Description of final project

1. Message Digest

1)cd in message_digest dir.

2)Run Sha_md5.java

There are two mode

1->SHA

2->MD5

You could choose one mode. Then enter the string you want to hash. The program would hash it and print the result out.

Result: For example: I choose SHA algorithm to hash "sawxasx"

```
what methods du you want to choose?
1->SHA
2->MD5
1
please enter the string you want to hash
sawxasx
37be39fc8e09eab3820db145df59f34a
```

2. various_crypto_tech_A Authentication

1)cd in various_crypto_tech_A dir.

2)Run ProtectedServer.java

3)Run ProtectedClient.java

Result: The protectedServer.java program would print "Client logged in". Which means client is authenticated by server.

3. various_crypto_tech_B Signature

1)cd in various_crypto_tech_B dir.

2)Run ElGamalBob.java

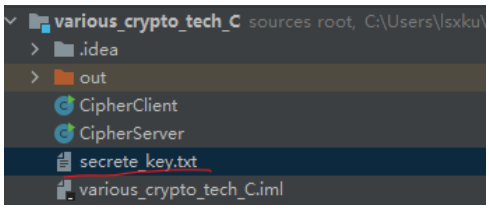
3)Run ElGamalAlice.java

Result: The ElGamalAlice.java program would print "The quick brown fox jumps over the lazy dog" and "Signature verified.".

4. various_crypto_tech_C Encryption

1)cd in various_crypto_tech_C dir.

2)Run CipherServer.java. It would create the "secrete_key.txt" file in"./various_crypto_tech_C". "secrete_key.txt" file contain the "DES" key.



3) run CipherClient.java.

Result: The CipherServer.java program would print:" The quick brown fox jumps over the lazy dog.", which means it already used "DES" key to decode the message the client sent to it.

5. various_crypto_tech_D Public-Key System

1) cd in various_crypto_tech_D dir.
2) Run RASGenerator.java. It would create four files.
   "AlicePri.txt" which contain Alice's private
   "AlicePub.txt" which contain Alice's public key
   "BobPri.txt" which contain Bob's private key
   "BobPub.txt" which contain Bob's public key
3) Run BobReceiver.java

4) Run AliceSender.java
   There are three messages.

```
String confidentiality = "used to test confidentiality";
String integrity = "used to test integrity";
String combination = "used to test confidentiality and integrity";
```

Confidentiality: it is encoded by Bob's public key. And it would be decoded by Bob's private key after Bob receive it.
Integrity: it is encoded by Alice's private key. And it would be decoded by Alice's public key after Bob receive it.
Combination: it is encoded by Alice private key firstly, and then encoded by Bob's public key.

Result: BobReceiver.java program would print:

```
used to test confidentiality
used to test integrity
used to test confidentiality and integrity
```

Which means Bob already decode the messages Alice sent.

6. various_crypto_tech_E X.509 certificates

1) cd in various_crypto_tech_E dir.
2)run Server.java
3)run Client.java

Result:

Client.java:

Print:

information of received certificate:[
[
  Version: V3
  Subject: CN=Shixiang Long, OU=pitt, O=pitt, L=pittsbuigh, ST=pittsburgh, C=US
  Signature Algorithm: SHA256withRSA, OID = 1.2.840.113549.1.1.11

  Key: Sun RSA public key, 2048 bits
  params: null
  modulus:
27783058976098814677552795332168739628029305892181324253977893344407154406681496075473334468115236104881745599173737702868329327299152207769424548925823689686595706602281089885256324163393009737764574641341182598189021856372867655635465397947621870738609238808023271501487219925621624936766523634801867021361408091631238975781635474280007063605922387118091676341060139854607200050426656325905055986731697248357451628707395660369316588582198325488513717780888756568224557310602752991000173660466916097682886998046446073458078637005756613119725160653413748681096941036230558665424168707498689341865205940136642246969187
  public exponent: 65537
  Validity: [From: Fri Apr 23 15:02:13 EDT 2021,
              To: Mon Apr 21 15:02:13 EDT 2031]
  Issuer: CN=Shixiang Long, OU=pitt, O=pitt, L=pittsbuigh, ST=pittsburgh, C=US
  SerialNumber: [2b478564]

