ELGAMAL CRYPTOSYSTEM

Diffie-Hellman key generation and exchange

PROBLEM STATEMENT

You are required to build clients A and B that wish to send messages from B to A (only B to A) but encrypted using Elgamal Cryptosystem, but only after having exchanged messages that finally result in computation of one-time keys using the Diffie Hellman protocol.

The Elgamal Cryptosystem uses the parameters (q, a). These are known to each other using “other means”. To ensure that man-in-the-middle attack is not possible, the two clients A and B send initial messages to each other by adding to each message a MAC (or an “integrity check”) that itself is based on HMAC algorithm and uses a shared “secret”.

Once the one-time keys have been computed (or can be computed on the fly), B should encrypt and send messages

“hello 1”, “hello 2”, “hello 3”.

INTRODUCTION

ElGamal encryption is an public-key cryptosystem. It uses asymmetric key encryption for communicating between two parties and encrypting the message.

This cryptosystem is based on the difficulty of finding discrete logarithm in a cyclic group that is even if we know a^{XA} and a^{XB}, it is extremely difficult to compute a^{XA*XB}.

Key : $a^{XA*XB} \bmod q$

Encryption : C = Key * M

Decryption : M = C * Inverse(Key)

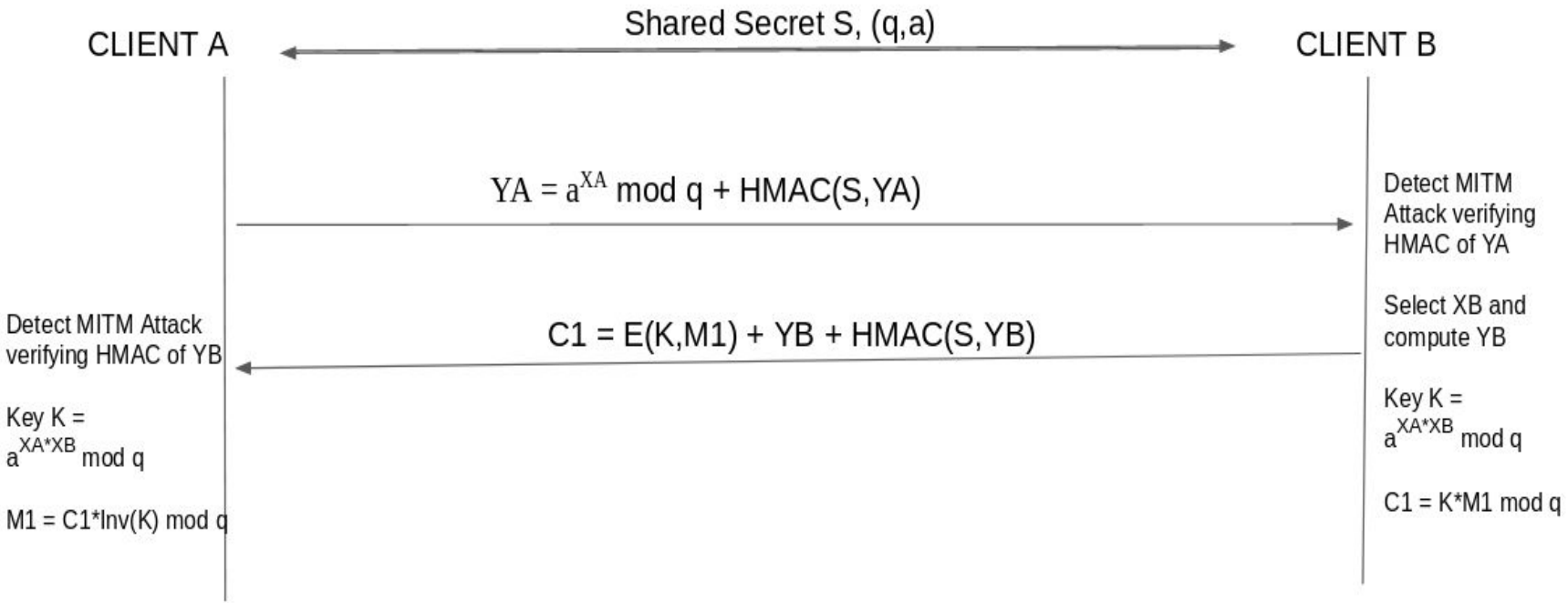
SYSTEM SPECIFICATION

Language used : Python

Assumptions

- ensure that clients already (somehow) know the shared “secret” to be used with HMAC
- assume that Diffie‑Hellman parameters (q, a) are already fixed, and known to them
- (once they know how to generate or compute one‑time passwords) messages sent from B to A are encrypted using Elgamal cryptosystem with parameters, (q, a).

