

# View Someone Else's Cart

Any time multiple users have separate data, you want to test how well they're kept apart. There's a shopping cart here, and each user should probably have their own cart, and should not be able to see what's in other people's carts. How does that work?

## Ideas

If the cart has an identifier associated with it, and if that identifier is a predictable value, what happens if you submit other similar values instead?

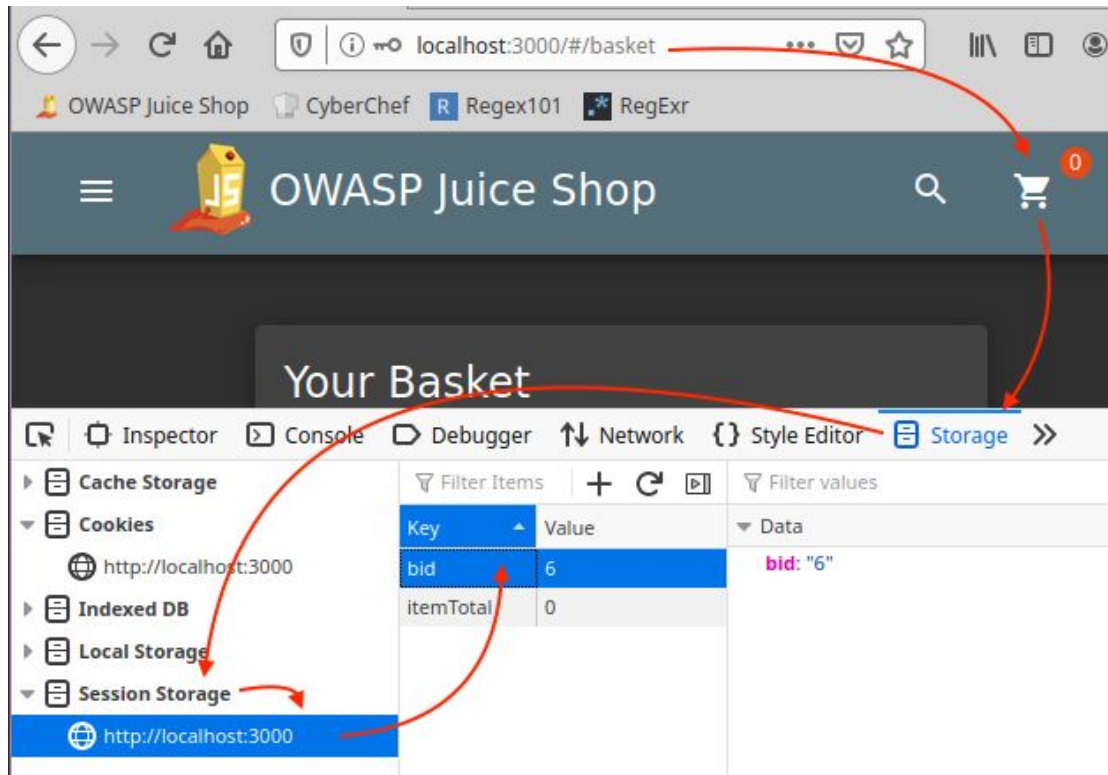
## Alternatives

This walk-thru shows two different solutions: first, using only the browser (and a lucky number guess) and the second using Burp Intruder (because the guessing isn't always this easy.)

