Forge a JWT

JSON Web Tokens offer a great place to practice using different tools to achieve the same objective. The specific vulnerability shown here is a real one and although the common libraries used for JWT don't seem to be vulnerable anymore, the issue does still occur.
More than that, though, JWTs offer a chance to learn how to attack input parameters that may be encoded or otherwise not easily manipulated. These fields will not be tested by vulnerability scanners, and cannot be tested by the simple methods normally used by hand.

We'll look at a Burp Suite extension that automates the basic attack, then see how to re-create that "by hand" in a way that offers far more flexibility and gets you comfortable with more advanced attack methods.

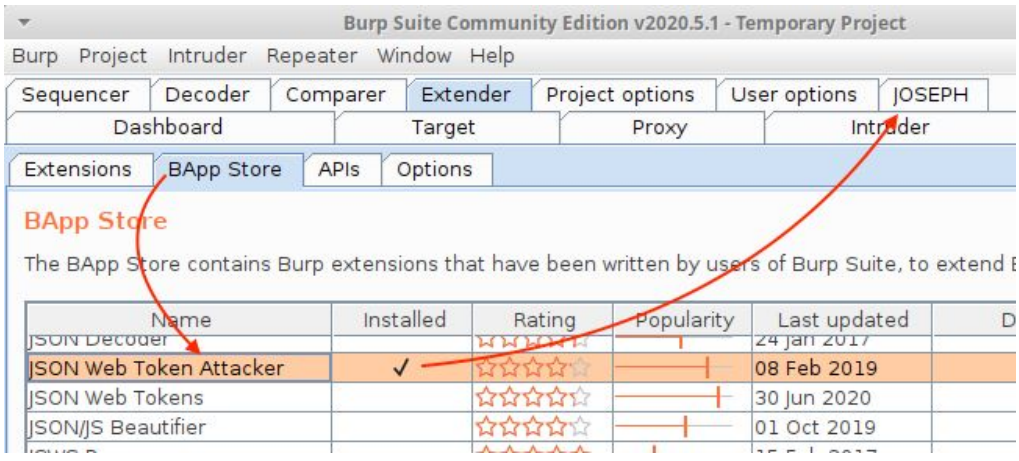
Walk-Thru

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

You get your award by forging a JWT for the user jwtn3d@juice-sh.op (which is not a user that exists in the application).

First: Burp Extension: JSON Web Token Attacker

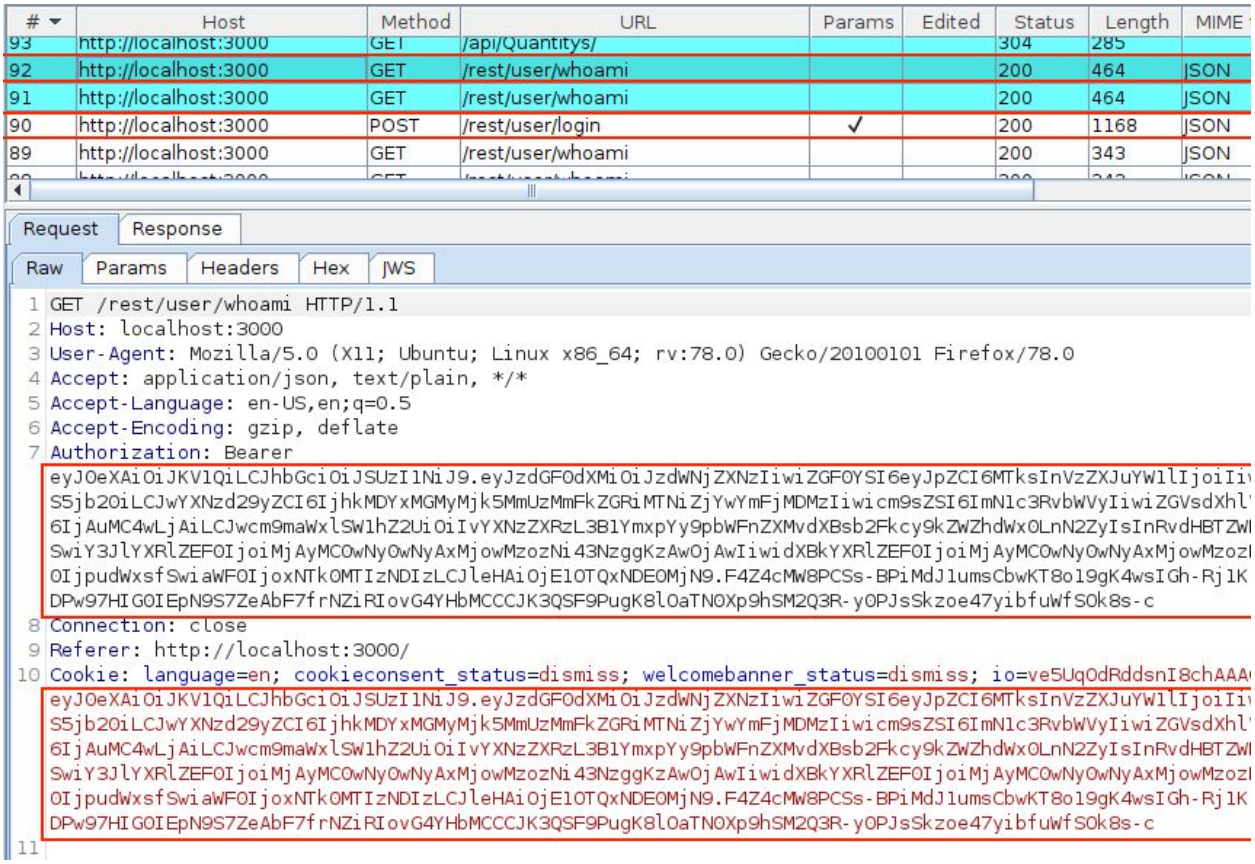
Tools are helpful, but you need to know what they're doing.
The Burp Extension is called "JSON Web Token Attacker" but the tab it creates in the UI says "JOSEPH". This extension is already installed on the VM for the class.



JSON Web Token Attacker Extension

1. Log in to Juice Shop as your user. Find the POST request to /rest/user/login in the proxy history that includes your successful logon, then find the GET request to /rest/user/whoami that

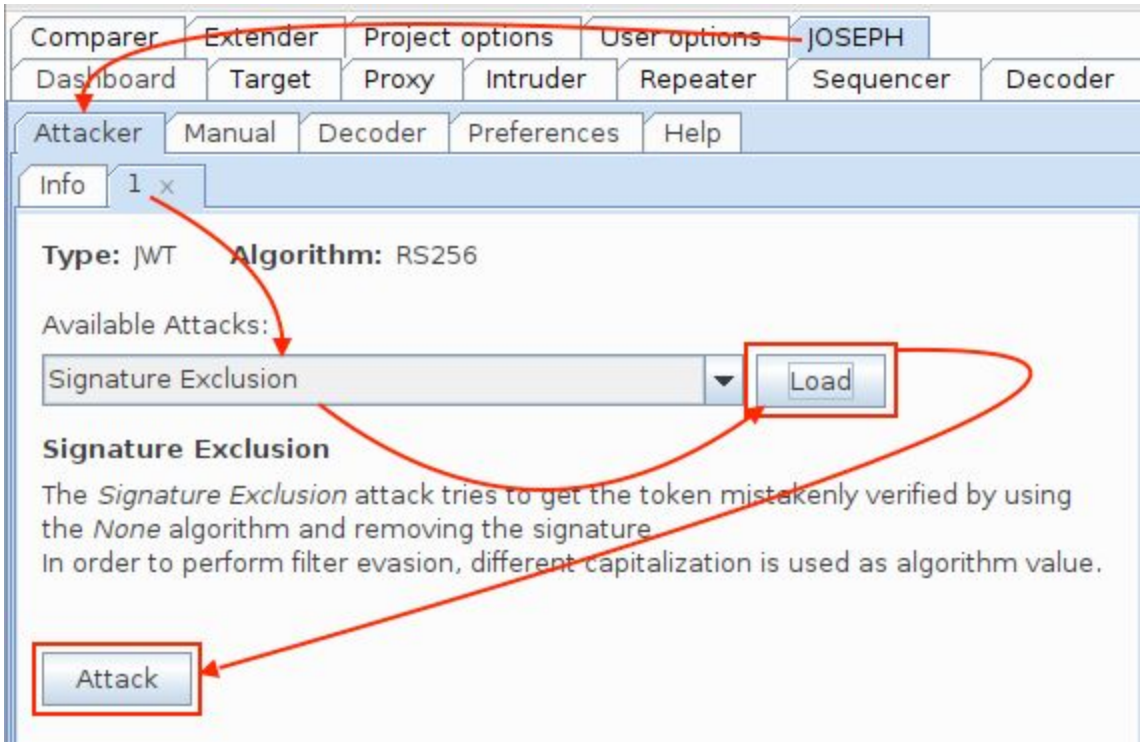
happens after that. The JSON Web Token Attacker extension uses teal highlighting to indicate traffic with JWTs.



