

SENG2250/6250 System and Network Security
School of Electrical Engineering and Computing
Semester 2, 2020

Lab 8: Distributed System Security

Objectives

- 1) Review Kerberos and OAuth2.0.
- 2) Learn Java socket programming.
- 3) Learn BigInteger class of Java for large number computation.

Part 1 Review Questions

1. What entities constitute a full-service Kerberos environment?
2. In the context of Kerberos, what is a realm?
3. Describe the message flow of Kerberos protocol version 4.
4. What are the principal differences between version 4 and version 5 Kerberos?
5. What are the two tickets generated in (intra-realm) Kerberos protocol version 5? How could they be different in usage? Can we reuse these tickets?
6. What are the principle differences between the intra-realm Kerberos and inter-realm Kerberos protocols?

Part 2 Exercises

7. **OAuth 2.0.** In this part, we will review the OAuth 2.0 web application flow and play with exercises to discover how it works in practice.

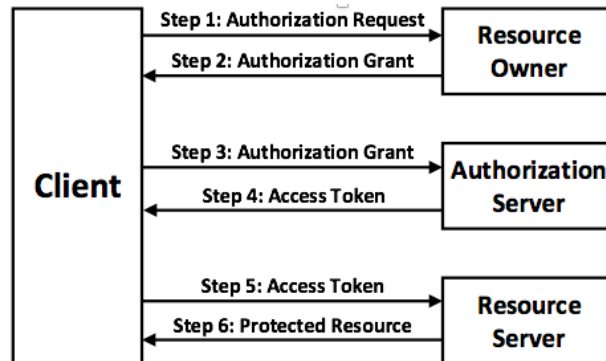


Figure 1 OAuth 2.0 protocol.

Step 0

Explain the OAuth steps based on Figure 1.

Step 1

Go to this link: <https://developers.google.com/oauthplayground/>

Step 2

Select “Contact v3” google contacts service (Figure 2), and then click Authorise API.

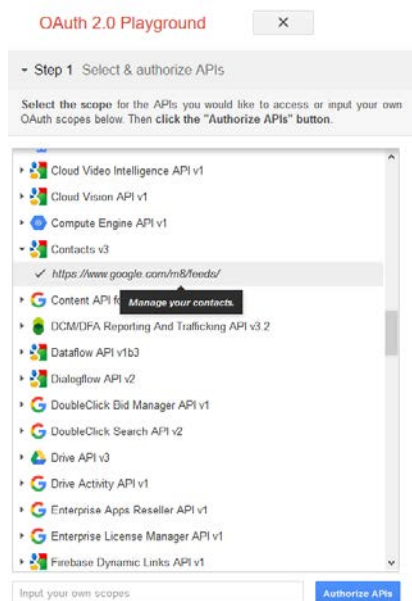


Figure 2

Google OAuth 2.0 Playground wants to access your Google Account

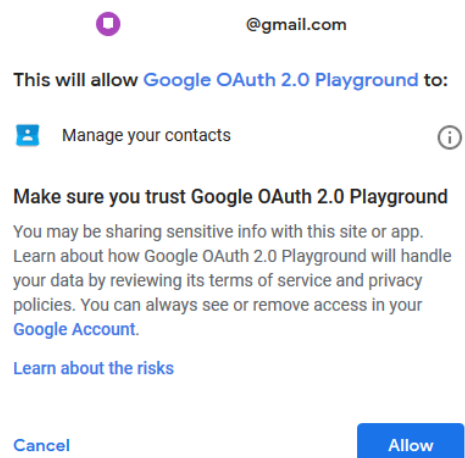


Figure 3

Step 3

Authorise using your Gmail account (Figure 3). If you do not have a Google account, you need to register a new one.

Step 4

Now, you should get the authorization code (Figure 4).

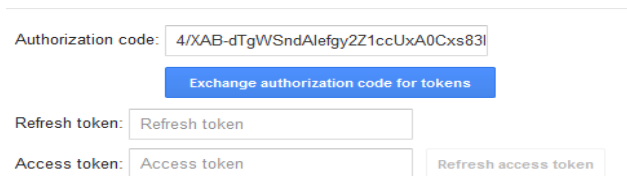


Figure 4 shows the OAuth 2.0 Playground interface. The 'Authorization code' field contains the code '4/XAB-dTgWSndAlefgy2Z1ccUxA0Cxs83l'. Below it is a blue button labeled 'Exchange authorization code for tokens'. The 'Refresh token' field is empty with a placeholder 'Refresh token'. The 'Access token' field is empty with a placeholder 'Access token'. There is a 'Refresh access token' button next to the access token field.

Figure 4

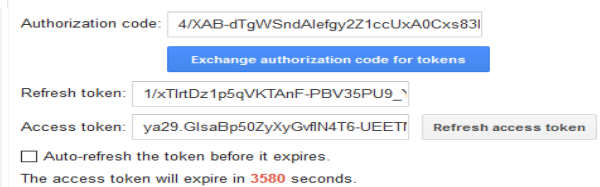


Figure 5 shows the OAuth 2.0 Playground interface after clicking the 'Exchange authorization code for tokens' button. The 'Authorization code' field still contains '4/XAB-dTgWSndAlefgy2Z1ccUxA0Cxs83l'. The 'Refresh token' field now contains '1/xTlntDz1p5qVKTAnF-PBV35PU9_'. The 'Access token' field contains 'ya29.GlsaBp50ZyXyGvfiN4T6-UEETI'. There is a 'Refresh access token' button next to the access token field. Below the access token field, there is a checkbox for 'Auto-refresh the token before it expires.' and a note: 'The access token will expire in 3580 seconds.'