## Walk-Thru: In-Browser

First Method: All in the browser, with Developer Tools

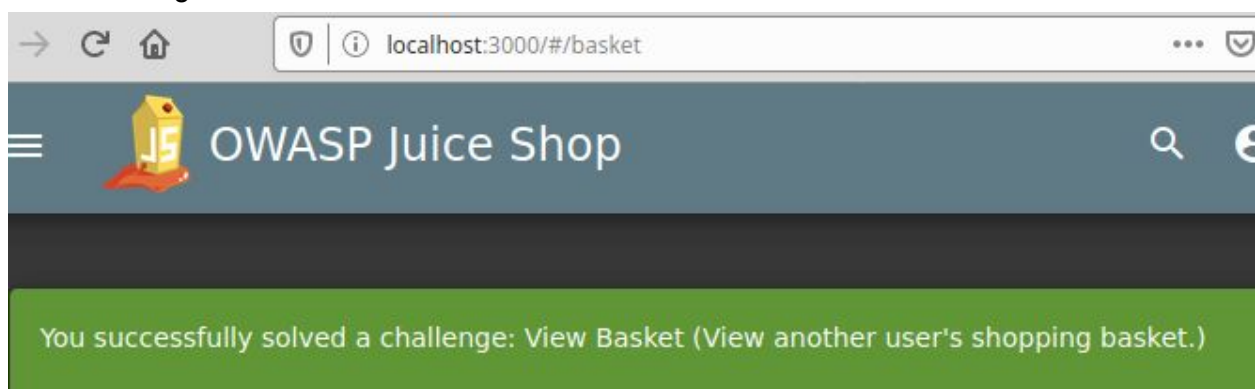
1. Log in as your user and click "Your Basket" at the top right of the page. It doesn't matter if you have things in there or not.
2. Open Developer Tools in your Browser (F12 or Ctrl-Shift-i) and open the Storage tab.
3. Open the Session Storage container and click on "http://localhost:3000/" to see what's there. Click on the "Key" called "bid"



*Viewing Local "Session Storage"*

4. Double-click on the number in the "Value" column of the "bid" row and replace the number there with the number 2, then hit "enter" or click outside the field to save your edit.

5. Click the "reload" button next to the browser's address bar and you should see the reward banner. Juice Shop tracks your achievements, so if you already saw this banner before, you won't see it again now.



