System and Network Security – Assignment 3

Harrison Rebesco C3237487

Q1. Network Security

a. ***Confidentiality:*** In the scenario provided, an adversary has the potential to steal private information and/or data by intercepting communications between the client(s) and the bank system – thus compromising the confidentiality of the system. To counter this interception attack, the banking system can employ the use of encrypted messages and/or web proxies to protect communications between client and server, eliminating the potential eavesdropping threat.

Authentication: Another potential threat the banking system is vulnerable to is an impersonation attack, where the adversary poses as an employee or client to exploit the system to their advantage. An authentication breach to a banking system could be devastating, harming many users. Therefore, cryptographic techniques – such as key exchanges, ciphers, etc. – should be utilized as a countermeasure to this authentication attack.

b. Proper use of security-related processes ensures essential security services – such as; Authentication, Access Control, Data Confidentiality, Data Integrity and Non-repudiation – are provided for the customer, thus protecting their transaction.

Login: In order to log in, the customer is asked for their password. This acts as a unique identifier for the customer, and if the password is correct, they are authorized access to the account – the Login process achieves Authorization and Access Control as it ensures a trusted entity accesses the account.

Peer Entity Authentication: Once the customer has entered their password, peer entity authentication – such as a handshake – commences. This stage employs encipherment algorithms to establish a secure method to exchange data between the customer and the server, while simultaneously providing the customer a digital signature. This is used throughout the transaction to verify the identity of the customer, further Authenticating the customer. This also provides Non-repudiation as the digital signature protects the identity of the customer against impersonation attacks.

Navigate Account: As the customer navigates their account – selecting the amount to send, selecting who to send it to, etc. – the server logs time stamps, and tracks access duration. This information is used to provide a limited form of protection against replay attacks, and also facilitates the use of cryptographic chaining. These mechanisms provide Data Integrity and Data Confidentiality by protecting communications between the customer and the server.

Complete Transaction: The same mechanisms listed above – such as digital signatures, timestamps, encipherment algorithms, cryptographic chaining, etc. – are still being used, constantly verifying the identity of the customer and fortifying the integrity of the data being sent. Further checks, such as security labels may be used, to provide additional Access Control services.

c. The Kerberos protocol uses an internal Key Distribution Centre composed of an Authentication Server and Service Server. The other protocols, SAML and OAuth are dependent on external parties, as they both query Identify Providers when validating employees.

The utilization of a Key Distribution Centre means that Kerberos can independently authenticate and authorize employees internally. For these reasons, Kerberos is the best protocol to employ for this scenario.

d. The main difference between each of the networking protocols listed is the layer on which they operate.

SHH: works on the application layer – its main appeal is its low cost to set up, and its easy integration between TCP based applications. However, SSH is not designed to work on a network layer, therefore, heavy overhead is required before it can handle a connection with meaningful traffic.

TLS/SSL: works on the transport layer – and is very popular in web and e-commerce as it provides session-oriented security. It is integrated within applications to provide server/client authentication, meaning, that in order to use TLS/SSL an application must have built in support.

