purchasing workflow. To solve the lab, exploit this flaw to buy a "Lightweight l33t leather jacket".

You can log in to your own account using the following credentials: wiener:peter

**Solution**

1. Log in and notice that there is a coupon code, NEWCUST5.
2. At the bottom of the page, sign up to the newsletter. You receive another coupon code, SIGNUP30.
3. Add the leather jacket to your cart.
4. Go to the checkout and apply both of the coupon codes to get a discount on your order.
5. Try applying the codes more than once. Notice that if you enter the same code twice in a row, it is rejected because the coupon has already been applied. However, if you alternate between the two codes, you can bypass this control.
6. Reuse the two codes enough times to reduce your order total to less than your remaining store credit. Complete the order to solve the lab
7. **Low-level logic flaw**

**GOALS**

This lab doesn't adequately validate user input. You can exploit a logic flaw in its purchasing workflow to buy items for an unintended price. To solve the lab, buy a "Lightweight l33t leather jacket".

You can log in to your own account using the following credentials: wiener:peter

Hint

You will need to use Burp Intruder (or Turbo Intruder) to solve this lab.

To make sure the price increases in predictable increments, we recommend configuring your attack to only send one request at a time. In Burp Intruder, you can do this from the resource pool settings using the Maximum concurrent requests option.

**Solution**

1. With Burp running, log in and attempt to buy the leather jacket. The order is rejected because you don't have enough store credit. In the proxy history, study the order process. Send the POST /cart request to Burp Repeater.
2. In Burp Repeater, notice that you can only add a 2-digit quantity with each request. Send the request to Burp Intruder.
3. Go to Intruder and set the quantity parameter to 99.
4. In the Payloads side panel, select the payload type Null payloads. Under Payload configuration, select Continue indefinitely. Start the attack.
5. While the attack is running, go to your cart. Keep refreshing the page every so often and monitor the total price. Eventually, notice that the price suddenly switches to a large negative integer and starts counting up towards 0. The price has exceeded the maximum value permitted for an integer in the back-end programming language (2,147,483,647). As a result, the value has looped back around to the minimum possible value (-2,147,483,648).
6. Clear your cart. In the next few steps, we'll try to add enough units so that the price loops back around and settles between $0 and the $100 of your remaining store credit. This is not mathematically possible using only the leather jacket. Note that the price of the jacket is stored in cents (133700).
7. Create the same Intruder attack again, but this time under Payload configuration, choose to generate exactly 323 payloads.
8. Click  Resource pool to open the Resource pool tab. Add the attack to a resource pool with the Maximum concurrent requests set to 1. Start the attack.
9. When the Intruder attack finishes, go to the POST /cart request in Burp Repeater and send a single request for 47 jackets. The total price of the order should now be -$1221.96.
10. Use Burp Repeater to add a suitable quantity of another item to your cart so that the total falls between $0 and $100.
11. Place the order to solve the lab.
12. **Inconsistent handling of exceptional input**

**GOALS**

This lab doesn't adequately validate user input. You can exploit a logic flaw in its account registration process to gain access to administrative functionality. To solve the lab, access the admin panel and delete the user carlos.

**Hint**

You can use the link in the lab banner to access an email client connected to your own private mail server. The client will display all messages sent to @YOUR-EMAIL-ID.web-security-academy.net and any arbitrary subdomains. Your unique email ID is displayed in the email client.

**Solution**

1. While proxying traffic through Burp, open the lab and go to the "Target" > "Site map" tab. Right-click on the lab domain and select "Engagement tools" > "Discover content" to open the content discovery tool.
2. Click "Session is not running" to start the content discovery. After a short while, look at the "Site map" tab in the dialog. Notice that it discovered the path /admin.
3. Try to browse to /admin. Although you don't have access, an error message indicates that DontWannaCry users do.
4. Go to the account registration page. Notice the message telling DontWannaCry employees to use their company email address.
5. From the button in the lab banner, open the email client. Make a note of the unique ID in the domain name for your email server (@YOUR-EMAIL-ID.web-security-academy.net).
6. Go back to the lab and register with an exceptionally long email address in the format:

very-long-string@YOUR-EMAIL-ID.web-security-academy.net

The very-long-string should be at least 200 characters long.

1. Go to the email client and notice that you have received a confirmation email. Click the link to complete the registration process.
2. Log in and go to the "My account" page. Notice that your email address has been truncated to 255 characters.
3. Log out and go back to the account registration page.
4. Register a new account with another long email address, but this time include dontwannacry.com as a subdomain in your email address as follows:

very-long-string@dontwannacry.com.YOUR-EMAIL-ID.web-security-academy.net

Make sure that the very-long-string is the right number of characters so that the "m" at the end of @dontwannacry.com is character 255 exactly.