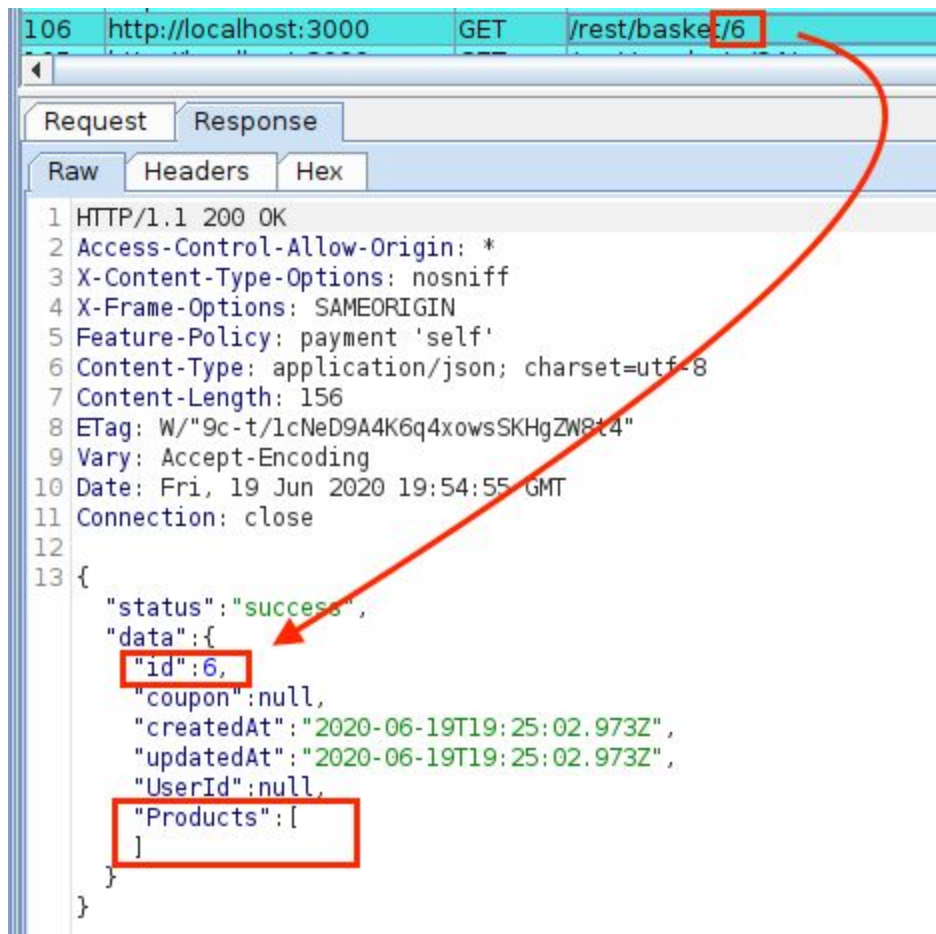
# Walk-Thru: Burp Suite

What if the numbers weren't so easy to guess? Let's look at how Burp Intruder can help automate guessing.

Make sure you have Firefox set to use your Burp Suite as a proxy, and that the Proxy > Intercept pane says "Intercept is off"

1. Log in as your user and click "Your Basket" at the top right of the page. It doesn't matter if you have things in there or not.

2. Look at Burp's Proxy History and find the GET request sent to /rest/basket/6 (you may have a different number). Notice that the number in the URL is also in the JSON in the response body, where it's called "id"