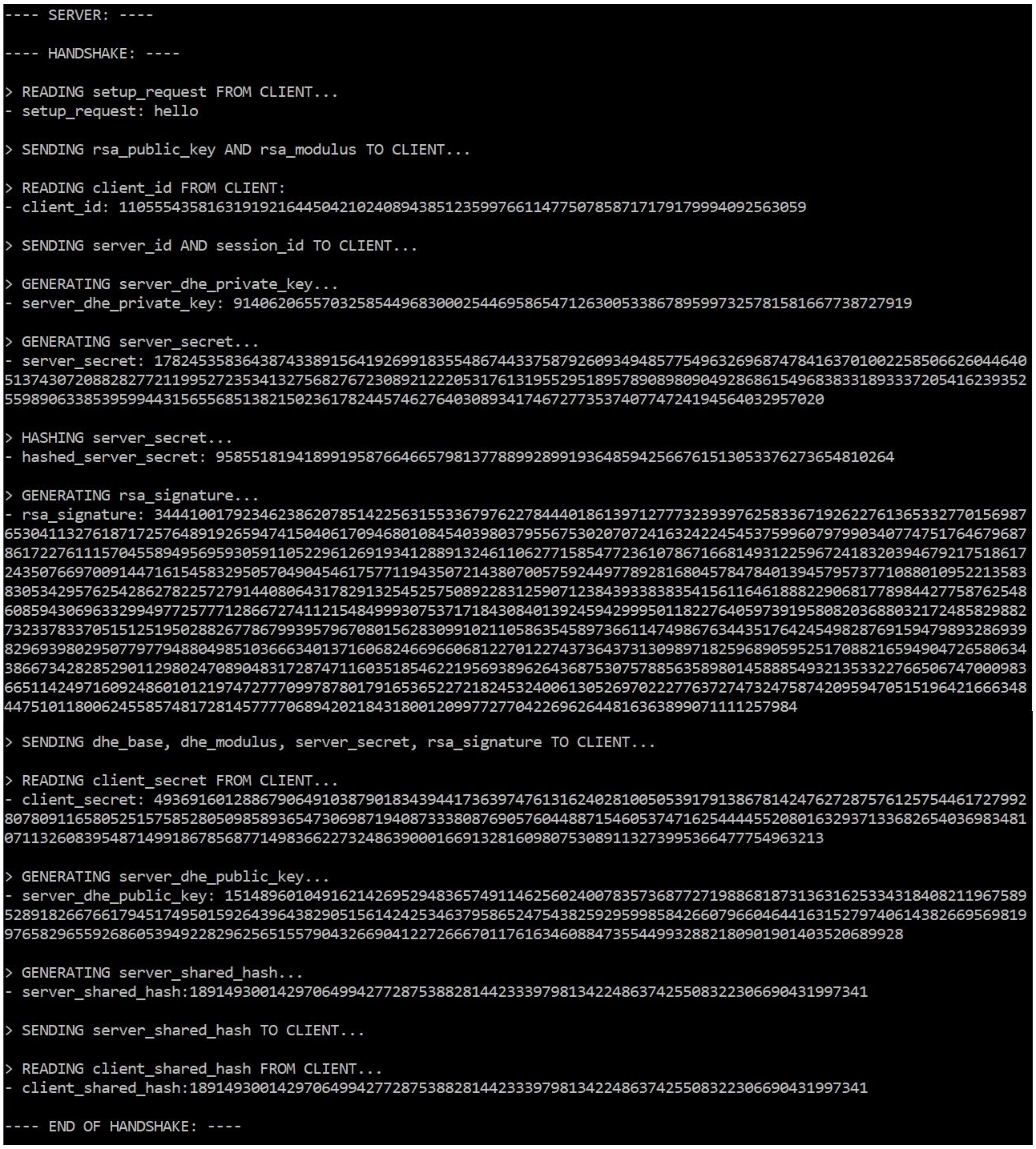
IPsec: operates on the network layer – meaning it can be fully incorporated into firewalls, routers, VPN concentrators, etc. providing direct security between client and the server. This makes IPsec the most efficient and secure protocol listed.

Upon analysing the protocols listed, I have determined that IPsec would be the most suitable option to provide a secure connection service for travelling employees.