SYSTEM DESIGN



IMPLEMENTATION

```
meh@meh-Len:~/Documents/nss/ass4$ python clientB.py
secret  3a9402ce5a7b14c4312ace99fc517cb70345332fd37deee7635fd245f6a19418
q  624274171418229804178293335295243686201126898337 82
alpha  695014620412076759967480322061432183457765550 7903
Message received from A 373974408443907134466764849515555194772735512906 81||350ec89638593ae488f0caeec3d666345c066abc1a68884ee44fb3c
b4b7016ea
a^XA mod q =  373974408443907134466764849515555194772735512906 81


Enter q to quit


a^XB mod q =  385862955104276110245906108620987324696230387685
a^(XA*XB) mod q =  277595510278023226250548859421947085416583244336 39
B: Hello1
Message sent by B (original):  Hello1||385862955104276110245906108620987324696230387685||48db62443db270ee13b758818877b5485d910001e3
91c3f4b40895470f97c528
Message sent by B (encrypted):  199868767400176722900395178783801901499939935922200 8#280371465380803458513054348016166556270749076 7
797539#299803151100265084350592768175702852249909903883301 2#299803151100265084350592768175702852249909903883301 2#308131016408605781
138109233958361264812407401213392 9#136021800036231380862768941116754071854125789724831 1||385862955104276110245906108620987324696230
387685||48db62443db270ee13b758818877b5485d910001e391c3f4b40895470f97c528


a^XB mod q =  171673779109455843854025022706141673211172635225 09
a^(XA*XB) mod q =  208769451145514455387441387762956553131871200153 61
B: Hello2
Message sent by B (original):  Hello2||171673779109455843854025022706141673211172635225 09||d6b289caaa8f787252ed30ef4013aca7b1e73c20
d7d5c21f904c1835f1f2a442
Message sent by B (encrypted):  150314004824770407878957799189328718254947264110599 2#210857145656969599941315801640586118663189912 1
551461#225471007237155611818436698783993077382420896165898 8#225471007237155611818436698783993077382420896165898 8#231734090771521045
480059940416881773976377032170507 1#104384725572757227693720693881478276565935600076805 0||171673779109455843854025022706141673211172
63522509||d6b289caaa8f787252ed30ef4013aca7b1e73c20d7d5c21f904c1835f1f2a442
```

```
meh@meh-Len:~/Documents/nss/ass4$ python clientA.py
secret  3a9402ce5a7b14c4312ace99fc517cb70345332fd37deee7635fd245f6a19418
q  624274171418229804178293335295243686201126898337 82
alpha  695014620412076759967480322061432183457765550 7903
a^XA mod q =  373974408443907134466764849515555194772735512906 81
Message sent to B 373974408443907134466764849515555194772735512906 81||350ec89638593ae488f0caeec3d666345c066abc1a68884ee44fb3cb4b701
6ea


Message received from B 199868767400176722900395178783801901499939935922200 8#280371465380803458513054348016166556270749076779753 9#2
998031511002650843505927681757028522499099038833012#29980315110026508435059276817570285224990990388330 12#308131016408605781138109 23
395836126481240740121339 29#136021800036231380862768941116754071854125789724831 1||385862955104276110245906108620987324696230387685||
48db62443db270ee13b758818877b5485d910001e391c3f4b40895470f97c528
a^XB mod q =  385862955104276110245906108620987324696230387685
YB  385862955104276110245906108620987324696230387685
a^(XA*XB) mod q =  277595510278023226250548859421947085416583244336 39
Message received from B (decrypted):  Hello1


Message received from B 150314004824770407878957799189328718254947264110599 2#210857145656969599941315801640586118663189912155146 1#2
254710072371556118184366987839930773824208961658988#225471007237155611818436698783993077382420896165898 8#231734090771521045480059 94
041688177397637703217050 71#104384725572757227693720693881478276565935600076805 0||171673779109455843854025022706141673211172635225 09
||d6b289caaa8f787252ed30ef4013aca7b1e73c20d7d5c21f904c1835f1f2a442
a^XB mod q =  171673779109455843854025022706141673211172635225 09
YB  171673779109455843854025022706141673211172635225 09
a^(XA*XB) mod q =  208769451145514455387441387762956553131871200153 61
Message received from B (decrypted):  Hello2
```

THANKYOU