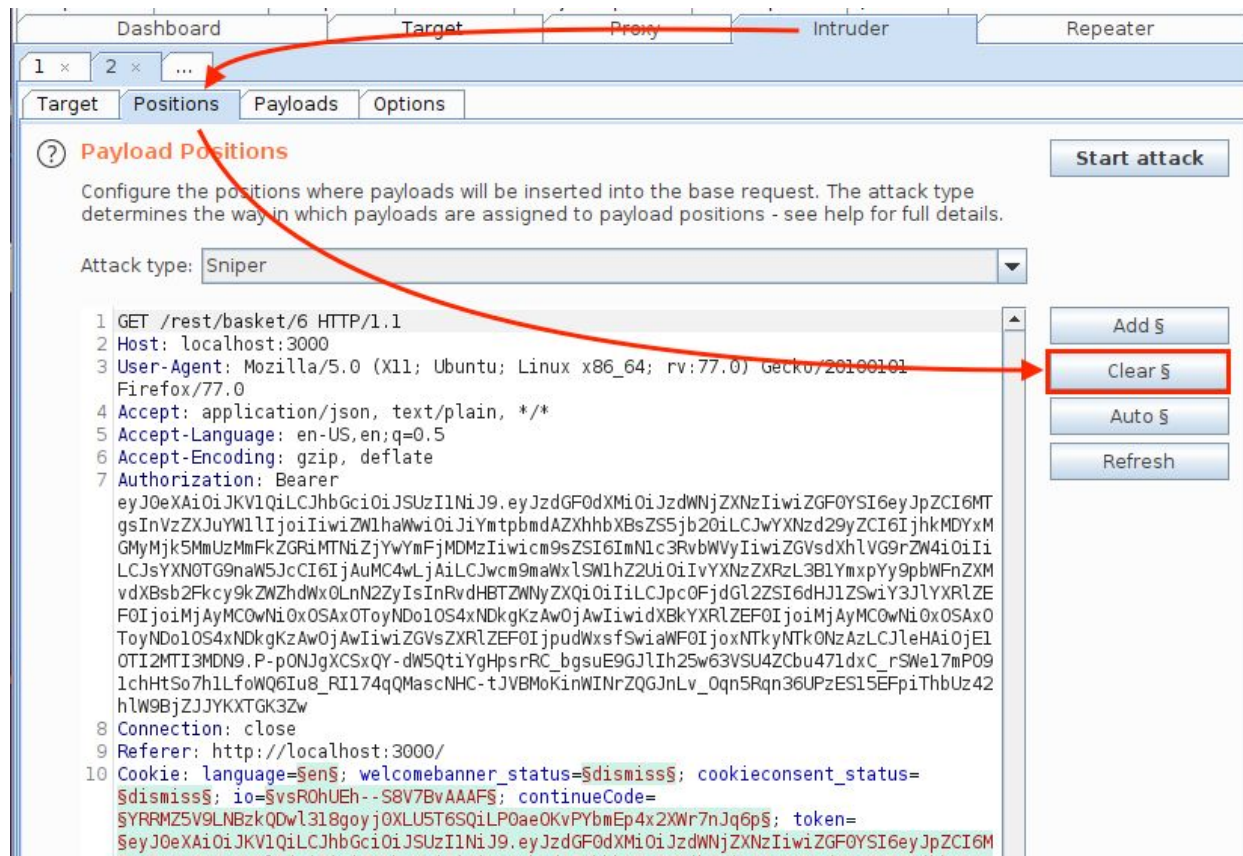


*My "basket" Had ID Number 6 (yours may be different)*

3. Right click on this request and choose "Send to Intruder"

Intruder is one of Burp's more powerful tools, and it is deliberately weakened in the Community version of Burp Suite. The limitations are not a problem for this lab, though.

4. Click over to the Intruder tab, then the Positions tab. Once you're there, click the "Clear" button to erase the section symbols that mark Burp's guess at where you want to send attacks.



*Clear Burp's Suggested Insertion Points*

5. Highlight the number at the end of the URL and click the "Add" button to tell Burp Intruder which part of this request you want to work with. When you're done, you should see your number with a section symbol right next to it on either side with no spaces between the section symbols and one space before "HTTP/1.1".

```
1 GET /rest/basket/6 HTTP/1.1
2 Host: localhost:3000
3 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:77.0) Gecko/20100101 Firefox/77.0
```

*Like this*

```
1 GET /rest/basket/$6$ HTTP/1.1
2 Host: localhost:3000
3 User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:77.0) Gecko/20100101 Firefox/77.0
```

*NOT like this.*

6. Leave the "Attack Type" as "sniper" and click on the "Payloads" tab.

7. Choose a Payload Type of "Numbers" and set the Payload Options to go "from" zero "to" 10 with at "step" of 1. Notice the "Payload count" tells you how many values there will be.

The screenshot shows the 'Payloads' tab of a web application. At the top, there are four tabs: 'Target', 'Positions', 'Payloads' (selected), and 'Options'. Below the tabs, there are two main sections. The first section is titled 'Payload Sets' and contains a description: 'You can define one or more payload sets. The number of payload sets depends on the attack type defined in the Positions tab. Various payload types are available for each payload set, and each payload type can be customized in different ways.' Below this, there are two rows of settings. The first row is 'Payload set: 1' with a dropdown arrow, and 'Payload count: 11'. The second row is 'Payload type: Numbers' with a dropdown arrow, and 'Request count: 11'. The 'Numbers' payload type is highlighted with a red box. The second section is titled 'Payload Options [Numbers]' and contains a description: 'This payload type generates numeric payloads within a given range and in a specified format.' Below this, there is a 'Number range' section. It has a 'Type:' label with two radio buttons: 'Sequential' (selected) and 'Random'. Below the radio buttons, there are three input fields: 'From: 0', 'To: 10', and 'Step: 1'. These three input fields are highlighted with a red box. At the bottom, there is a 'How many:' label followed by an empty input field.

*Setting Payload Options*

8. Click "Start attack" and click "OK" on the warning message about how Intruder's features are limited in the Community version.

9. Notice the "Length" column in the results table. Some responses are larger than others. Click on the rows in the table one by one and inspect the responses. Notice that several of these requests returned information about other users' shopping baskets. Remember that basket number 6 is yours, so seeing that one is not a problem.