Request to /rest/user/whoami Includes JWT

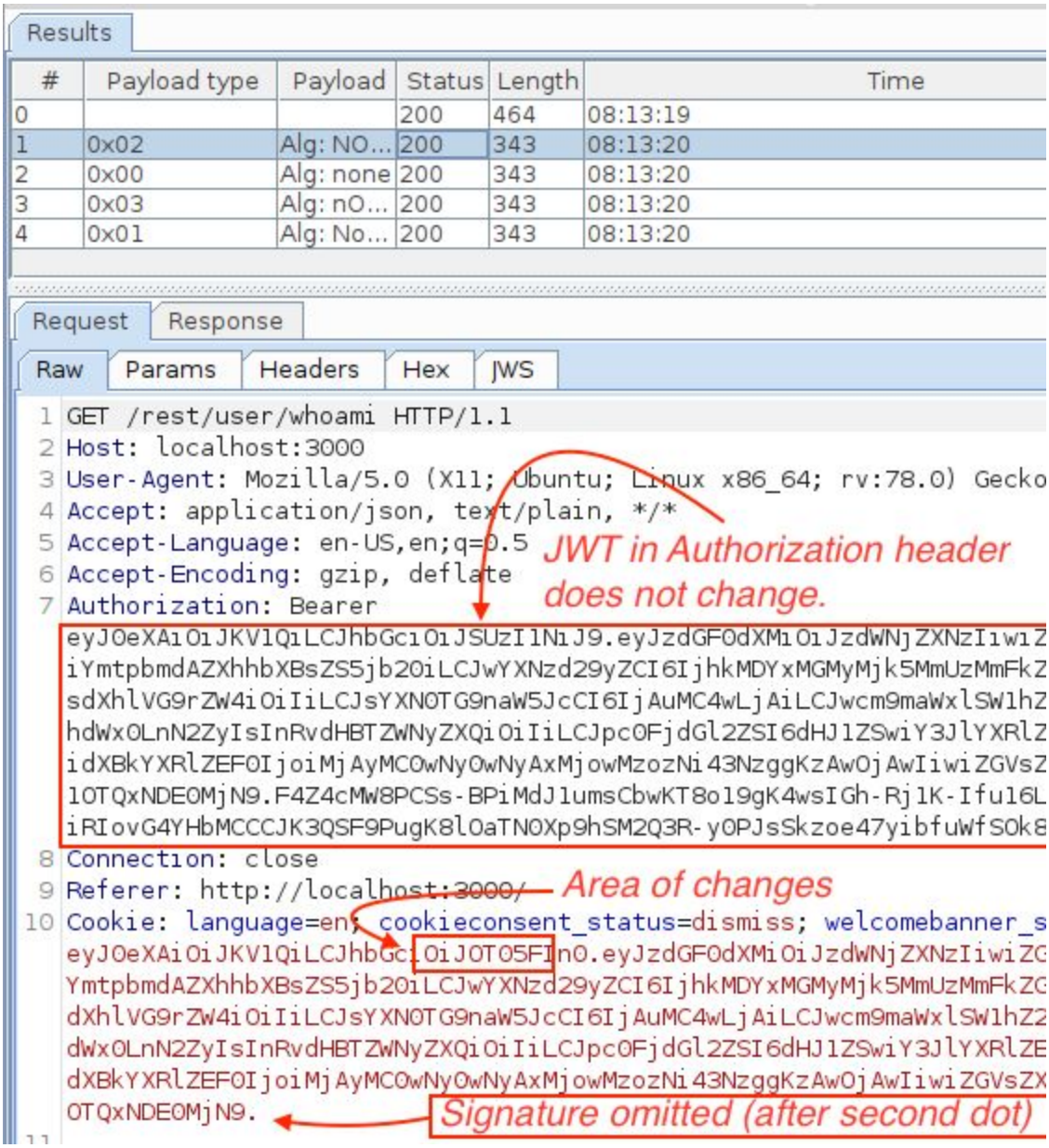
2. Right-click on this request in the list of requests in the top pane (the option is disabled if you're in the request/response pane) and choose "Send to JOSEPH".

3. In the JOSEPH tab, choose the "Signature Exclusion" attack, click "Load" then click "Attack"



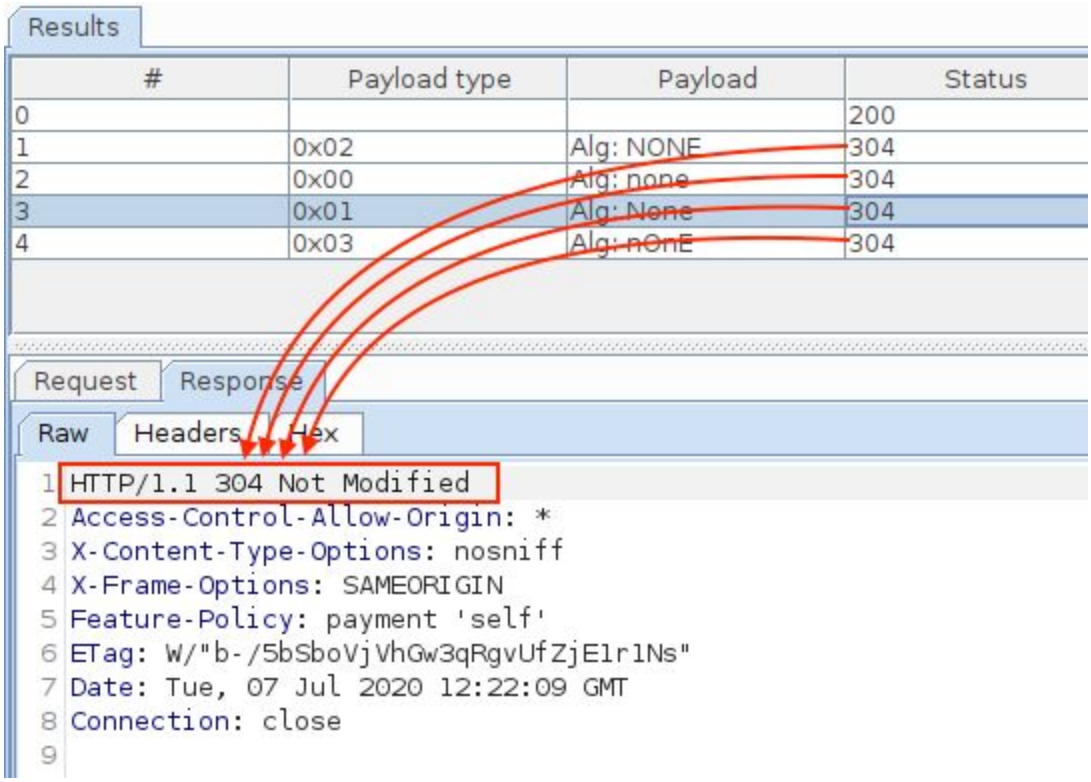
Starting JSON Web Attacker Attack

4. Notice in the table that none of the attacks seemed to be successful. Open the "Request" tab and click through the tests in the table. Notice that only the token in the Cookie is being manipulated. The "Authorization" header is the same for all requests.

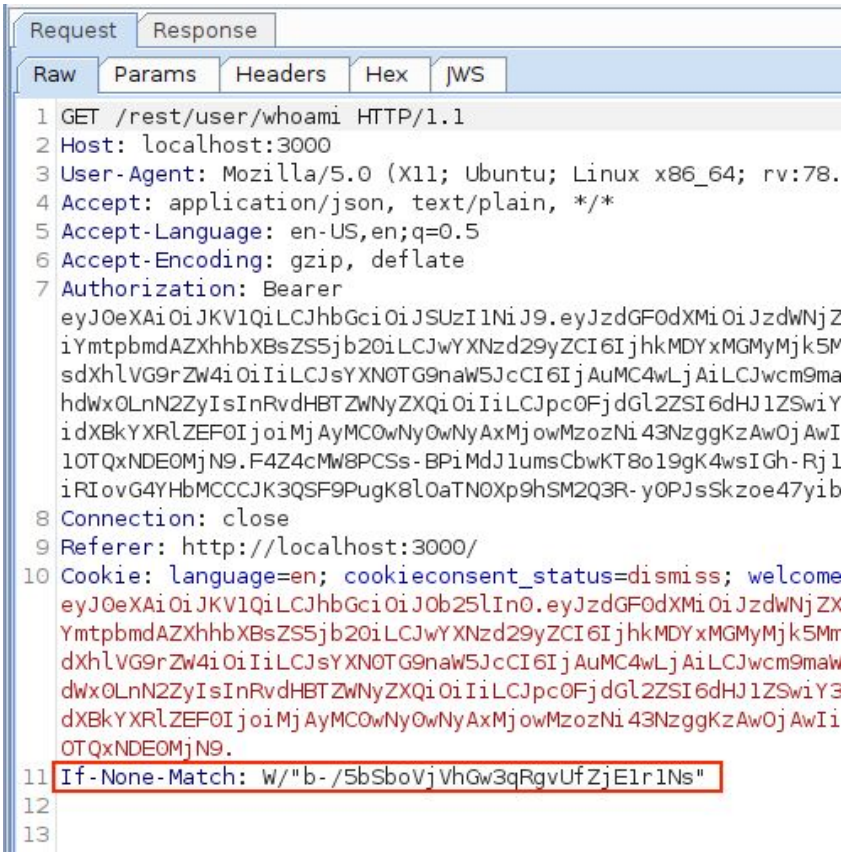


Some Changes, Some Not

Note: If you get HTTP 304 "Not Modified" responses, it's probably because you got a request with the "If-None-Match" request header. This is part of caching and efficiency - in this case it tells the server not to bother processing the request if it's the same as a previous one. Delete that header and try again.



