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.

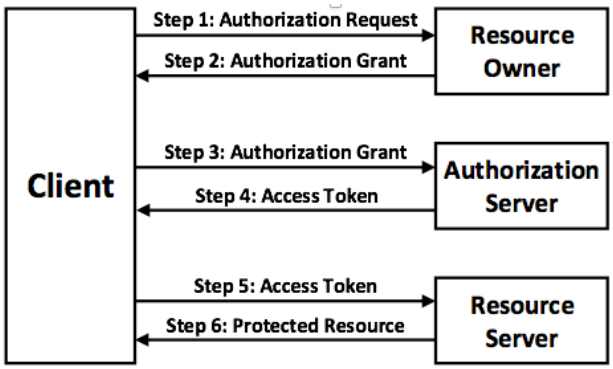


Figure 1 OAuth 2.0 protocol.

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.

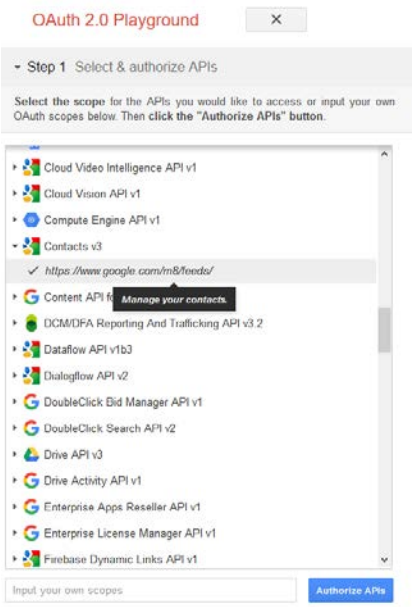


Figure 2

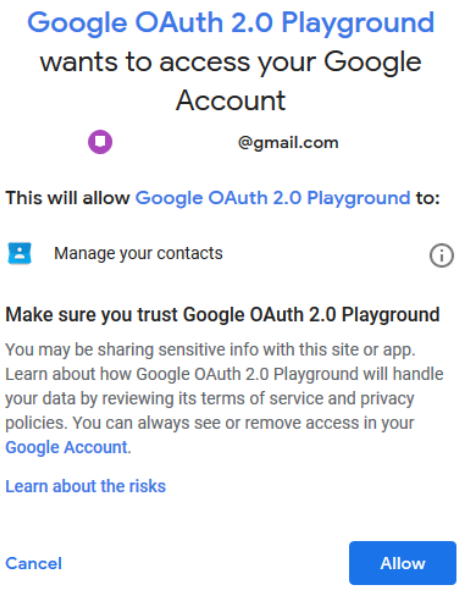


Figure 3

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).

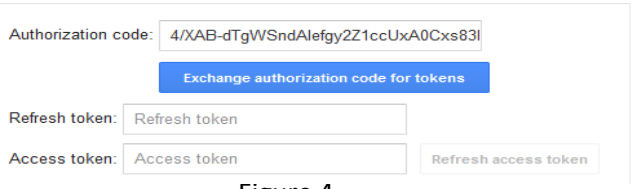


Figure 4

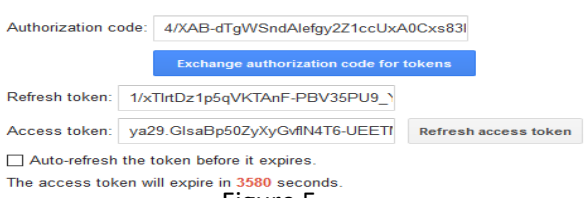


Figure 5

Step 5

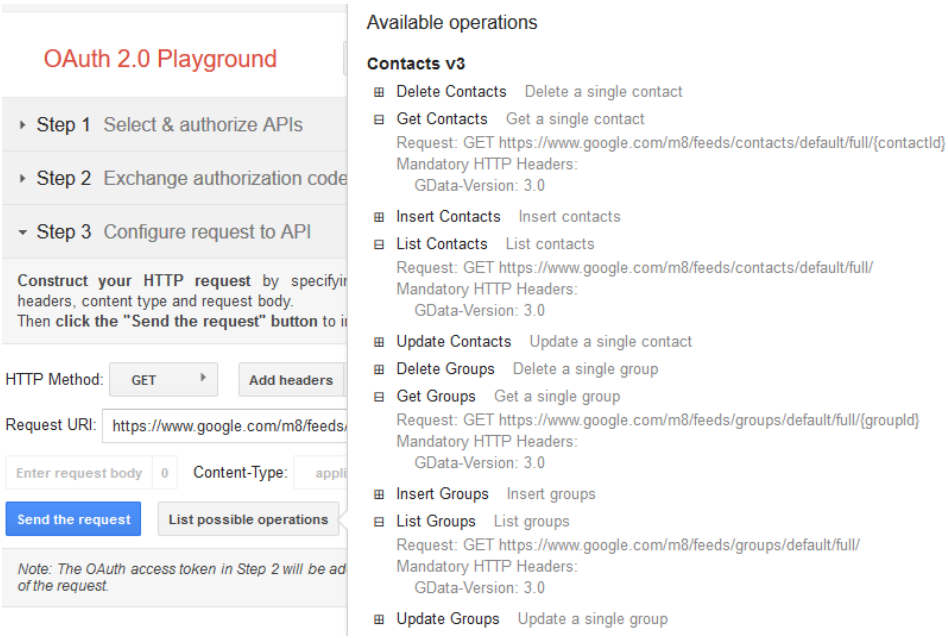
Exchange the Authorization Code to get the Access token (step 3 & 4 from Figure 1), by clicking **Exchange authorization code for tokens** button.

Step 6

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

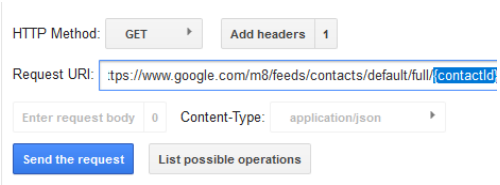
Step 7

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



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?



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 $b^e \ mod \ n =?$)
  b=825712389074103895219384718926351289374923479287489278592837491351
  24o981273590721395807213095872509872350917509827342983 4
  e=65537
  n=985710919071908612589712071039482573094761907453759038750923874620
  394876230958472903867328946793486734890673890673468937463587109581 58
  91759812734918659128347182956978124984 3
- Check the above result using the method modPow() of BigInteger.
- How could you generate a 1024-bit random number?