Q2. Programming Task Screenshots

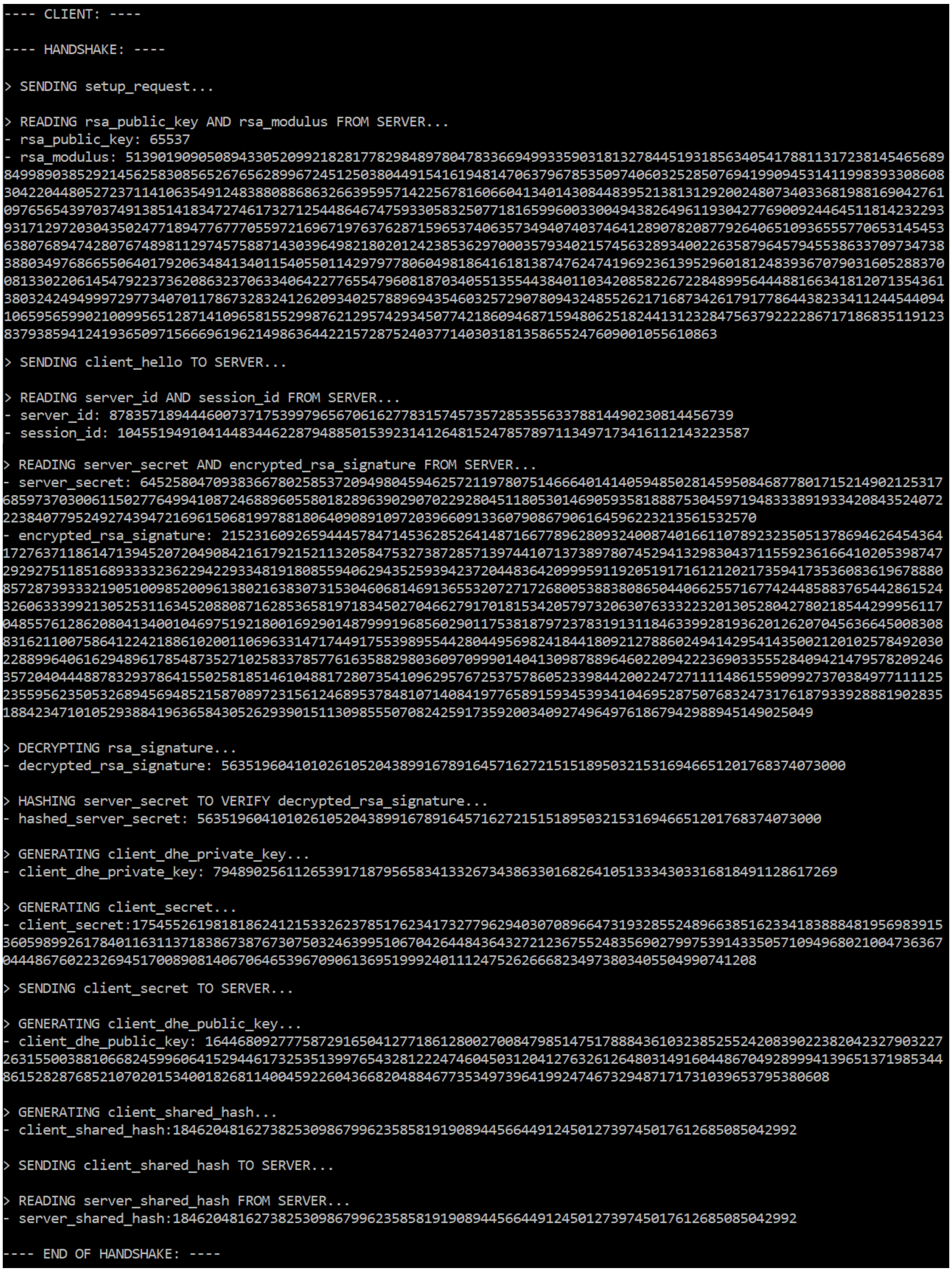
For ease of reading, I have split the output into 4 parts; Server Handshake, Client Handshake, Server Data Exchange, Client Data Exchange.

SERVER HANDSHAKE:



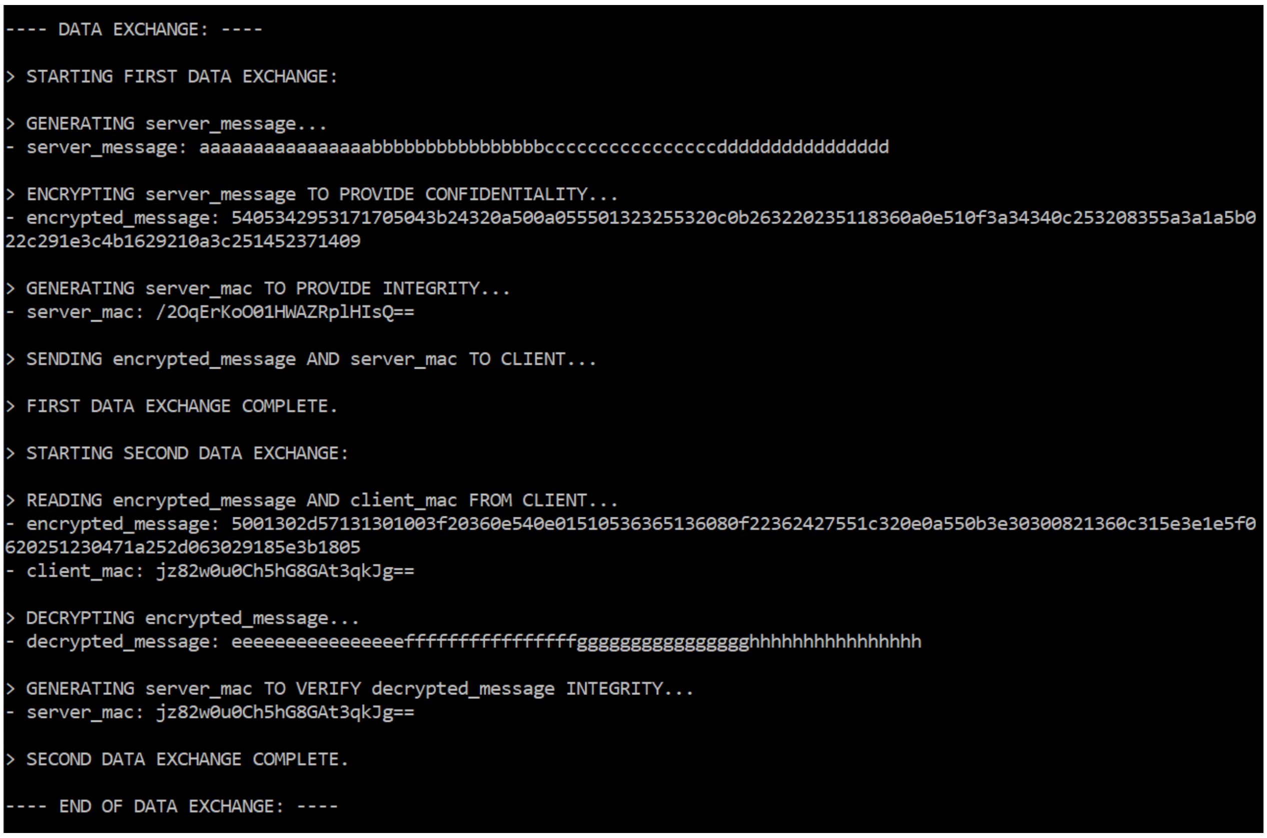
```
---- SERVER: ----

---- HANDSHAKE: ----

> READING setup_request FROM CLIENT...
- setup_request: hello

> SENDING rsa_public_key AND rsa_modulus TO CLIENT...

> READING client_id FROM CLIENT:
- client_id: 110555435816319192164450421024089438512359976611477507858717179179994092563059

> SENDING server_id AND session_id TO CLIENT...

> GENERATING server_dhe_private_key...
- server_dhe_private_key: 91406206557032585449683000254469586547126300533867895997325781581667738727919

> GENERATING server_secret...
- server_secret: 17824535836438743389156419269918355486744337587926093494857754963269687478416370100225850662604464O
513743072088282772119952723534132756827672308921222053176131955295189578908980904928686154968383318933372054162393352
559890633853959944315655685138215023617824457462764030893417467277353740774724194564032957020

> HASHING server_secret...
- hashed_server_secret: 95855181941899195876646657981377889928991936485942566761513053376273654810264

> GENERATING rsa_signature...
- rsa_signature: 344410017923462386207851422563155336797622784440186139712777323939762583367192622761365332770156987
653041132761871725764891926594741504061709468010845403980379556753020707241632422454537599607979903407747517646796 87
861722761115704558949569593059110522961269193412889132461106277158547723610786716681493122596724183203946792175186 17
243507669700914471615458329505704904546175771194350721438070057592449778928168045784784013945795737710880109522135 83
830534295762542862782257279144080643178291325452575089228312590712384393383835415611646188822906817789844277587625 48
608594306963329949772577712866727411215484999307537171843084013924594299950118227640597391958082036880321724858298 82
732337833705151251950288267786799395796708015628309910211058635458973661147498676344351764245498287691594798932869 39
829693980295077977948804985103666340137160682466966068122701227437364373130989718259689059525170882165949047265806 34
386673428285290112980247089048317287471160351854622195693896264368753075788563589801458885493213533227665067470009 83
665114249716092486010121974727770997878017916536522721824532400613052697022277637274732475874209594705151964216663 48
44751011800624558574817281457777068942021843180012099772770422696264481636389907111125 7984

> SENDING dhe_base, dhe_modulus, server_secret, rsa_signature TO CLIENT...

> READING client_secret FROM CLIENT...
- client_secret: 493691601288679064910387901834394417363974761316240281005053917913867814247627287576125754461727992
807809116580525157585280509858936547306987194087333808769057604488715460537471625444455208016329371336826540369834 81
071132608395487149918678568771498366227324863900016691328160980753089113273995366477754963213

> GENERATING server_dhe_public_key...
- server_dhe_public_key: 151489601049162142695294836574911462560240078357368772719886818731363162533431840821196758 9
528918266766179451749501592643964382905156142425346379586524754382592959985842660796604644163152797406143826695698 19
976582965592686053949228296256515579043266904122726667011761634608847355449932882180901901403520689928

> GENERATING server_shared_hash...
- server_shared_hash:18914930014297064994277287538828144233397981342248637425508322306690431997341

> SENDING server_shared_hash TO CLIENT...

> READING client_shared_hash FROM CLIENT...
- client_shared_hash:18914930014297064994277287538828144233397981342248637425508322306690431997341

---- END OF HANDSHAKE: ----
```