JOSEPH Results Table: HTTP 304: Server Did Not Process Request

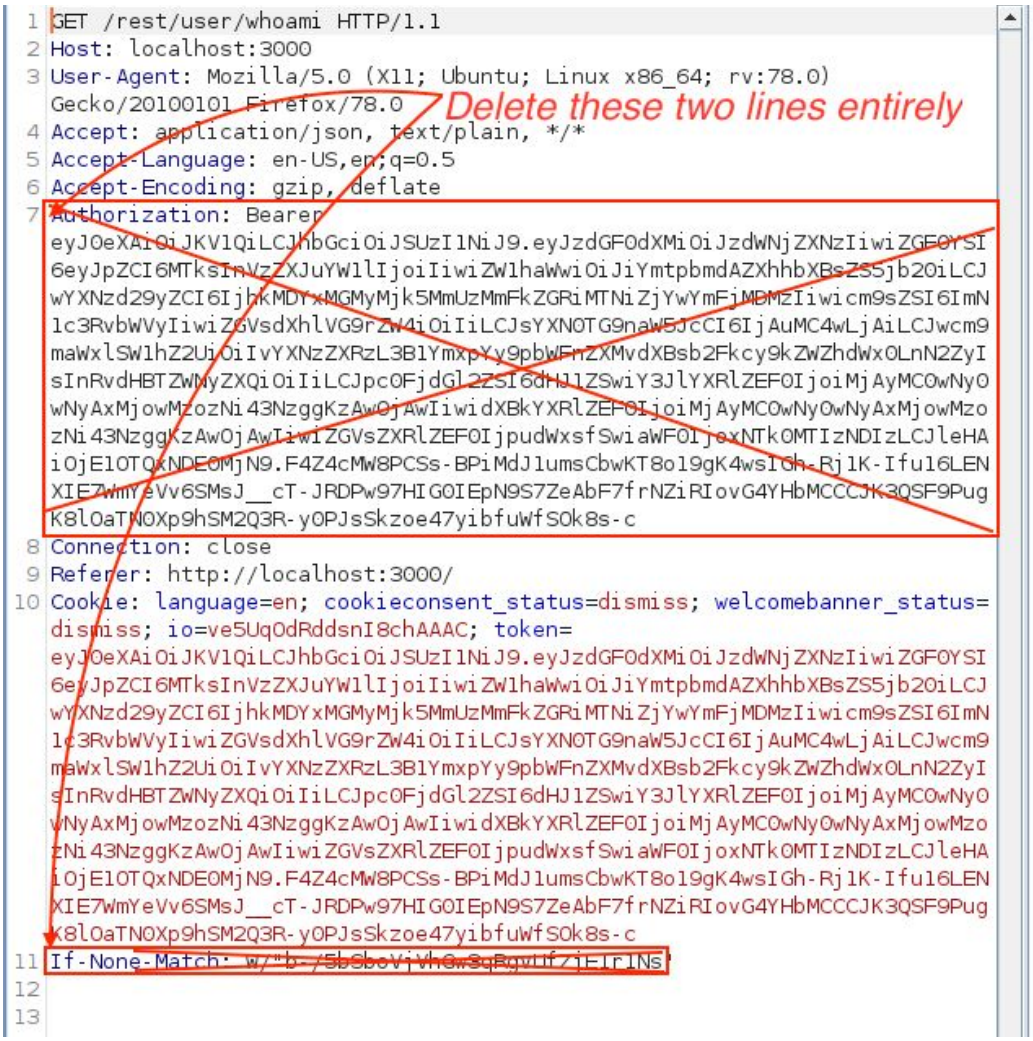


JOSEPH Results Table: If-None-Match Header was the Cause

5. Go back to Proxy History, find the same request and send it to Repeater this time.

6. In Repeater, send it once to make sure it still works (as usual). If you get HTTP 304, delete the In-None-Match header and try again.

7. Delete the "Authorization" header completely. Make sure that "If-None-Match" header is gone, too. Send the request again. If you get the same response, that means the information in the deleted headers wasn't needed.

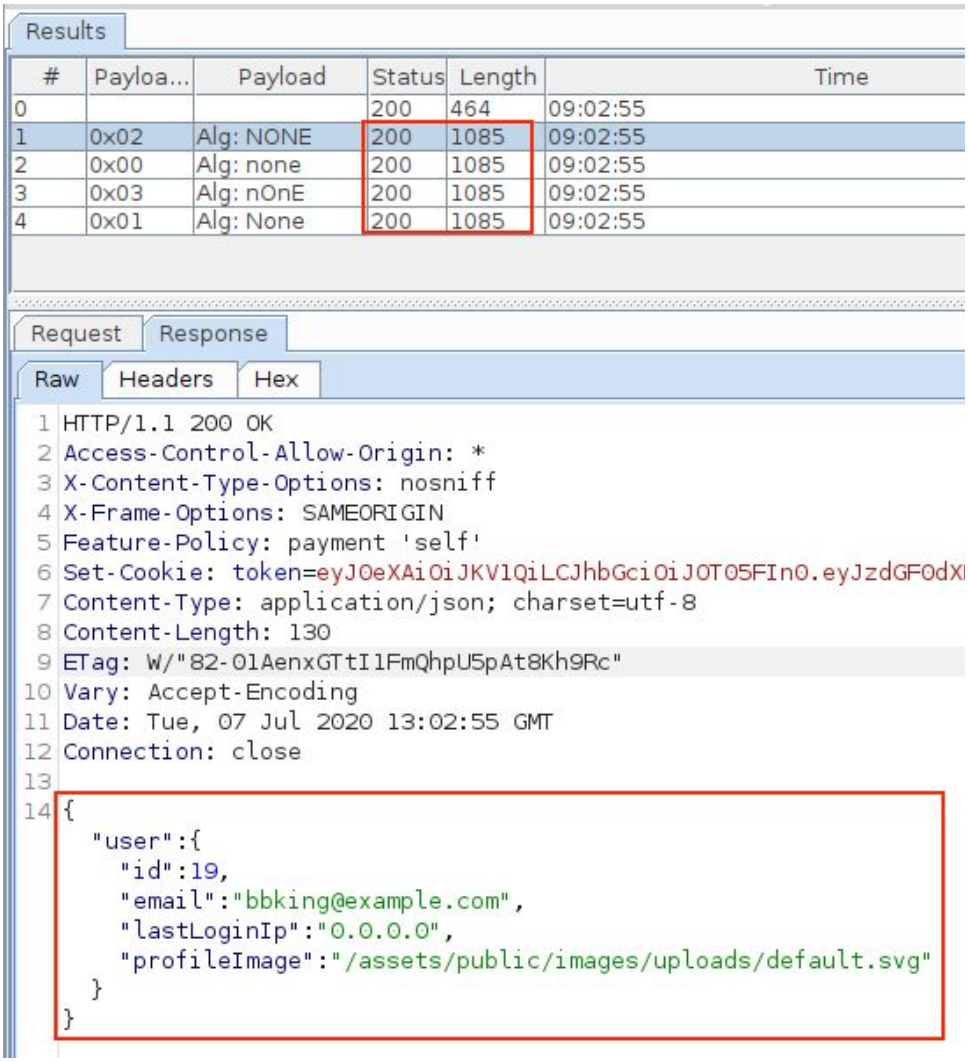


Delete the Authorization header and the If-None-Match header

Remember that the end of HTTP headers is indicated by a single blank line (i.e. two CR/LF in a row) so don't leave any space between headers, and make sure you have two line numbers at the end with no content on them.