Request	Payload	Status	Error	Timeout	Length
0		200	<input type="checkbox"/>	<input type="checkbox"/>	490
1	0	200	<input type="checkbox"/>	<input type="checkbox"/>	363
2	1	200	<input type="checkbox"/>	<input type="checkbox"/>	1646
3	2	200	<input type="checkbox"/>	<input type="checkbox"/>	892

Request	Response
Raw	Headers
15	1

```

"status": "success",
"data": {
  "id": 1,
  "coupon": null,
  "createdAt": "2020-06-19T14:44:45.372Z",
  "updatedAt": "2020-06-19T14:44:45.372Z",
  "UserId": 1,
  "Products": [
    {
      "id": 1,
      "name": "Apple Juice (1000ml)",
      "description": "The all-time classic.",
      "price": 1.99,
      "deluxePrice": 0.99,

```

*Basket #1 belongs to UserId #1 and has several items in it*

Request	Payload	Status	Error	Timeout	Length
6	5	200	<input type="checkbox"/>	<input type="checkbox"/>	1281
7	6	200	<input type="checkbox"/>	<input type="checkbox"/>	490
8	7	200	<input type="checkbox"/>	<input type="checkbox"/>	363

Request	Response
Raw	Headers
10	Date: Fri, 19 Jun 2020 20:09:25 GMT
11	Connection: close
12	
13	{

```

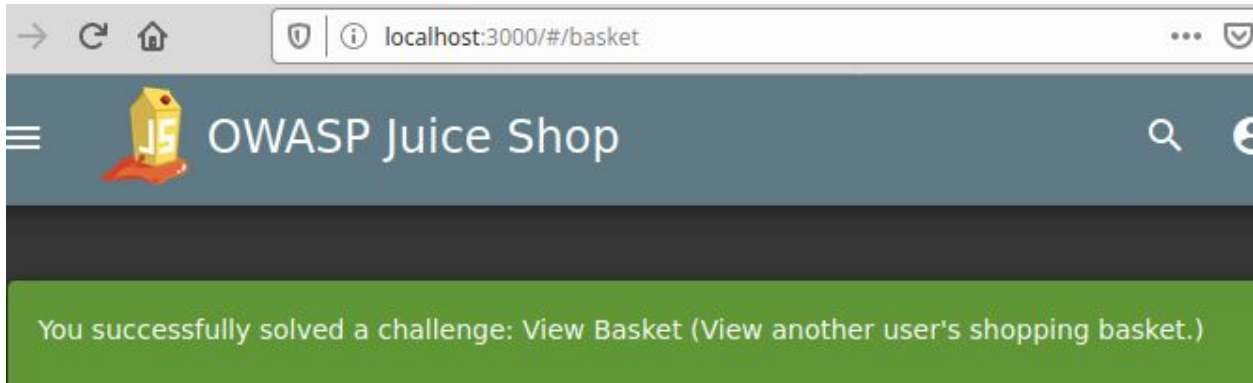
"status": "success",
"data": {
  "id": 5,
  "coupon": null,
  "createdAt": "2020-06-19T14:44:45.372Z",
  "updatedAt": "2020-06-19T14:44:45.372Z",
  "UserId": 16,
  "Products": [
    {
      "id": 3,
      "name": "Eggfruit Juice (500ml)",
      "description": "Now with even more exotic flavour.",
      "price": 8.99,
      "deluxePrice": 8.99,
      "image": "eggfruit_juice.jpg",

```

*Basket #5 belongs to UserId #16*

Notice that the basket numbers and the user numbers are not always the same. That doesn't get you anything in this lab, but it is the kind of thing you should take notice of when you're testing webapps.

10. Return to the browser to see your achievement. Juice Shop tracks your achievements, so if you already earned this banner before, you won't see it again now.



## For Further Practice: Digi.Ninja labs

- Client-Side Authentication: <https://authlab.digi.ninja/ClientSide>
- Permissions based on user-agent: <https://authlab.digi.ninja/UserAgent>
- IP Address based authentication: <https://authlab.digi.ninja/Bypass>