Certificate Extensions: 2
[1]: ObjectId: 2.5.29.17 Criticality=false
SubjectAlternativeName [
  IPAddress: 127.0.0.1
]

[2]: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: 41 5A 60 27 5C D7 C0 BA   37 C9 34 8D 9E B8 64 55 AZ`'\...7.4...dU
0010: 39 13 BB 1C   9...
]
]

]
  Algorithm: [SHA256withRSA]
  Signature:
0000: 7B CB 19 08 1F 65 67 AF   BE 14 F8 7E 1D 4B CB 06 .....eg......K..
0010: 72 BF A0 7A F2 F2 2B 95   4A 02 73 F1 FD C8 4C 4E r..z..+.J.s...LN
0020: A2 AA A8 4C 73 BA 09 CA   E8 B7 EE 2B 76 98 25 B7 ...Ls......+v.%.

```
0030: 2E 79 AF D4 7D 9E D1 DE     1E D3 D7 CF 59 DB 8B 32    .y.....Y..2
0040: 71 9F 3B DA CF E4 C4 0E     E8 59 5D 70 29 BA 98 7E    q.;.....Y]p)...
0050: C1 9A 51 0F BD 18 35 B5     F9 95 56 C0 2D 11 CC 0C    ..Q...5...V.-...
0060: F7 D1 87 71 25 E9 C2 11     EC DC 0F 31 FE 3C 05 53    ...q%.....1.<.S
0070: 30 10 52 A4 52 15 0C D2     DB BD 42 F4 0A 9D 4F D3    0.R.R.....B...O.
0080: 58 8B 0F 4B A8 AB 63 EE     82 F7 45 00 E5 B5 A1 C0    X..K..c...E.....
0090: 37 F0 7F 5B 7E B3 3D C4     B5 B2 3C F5 90 EF A9 3A    7..[..=...<....:
00A0: C9 C1 64 A9 7C 89 C6 CC     DA E7 EE CD 85 96 37 93    ..d.....7.
00B0: 27 47 69 3F D5 98 B4 33     CD C7 64 89 12 14 FB 6E    'Gi?...3..d....n
00C0: 51 AB 86 25 45 0D 46 08     68 17 92 9B 22 FB D0 CD    Q..%E.F.h..."...
00D0: D9 8D 8B F2 2F AC 5A 26     C9 B6 EE 47 7D 45 29 A5    ..../.Z&...G.E).
00E0: 17 78 80 57 B9 7C EA FB     21 91 EF 2A 12 79 B3 5D    .x.W.....!..*.y.]
00F0: 01 E1 40 56 3C A9 FA 6D     8F DA 4B CC 7C FD E7 43    ..@V<..m..K....C

]
the Issuer is CN=Shixiang Long, OU=pitt, O=pitt, L=pittsbuigh, ST=pittsburgh, C=US
the Subject is CN=Shixiang Long, OU=pitt, O=pitt, L=pittsbuigh, ST=pittsburgh, C=US
the signature can be decoded by the public key. the signature validation is ok
the certificate is valid
```

Server.java:

Print: used to test confidentiality

What are the limitations of using self-signed certificates?

① It is risky, since no other third-party CA, like google CA, to guarantee it. The attacker is easy to attack it.

② It is not good for website owner. Because whenever a person browses your website, the browser would tell the person that your website is not safe. Therefore, some customer may go to another website but not yours.

What are self-signed certificates useful for?

① When one application only can be accessed in local network, self-signed certificates can be used. In this case, man-in-the-middle attack is very rare. We need not spend money on CA.

② A development server. In this situation, no real customer, therefore spending money on CA is needless.

③ Website which is only designed for small group people. In this case, the amount of customers is small, the possibility of attack is small. We need not spend money on CA.