8. Once the request has only one JWT and gets a successful response (HTTP 200 with your user details in the body) in Repeater, right click on it and send it to JOSEPH.

Choose "Signature Exclusion" as the attack, click "Load" then click "Attack". This time, you should see longer response bodies and your user's information inside them.



Successful Signature Exclusion Attack

9. Notice the JWT in each request of the attack has no signature (i.e. it ends at the second dot), and the extension tried four different ways of saying "none" : {NONE, none, nOnE, None}
Do you remember how "none" was spelled in the RFC?

6. Unsecured JWTs

To support use cases in which the JWT content is secured by a means other than a signature and/or encryption contained within the JWT (such as a signature on a data structure containing the JWT), JWTs MAY also be created without a signature or encryption. An Unsecured JWT is a JWS using the "alg" Header Parameter value "none" and with the empty string for its JWS Signature value, as defined in the JWA specification [JWA]; it is an Unsecured JWS with the JWT Claims Set as its JWS Payload.

Because the RFC has it as "none" (all lower-case), an obvious way to try to exploit an implementation is by varying the case. Maybe the developers followed the RFC as closely as they could, but their string parsing library (which they did not write) does case-insensitive matches. This is one *kind of issue* to look for when reading RFCs for vulnerabilities.

In Juice Shop, it doesn't matter. In other real-world implementations, it absolutely does.

OK. Back to the lab. You're halfway there! You've established that Juice Shop will accept an Unsecured JWT when the payload of the JWT is the same as a legitimate one. You have a theoretical attack. Now, to make it practical, try modifying the payload of the JWT.

This is where tools that help can also hurt. The JSON Web Token Attacker goes only this far: it tells you that you CAN forge a JWT, but it won't help you actually do the forgery.

The next section shows how to get the same results JOSEPH helped with, but by a different method. The "Followup" item below that shows how to exploit the condition.

Second: Same Attack, but Manually

You can do all of those same things in Burp without the JOSEPH extension. Doing it this way helps you build the skills you'll need for the follow-on attack.

To start this method, you need the normal request that was sent by the browser as your starting point. We'll use Burp Intruder and some more advanced payload rules to do the same kind of testing that JOSEPH did.

1. Log out of Juice Shop, then log back in. If you did the steps above, you may be logged in as the Admin user...

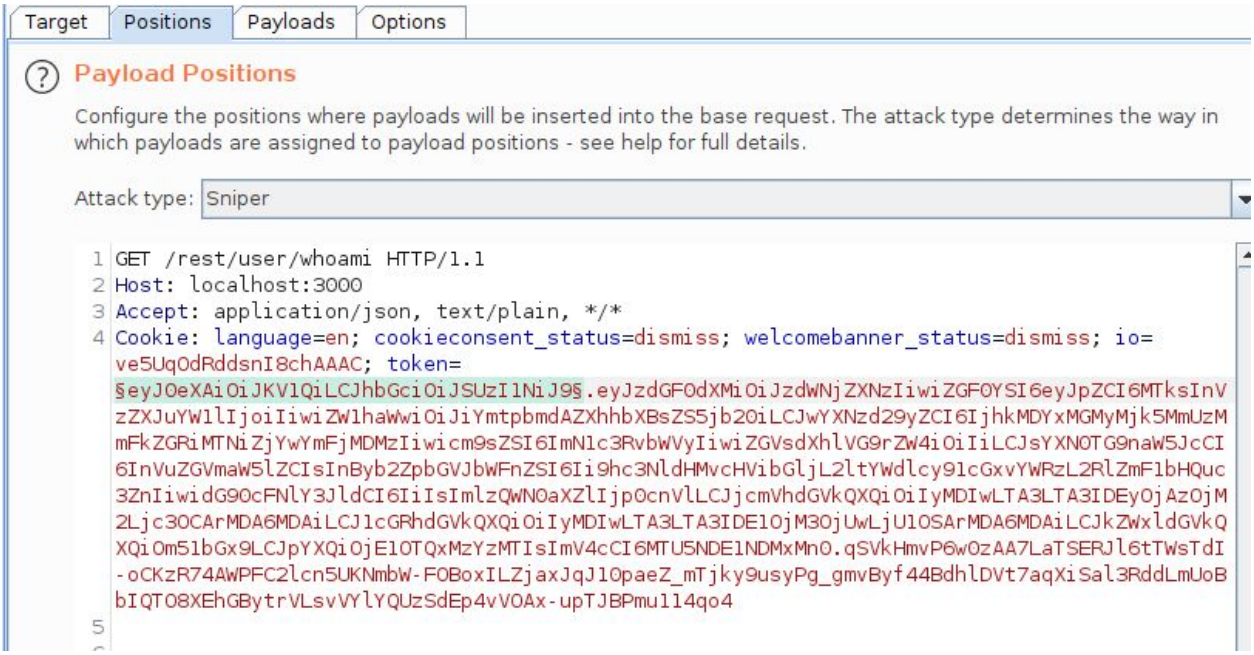
2. Find the request to /rest/user/whoami and send it to Repeater.

3. In Repeater, delete the "Authorization" header and send the request to make sure it still works. Double check the "email" in the response to be sure it's your user.

4. Right-click the request and choose "Send to Intruder"

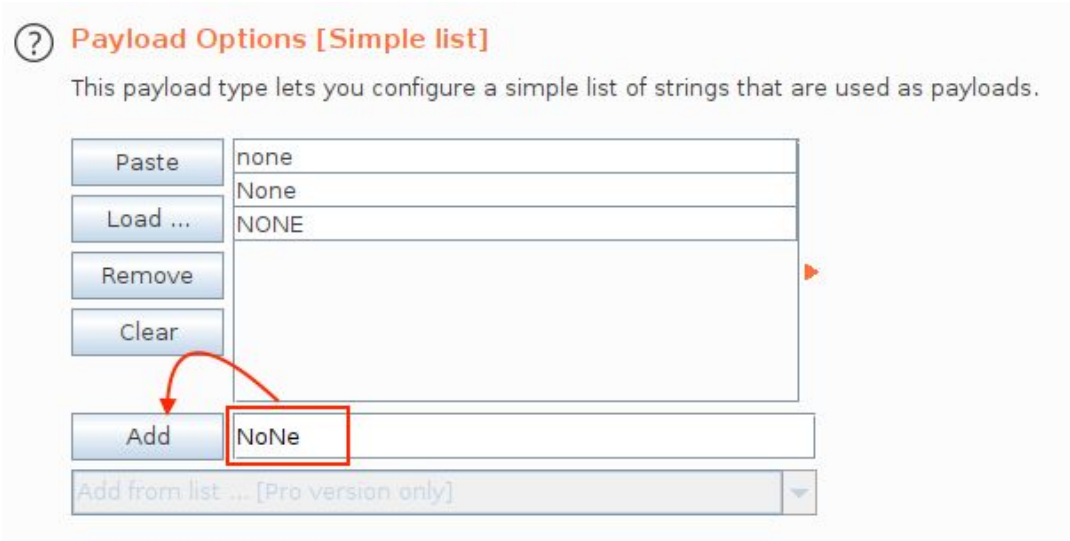
5. In Intruder's "Positions" tab, click "Clear" to remove the default insertion points.

6. In Intruder's "Positions" tab, double click on the JWT header to select it, then click "Add" to add the position markers. Your display should look something like this:



Intruder Setup

7. You want to send different versions of "none" in the JWT header. Go to Intruder's "Payloads" tab and enter the following payloads, clicking "Add" after each one:
none
None
NONE
NoNe



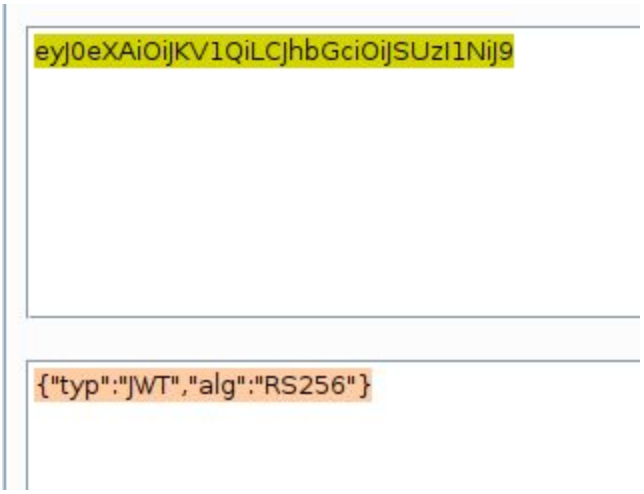
Payload Setup

...but if you leave things like this, your attempts will fail because the entire header would be replaced with a single word, which would make for a broken JWT.