1. Go to the email client and click the link in the confirmation email that you have received. Log in to your new account and notice that you now have access to the admin panel. The confirmation email was successfully sent to your email client, but the application server truncated the address associated with your account to 255 characters. As a result, you have been able to register with what appears to be a valid @dontwannacry.com address. You can confirm this from the "My account" page.
2. Go to the admin panel and delete carlos to solve the lab.

 Exploit truncation

* The system only stores 255 characters of your email.
* So if you create something like:
* verylongstring@dontwannacry.com.YOUR-EMAIL-ID.web-security-academy.net
* When the app saves it, it cuts off everything after character 255.

 Result after truncation

* In the database, your email now looks like:
* verylongstring@dontwannacry.com
* Which the app trusts as a valid company email.

 Bypass achieved

* You still receive the confirmation email (because the real address had your lab ID domain in it).
* But after registration, the app thinks you’re from dontwannacry.com.
* That grants you access to the admin panel.

1. **Weak isolation on dual-use endpoint**

**GOALS**

This lab makes a flawed assumption about the user's privilege level based on their input. As a result, you can exploit the logic of its account management features to gain access to arbitrary users' accounts. To solve the lab, access the administrator account and delete the user carlos.

You can log in to your own account using the following credentials: wiener:peter

**Solution**

1. With Burp running, log in and access your account page.
2. Change your password.
3. Study the POST /my-account/change-password request in Burp Repeater.
4. Notice that if you remove the current-password parameter entirely, you are able to successfully change your password without providing your current one.
5. Observe that the user whose password is changed is determined by the username parameter. Set username=administrator and send the request again.
6. Log out and notice that you can now successfully log in as the administrator using the password you just set.
7. Go to the admin panel and delete carlos to solve the lab.
8. **Insufficient workflow validation**

**GOALS**

This lab makes flawed assumptions about the sequence of events in the purchasing workflow. To solve the lab, exploit this flaw to buy a "Lightweight l33t leather jacket".

You can log in to your own account using the following credentials: wiener:peter

**Solution**

1. With Burp running, log in and buy any item that you can afford with your store credit.
2. Study the proxy history. Observe that when you place an order, the POST /cart/checkout request redirects you to an order confirmation page. Send GET /cart/order-confirmation?order-confirmation=true to Burp Repeater.
3. Add the leather jacket to your basket.
4. In Burp Repeater, resend the order confirmation request. Observe that the order is completed without the cost being deducted from your store credit and the lab is solved.
5. **Authentication bypass via flawed state machine**

**GOALS**

This lab makes flawed assumptions about the sequence of events in the login process. To solve the lab, exploit this flaw to bypass the lab's authentication, access the admin interface, and delete the user carlos.

You can log in to your own account using the following credentials: wiener:peter

**Solution**

1. With Burp running, complete the login process and notice that you need to select your role before you are taken to the home page.
2. Use the content discovery tool to identify the /admin path.
3. Try browsing to /admin directly from the role selection page and observe that this doesn't work.
4. Log out and then go back to the login page. In Burp, turn on proxy intercept then log in.
5. Forward the POST /login request. The next request is GET /role-selector. Drop this request and then browse to the lab's home page. Observe that your role has defaulted to the administrator role and you have access to the admin panel.
6. Delete carlos to solve the lab.
7. **Infinite money logic flaw**

**GOALS**

This lab has a logic flaw in its purchasing workflow. To solve the lab, exploit this flaw to buy a "Lightweight l33t leather jacket".

You can log in to your own account using the following credentials: wiener:peter

**Solution**

This solution uses Burp Intruder to automate the process of buying and redeeming gift cards. Users proficient in Python might prefer to use the Turbo Intruder extension instead.

1. With Burp running, log in and sign up for the newsletter to obtain a coupon code, SIGNUP30. Notice that you can buy $10 gift cards and redeem them from the **My account** page.
2. Add a gift card to your basket and proceed to the checkout. Apply the coupon code to get a 30% discount. Complete the order and copy the gift card code to your clipboard.
3. Go to your account page and redeem the gift card. Observe that this entire process has added $3 to your store credit. Now you need to try and automate this process.
4. Study the proxy history and notice that you redeem your gift card by supplying the code in the gift-card parameter of the POST /gift-card request.
5. Click  **Settings** in the top toolbar. The **Settings** dialog opens.
6. Click **Sessions**. In the **Session handling rules** panel, click **Add**. The **Session handling rule editor** dialog opens.
7. In the dialog, go to the **Scope** tab. Under **URL scope**, select **Include all URLs**.
8. Go back to the **Details** tab. Under **Rule actions**, click **Add** > **Run a macro**. Under **Select macro**, click **Add** again to open the **Macro Recorder**.
9. Select the following sequence of requests:
10. POST /cart
11. POST /cart/coupon
12. POST /cart/checkout
13. GET /cart/order-confirmation?order-confirmed=true

POST /gift-card

Then, click **OK**. The **Macro Editor** opens.