CLIENT HANDSHAKE:



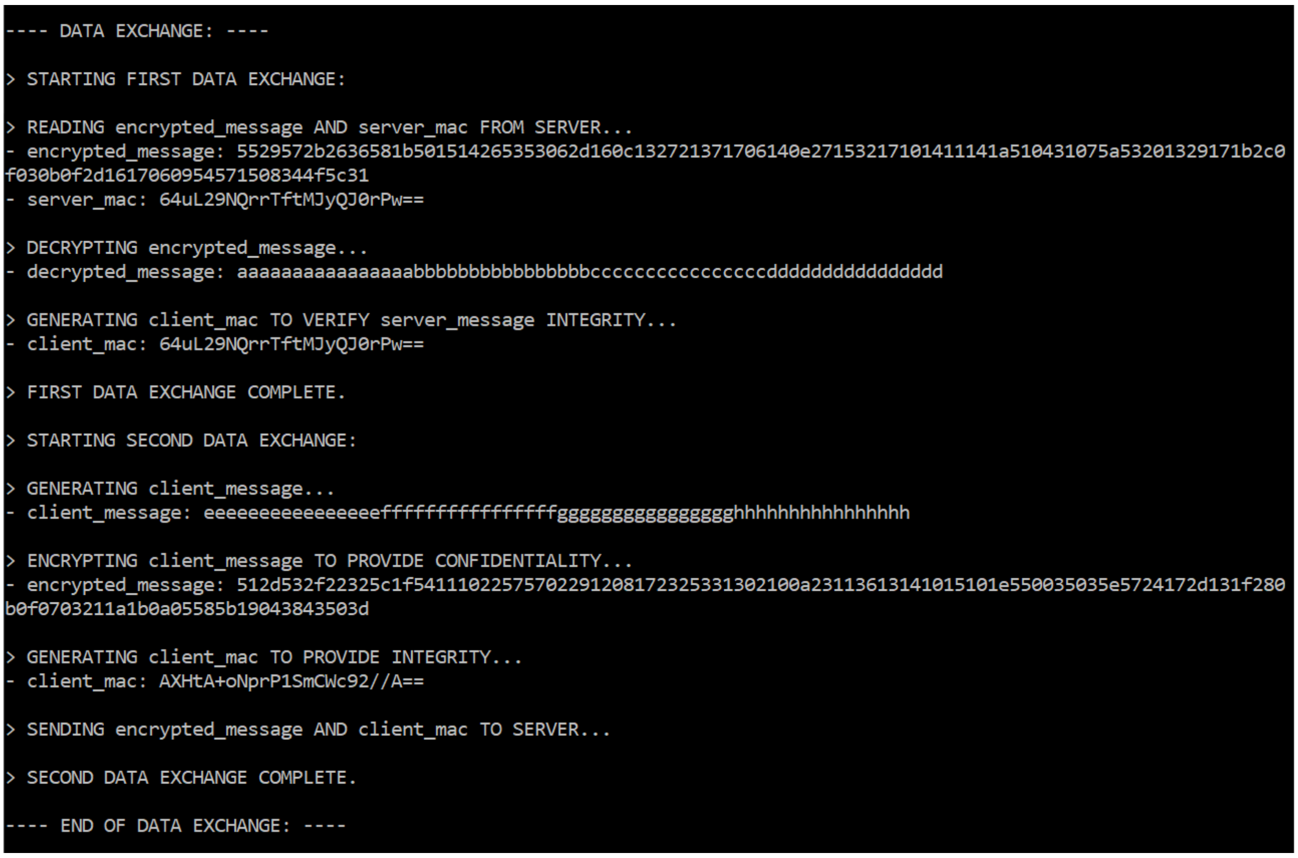
```
---- CLIENT: ----

---- HANDSHAKE: ----

> SENDING setup_request...

> READING rsa_public_key AND rsa_modulus FROM SERVER...
- rsa_public_key: 65537
- rsa_modulus: 51390190905089433052099218281778298489780478336694993359031813278445193185634054178811317238145465689
8499890385292145625830856526765628996724512503804491541619481470637967853509740603252850769419909453141199839330860 8
304220448052723711410635491248388088686326639595714225678160660413401430844839521381312920024807340336819881690427 61
097656543970374913851418347274617327125448646747593305832507718165996003300494382649611930427769009244645118142322 93
931712972030435024771894776777055972169671976376287159653740635734940740374641289078208779264065109365557706531454 53
638076894742807674898112974575887143039649821802012423853629700035793402157456328934002263587964579455386337097347 38
388034976866550640179206348413401154055011429797780604981864161813874762474196923613952960181248393670790316052883 70
081330220614547922373620863237063340642277655479608187034055135544384011034208582267228489956444881663418120713543 61
380324249499972977340701178673283241262093402578896943546032572907809432485526217168734261791778644382334112445440 94
106595659902100995651287141096581552998762129574293450774218609468715948062518244131232847563792222867171868351191 23
83793859412419365097156669619621498636442215728752403771403031813586552476090010556108 63
> SENDING client_hello TO SERVER...

> READING server_id AND session_id FROM SERVER...
- server_id: 878357189444600737175399796567061627783157457357285355633788144902308144567 39
- session_id: 104551949104144834462287948850153923141264815247857897113497173416112143223587