8. In Intruder's "Positions" tab, double-click on the JWT header to select it, then copy it to your clipboard.

9. Go to Burp Decoder and paste the header there and choose "Decode ... Base64"

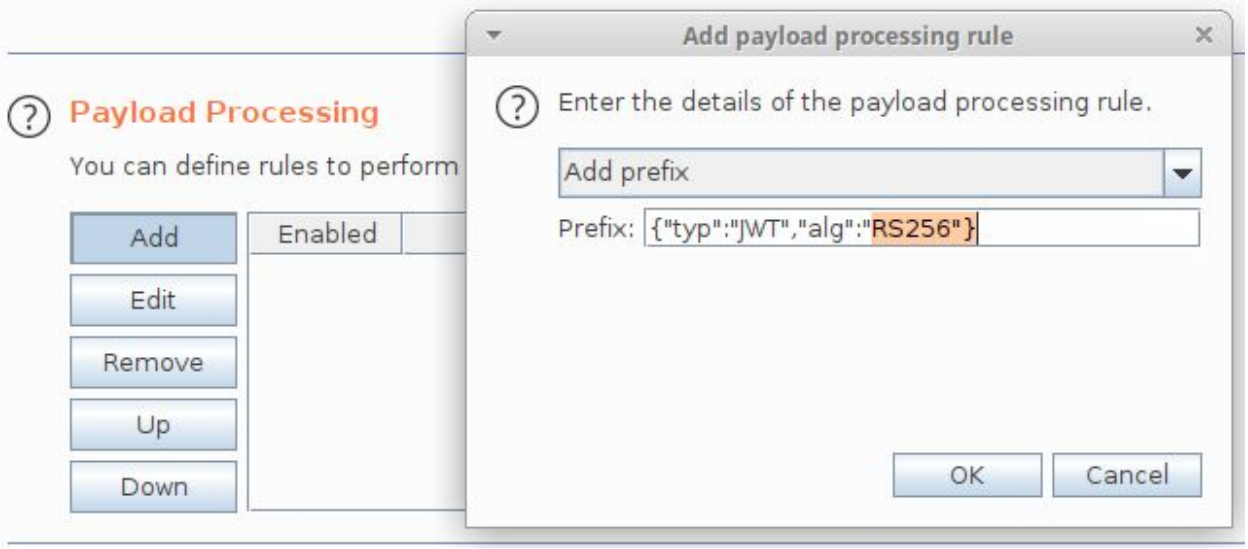
10. Select the entire decoded value and copy it to your clipboard.



Select the decoded header and copy it to your clipboard

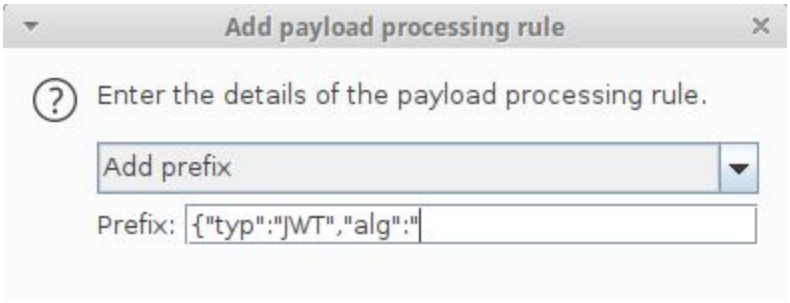
11. Return to Intruder's "Payloads" tab and click on "Add" in the "Payload Processing" section.

12. Choose "Add prefix" from the drop-down menu, then paste from your clipboard into the "Prefix" box and delete everything after the quote that's right before the legitimate algorithm's name. In the screenshot below, delete everything that's highlighted in orange.



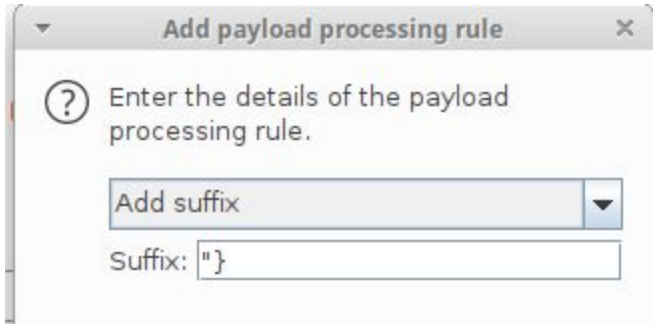
Delete from Start of Algorithm Name to End of String

When you're done, it should look like this:



Prefix Added.

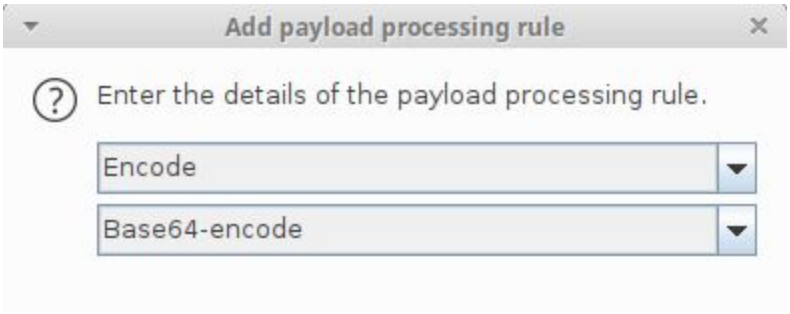
13. Click OK, then click "Add" to add another rule. Choose "Add suffix" as the rule type and paste from your clipboard into the text field again. This time, delete everything except the last two characters.



Suffix Rule

14. These two rules would take each payload in turn and add the prefix and suffix so as to make a valid JWT header sting. Now we need to Base64URL encode it. There is no "Base64URL Encoding" option, so we need to create one.

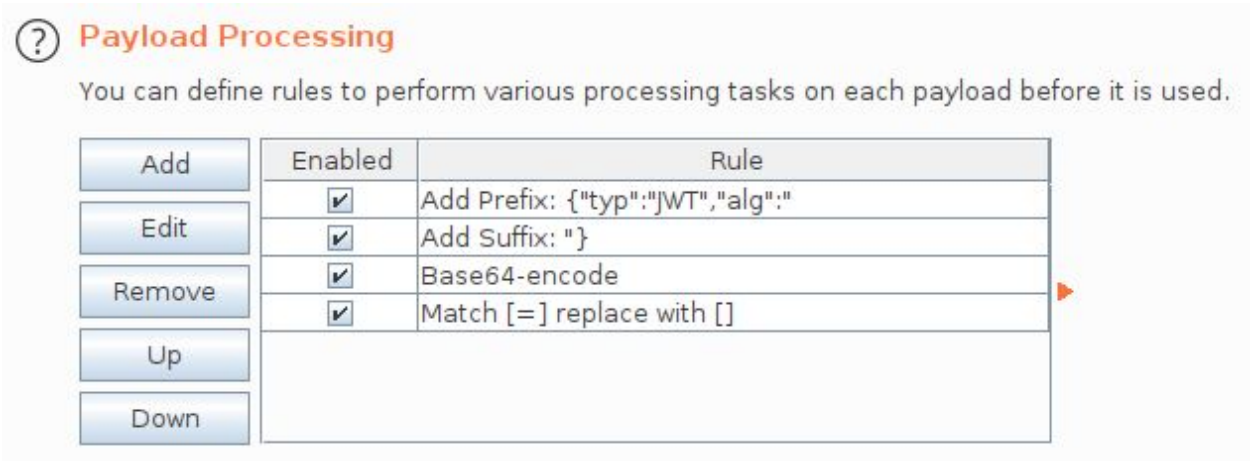
15. Click "Add" to create another rule. Choose "Encode" as the type of rule, and choose "Base64-encode" in the second drop-down.



Base64-Encoding Rule

16. The last rule needs to remove the equals sign, if it appears. Click "Add" again and choose the "Match/Replace" rule. Enter the equals sign in the "Match regex" field and leave the "replace with" field blank and click OK.