Figure 5

Step 5

Exchange the Authorization Code to get the Access token (step 3 & 4 from Figure 1), by clicking **Exchange authorization code for tokens** button.

Step 6

Figure 5 shows the refresh, access token and access token validity time (about 1 hour).

Step 7

To access the resources, click *List of possible operations* -> any operation (e.g., List Contacts).

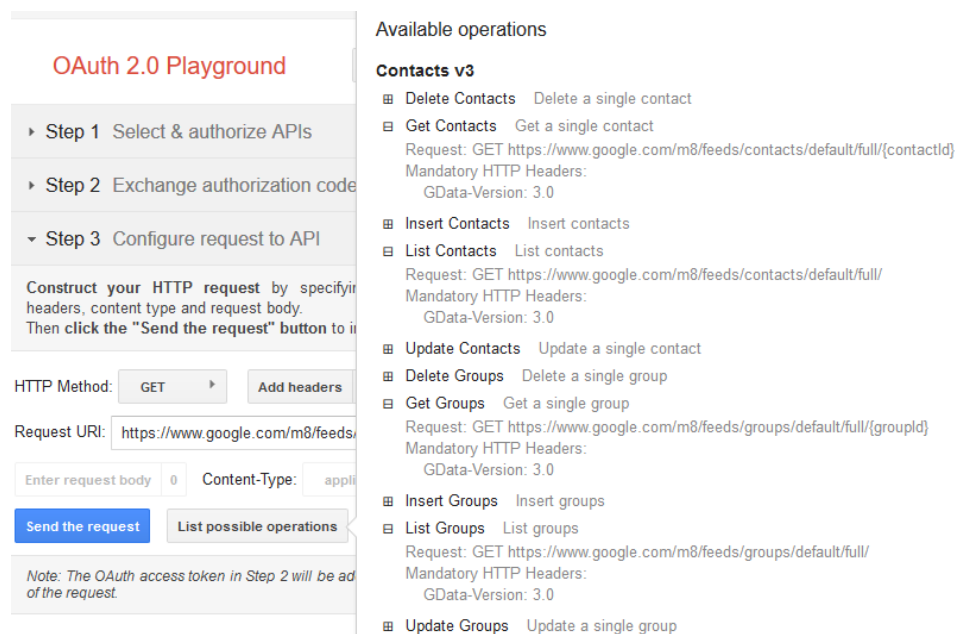
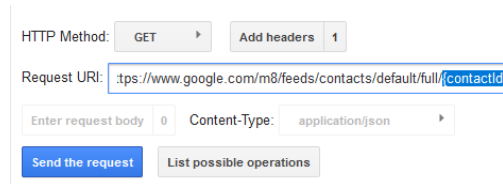


Figure 6 shows the OAuth 2.0 Playground interface. The left sidebar has a red header 'OAuth 2.0 Playground' and three steps: 'Step 1 Select & authorize APIs', 'Step 2 Exchange authorization code', and 'Step 3 Configure request to API'. Under 'Step 3', there is a text box: 'Construct your HTTP request by specifying headers, content type and request body. Then click the "Send the request" button to i'. Below this, the 'HTTP Method' is set to 'GET' and the 'Request URI' is 'https://www.google.com/m8/feeds/contacts/default/full/'. The 'Content-Type' is set to 'application/json'. There are buttons for 'Send the request' and 'List possible operations'. A note at the bottom says: 'Note: The OAuth access token in Step 2 will be ad of the request'. The right panel, titled 'Available operations', lists several operations under 'Contacts v3': 'Delete Contacts', 'Get Contacts', 'Insert Contacts', 'List Contacts', 'Update Contacts', 'Delete Groups', 'Get Groups', 'Insert Groups', 'List Groups', and 'Update Groups'. Each operation has a brief description and a request example.

Step 8

Click Send the request which equivalent to Step 5 & 6 in Figure 1. The Playground will list all the information using XML format. Read the response content and what's there?



HTTP Method: GET Add headers 1

Request URI: <https://www.google.com/m8/feeds/contacts/default/full/{contactId}>

Enter request body 0 Content-Type: application/json

Send the request List possible operations

Questions

1. Do I have to create an OAuth Access Token every time I need to access a resource?
2. What is the purpose of the refresh token?
3. Which type of token should a client use to access the resources?

8. Programming (Sockets).

- Self-study: <https://docs.oracle.com/javase/tutorial/networking/sockets/>
- Write a client/server program to exchange messages.
 - Either the client or server enters "EXIT", both the client and the server program will be terminated.

9. Programming (BigInteger). Self-study the basic programming using Java BigInteger package.

Tutorials and sample code for both are available from the following references.

- a. <https://www.concretepage.com/java/java-biginteger-tutorial-with-example>
 - b. https://www.tutorialspoint.com/java/math/java_math_biginteger.htm
 - c. <https://docs.oracle.com/javase/7/docs/api/java/math/BigInteger.html>
- Implement the Fast Modular Exponentiation using Java BigInteger. What is the result of the following setting (i.e $b^e \bmod n = ?$)
 $b=8257123890741038952193847189263512893749234792874892785928374913512409812735907213958072130958725098723509175098273429834$
 $e=65537$
 $n=98571091907190861258971207103948257309476190745375903875092387462039487623095847290386732894679348673489067389067346893746358710958158917598127349186591283471829569781249843$
 - Check the above result using the method `modPow()` of BigInteger.
 - How could you generate a 1024-bit random number?