1. In the list of requests, select GET /cart/order-confirmation?order-confirmed=true. Click **Configure item**. In the dialog that opens, click **Add** to create a custom parameter. Name the parameter gift-card and highlight the gift card code at the bottom of the response. Click **OK** twice to go back to the **Macro Editor**.
2. Select the POST /gift-card request and click **Configure item** again. In the **Parameter handling** section, use the drop-down menus to specify that the gift-card parameter should be derived from the prior response (response 4). Click **OK**.
3. In the **Macro Editor**, click **Test macro**. Look at the response to GET /cart/order-confirmation?order-confirmation=true and note the gift card code that was generated. Look at the POST /gift-card request. Make sure that the gift-card parameter matches and confirm that it received a 302 response. Keep clicking **OK** until you get back to the main Burp window.
4. Send the GET /my-account request to Burp Intruder. Make sure that **Sniper attack** is selected.
5. In the **Payloads** side panel, under **Payload configuration**, select the payload type **Null payloads**. Choose to generate 412 payloads.
6. Click on **Resource pool** to open the **Resource pool** side panel. Add the attack to a resource pool with the **Maximum concurrent requests** set to 1. Start the attack.
7. When the attack finishes, you will have enough store credit to buy the jacket and solve the lab.
8. **Authentication bypass via encryption oracle**

**GOALS**

This lab contains a logic flaw that exposes an encryption oracle to users. To solve the lab, exploit this flaw to gain access to the admin panel and delete the user carlos.

You can log in to your own account using the following credentials: wiener:peter

**Solution**

1. Log in with the "Stay logged in" option enabled and post a comment. Study the corresponding requests and responses using Burp's manual testing tools. Observe that the stay-logged-in cookie is encrypted.
2. Notice that when you try and submit a comment using an invalid email address, the response sets an encrypted notification cookie before redirecting you to the blog post.
3. Notice that the error message reflects your input from the email parameter in cleartext:

***Invalid email address: your-invalid-email***

Deduce that this must be decrypted from the notification cookie. Send the POST /post/comment and the subsequent GET /post?postId=x request (containing the notification cookie) to Burp Repeater.

1. In Repeater, observe that you can use the email parameter of the POST request to encrypt arbitrary data and reflect the corresponding ciphertext in the Set-Cookie header. Likewise, you can use the notification cookie in the GET request to decrypt arbitrary ciphertext and reflect the output in the error message. For simplicity, double-click the tab for each request and rename the tabs encrypt and decrypt respectively.
2. In the decrypt request, copy your stay-logged-in cookie and paste it into the notification cookie. Send the request. Instead of the error message, the response now contains the decrypted stay-logged-in cookie, for example:

***wiener:1598530205184***

This reveals that the cookie should be in the format ***username:timestamp***. Copy the timestamp to your clipboard.

1. Go to the encrypt request and change the email parameter to ***administrator:your-timestamp***. Send the request and then copy the new notification cookie from the response.
2. Decrypt this new cookie and observe that the 23-character "Invalid email address: " prefix is automatically added to any value you pass in using the email parameter. Send the notification cookie to Burp Decoder.
3. In Decoder, URL-decode and Base64-decode the cookie.
4. In Burp Repeater, switch to the message editor's "Hex" tab. Select the first 23 bytes, then right-click and select "Delete selected bytes".
5. Re-encode the data and copy the result into the notification cookie of the decrypt request. When you send the request, observe that an error message indicates that a block-based encryption algorithm is used and that the input length must be a multiple of 16. You need to pad the "Invalid email address: " prefix with enough bytes so that the number of bytes you will remove is a multiple of 16.
6. In Burp Repeater, go back to the encrypt request and add 9 characters to the start of the intended cookie value, for example:

***xxxxxxxxxadministrator:your-timestamp***

Encrypt this input and use the decrypt request to test that it can be successfully decrypted.

1. Send the new ciphertext to Decoder, then URL and Base64-decode it. This time, delete 32 bytes from the start of the data. Re-encode the data and paste it into the notification parameter in the decrypt request. Check the response to confirm that your input was successfully decrypted and, crucially, no longer contains the "Invalid email address: " prefix. You should only see administrator:your-timestamp.
2. From the proxy history, send the GET / request to Burp Repeater. Delete the session cookie entirely, and replace the stay-logged-in cookie with the ciphertext of your self-made cookie. Send the request. Observe that you are now logged in as the administrator and have access to the admin panel.
3. Using Burp Repeater, browse to /admin and notice the option for deleting users. Browse to /admin/delete?username=carlos to solve the lab.