When you're done, your rules list should look like this:

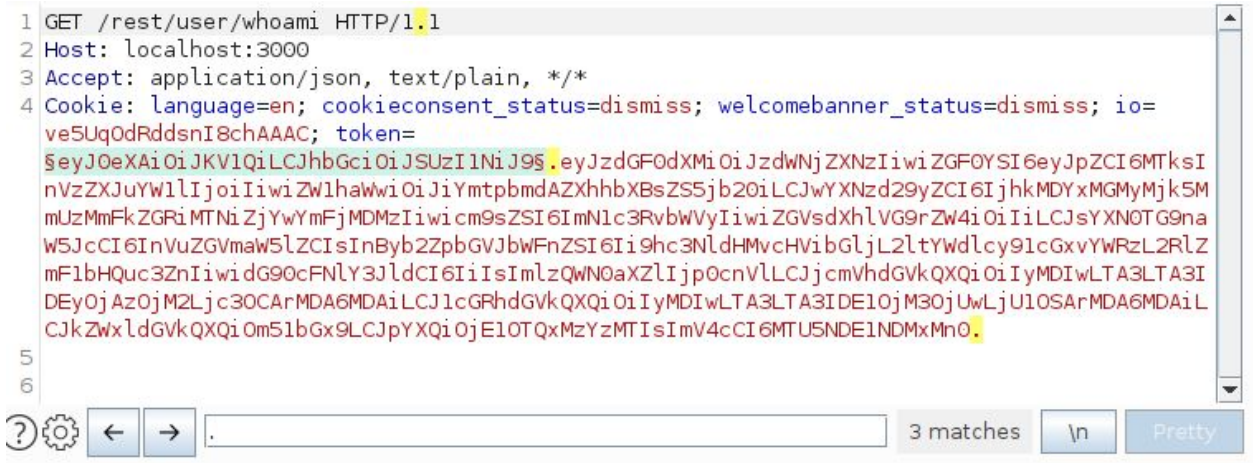


Payload Processing Rules Complete

17. One more thing: An unsecured JWT (that is, one that uses the "none" algorithm) cannot have a signature on it. Return to Intruder's "Positions" tab, and delete everything in the token cookie after the second dot. Use the "search" field at the bottom if you have trouble finding the dot.

Delete the Signature: All of the Orange. Keep the dot.

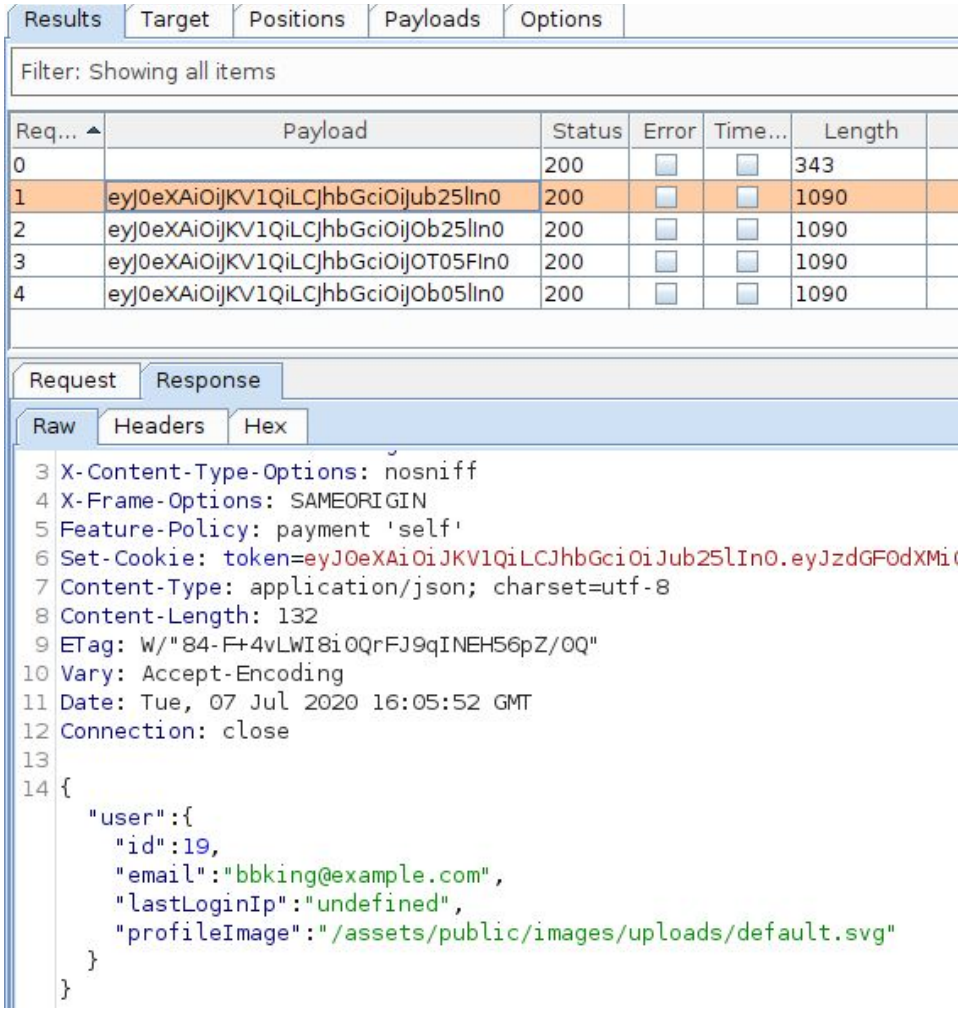
18. When you're done, the "Positions" tab should look something like this:



Ready to Attack

19. Click "Start Attack"

If you got all that right, the results table in Intruder should look a lot like the results table from the JOSEPH method earlier. The screenshot below shows what you might see.



Intruder Results

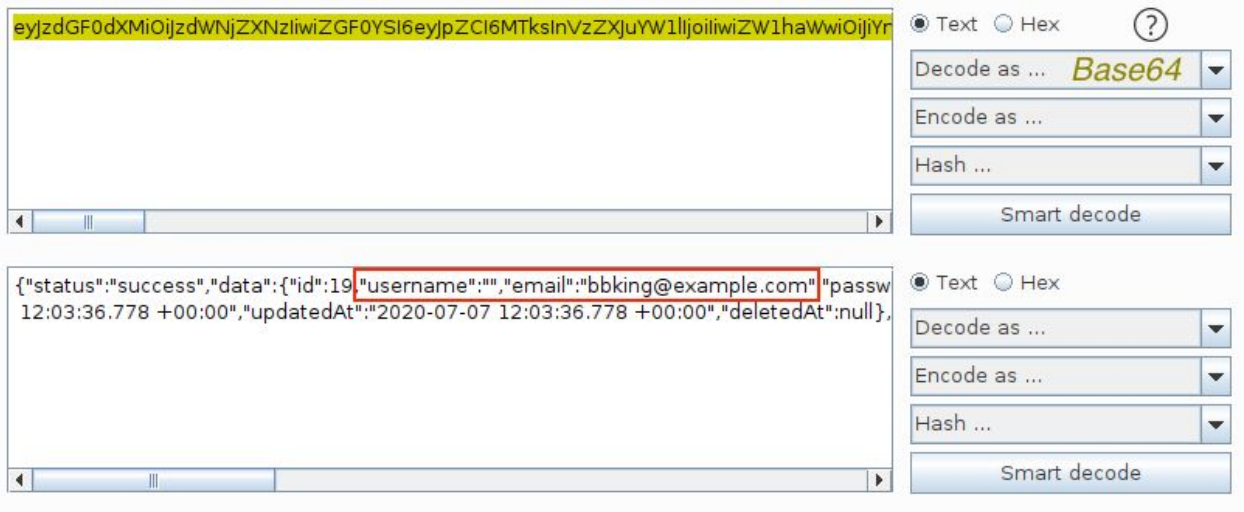
You didn't learn anything here that JOSEPH didn't already help you with, but now you have a general method to continue attacking and now you know exactly how to expose the general problem.

Instead of changing the header, for example, you could change parts of the payload now, testing for injection flaws or other common webapp flaws that the scanner would completely overlook. The next steps show how to do just that.

Followup: Forge a USEFUL New Token

20. Copy the base64 encoded header of the forged JWT from any of the successful JOSEPH requests. Paste it into a blank text file called "jwt-head.txt" and save that file.

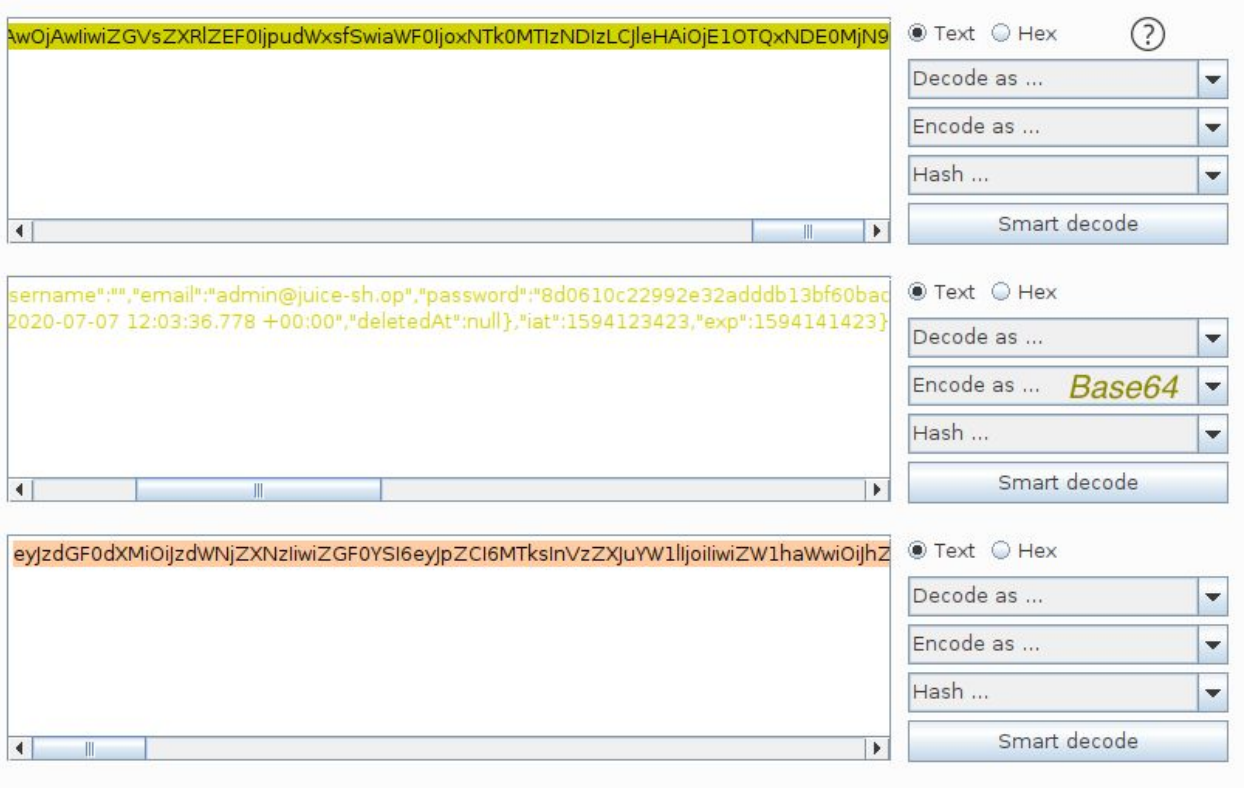
21. Copy the base64 encoded "payload" of the JWT - the part in between the two dots - and paste it into Burp Decoder. Then choose "Decode As ... Base64" and notice what you see in the decoding.



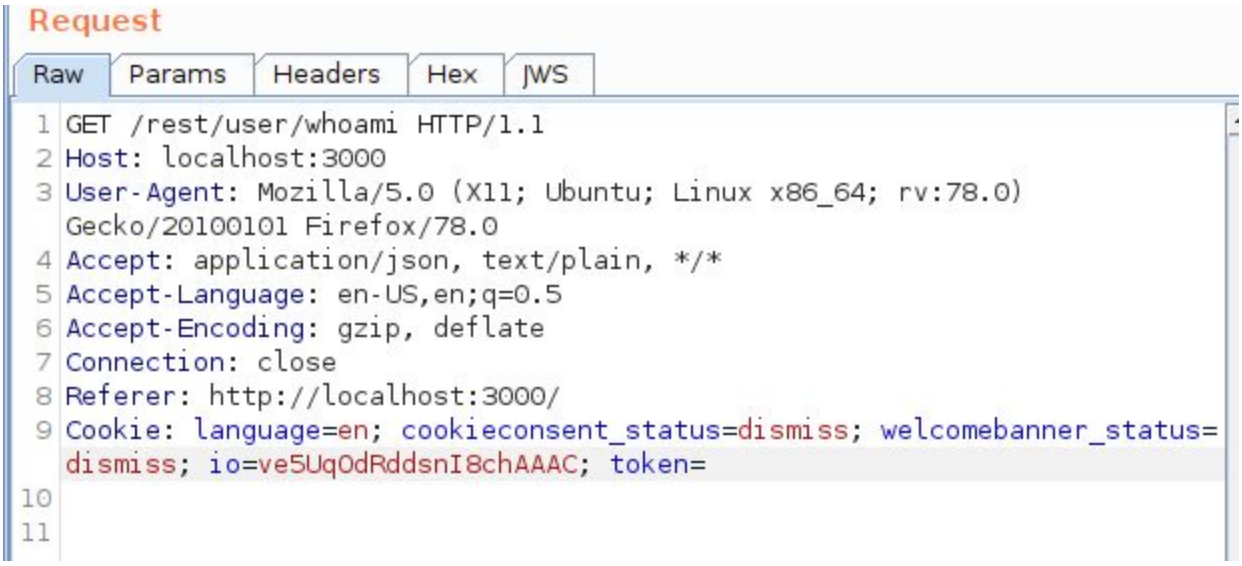
Email Address is in the Payload

Remember when we said not to trust user identifiers in user input? That's what we have here. Change the email address to be the administrator's email address: admin@juice-sh.op and then choose "Encode as ... Base64" from the menu next to it.

22. Select the entire contents of the bottom text area and copy it to your clipboard.

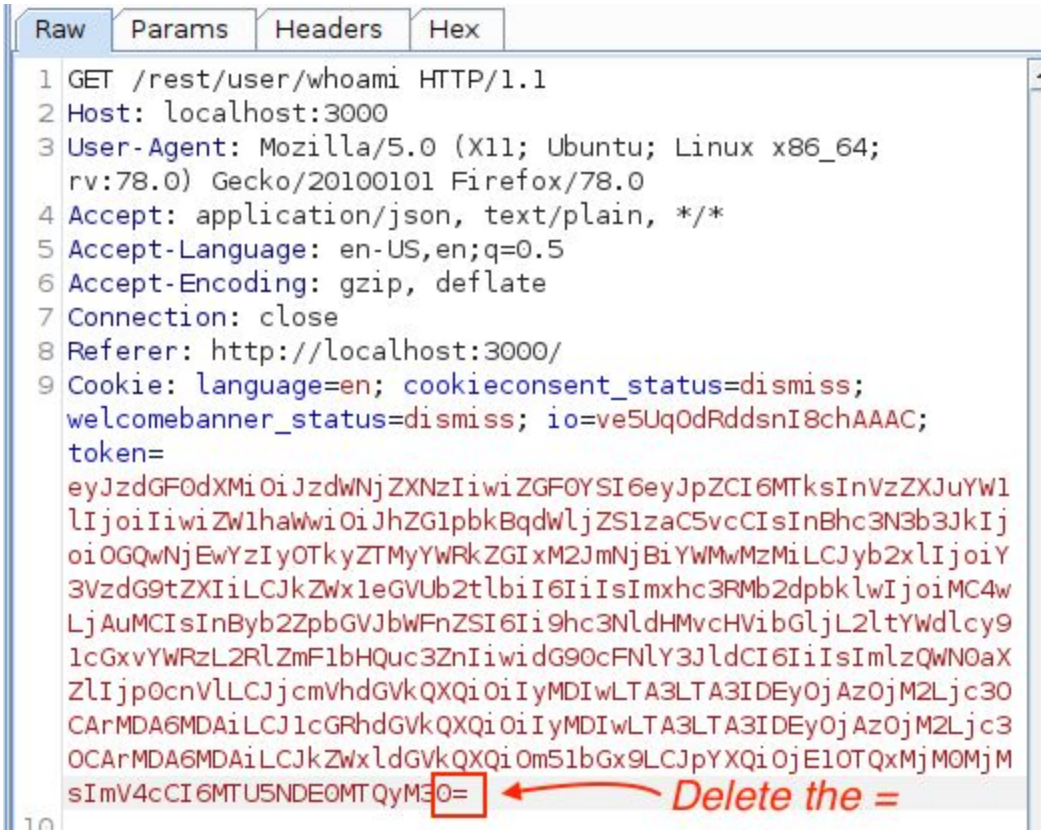


23. Return to your Repeater tab. Right-click on the request and choose "Send to Repeater" to get another copy of it (in case you want to return to the original one). In that copy, delete the JWT so you can replace it with your forged one. Be sure to leave the part that says "token="



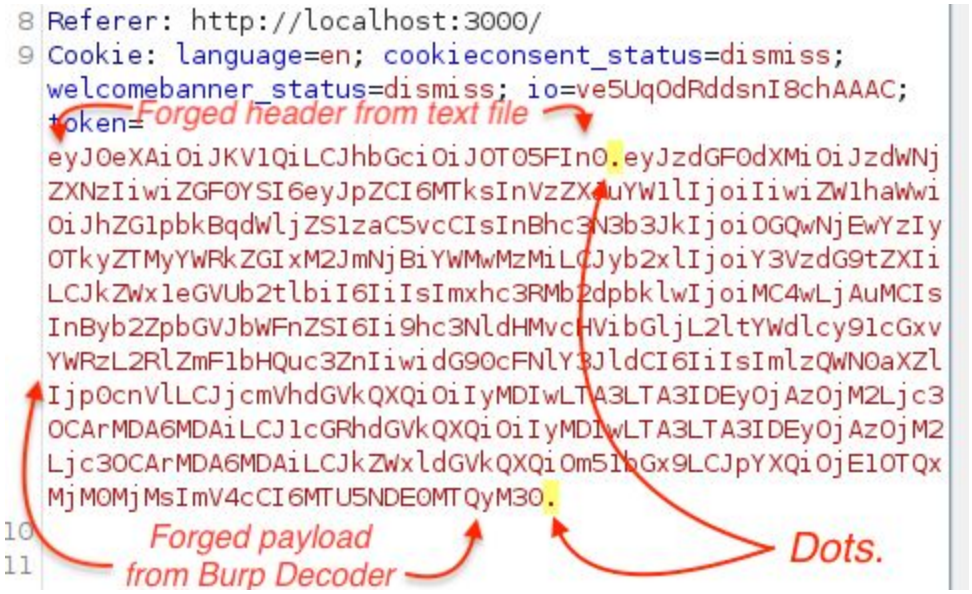
JWT Removed from Headers: Ready to Add Your Forged One

24. Paste the encoded "payload" of the JWT that you built in Burp Decoder right after "token="



JWT Payload Pasted. Delete the = Because we can't have that here

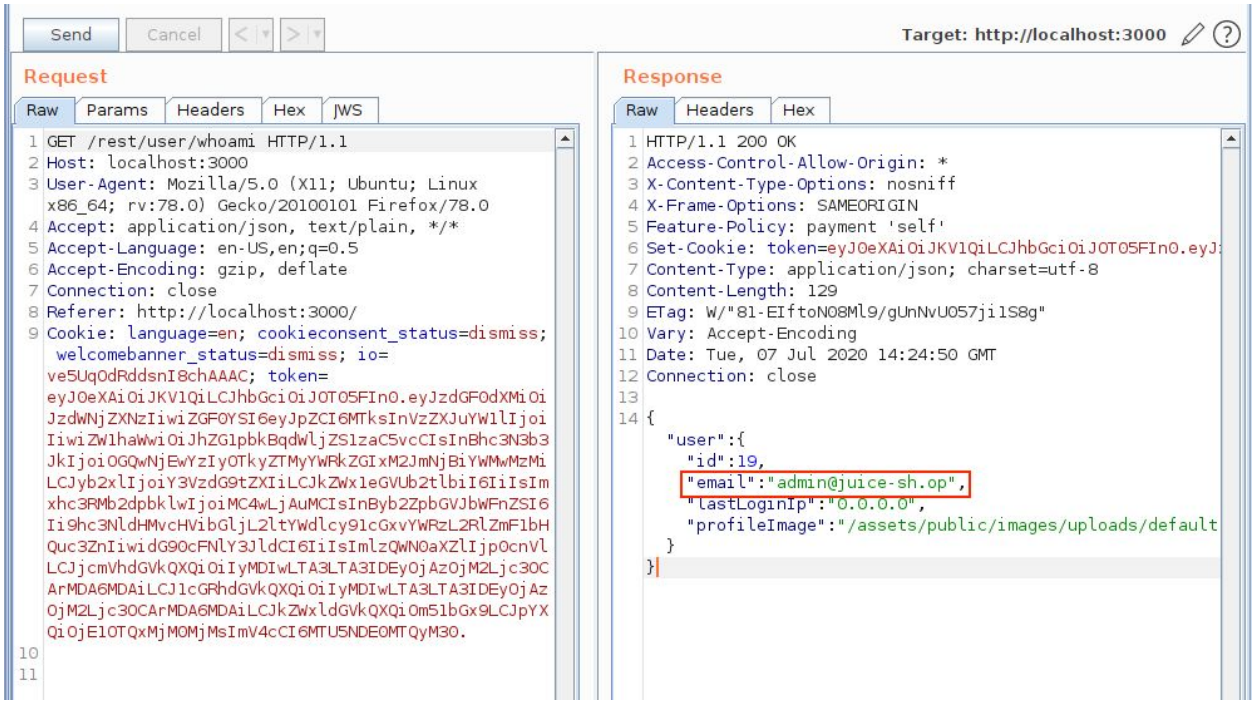
25. Go to your text editor and copy the JWT header you have there. We need to put this in the "token" cookie as the header, so it needs to come first, and it needs to have a dot between the encoded header and the encoded payload. When you're done, it should look something like this:



Forged JWT Header [dot] Forged JWT Payload [dot] no signature

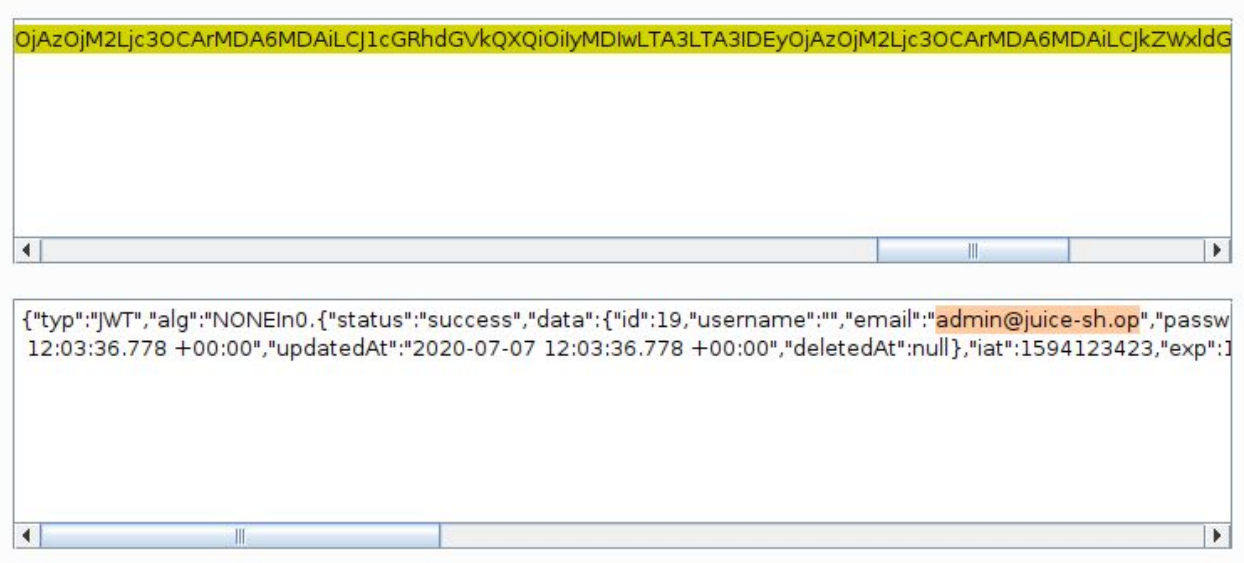
Paste after the Paste it into the Repeater "Request" area right after "token=" so it comes before the payload.like this:

26. Click "Send" in Repeater to send your forged token...



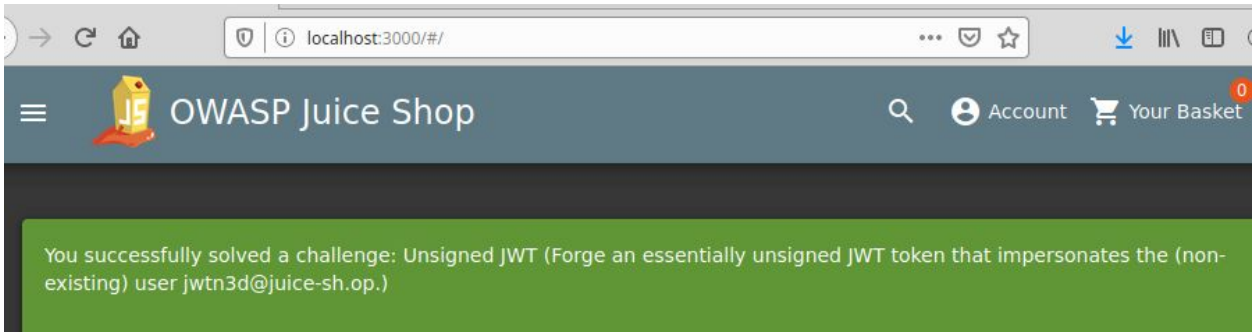
Juice Shop Believed You! You're Admin!

27. Notice the Set-Cookie header in the response. Copy that and paste it into Burp Decoder, then Base64-decode it.



Confirmed: Juice Shop Put "Admin" In Your Cookie

To get the green banner, Juice Shop wants to see an email address of "jwtn3d@juice-sh.op" in the JWT payload.



For Further Practice: Digi.Ninja labs

- Leaky JWT: https://authlab.digi.ninja/Leaky_JWT
- JWT None algorithm: https://authlab.digi.ninja/JWT_None
- Cracking JWT Keys: https://authlab.digi.ninja/JWT_Cracking
- JWT Signature Disclosure CVE-2019-7644: https://authlab.digi.ninja/Auth1