> READING server_secret AND encrypted_rsa_signature FROM SERVER...
- server_secret: 645258047093836678025853720949804594625721197807514666401414059485028145950846877801715214902125317
685973703006115027764994108724688960558018289639029070229280451180530146905935818887530459719483338919334208435240 72
223840779524927439472169615068199788180640908910972039660913360790867906164596223213561532570
- encrypted_rsa_signature: 215231609265944457847145362852641487166778962809324008740166110789232350513786946264543 64
172763711861471394520720490842161792152113205847532738728571397441071373897807452941329830437115592361664102053987 47
292927511851689333236229422933481918085594062943525939423720448364209995911920519171612120217359417353608361967888 0
857287393332190510098520096138021638307315304606814691365532072717268005388380865044066255716774244858837654428615 24
326063339921305253116345208808716285365819718345027046627917018153420579732063076333223201305280427802185442999561 17
048557612862080413400104697519218001692901487999196856029011753818797237831913118463399281936201262070456366450083 08
831621100758641224218861020011069633147174491755398955442804495698241844180921278860249414295414350021201025784920 30
228899640616294896178548735271025833785776163588298036097099901404130987889646022094222369033555284094214795782092 46
357204044488783293786415502581851461048817280735410962957672537578605233984420022472711114861559099273703849771111 25
235595623505326894569485215870897231561246895378481071408419776589159345393410469528750768324731761879339288819028 35
188423471010529388419636584305262939015113098555070824259173592003409274964976186794298894514902504 9

> DECRYPTING rsa_signature...
- decrypted_rsa_signature: 56351960410102610520438991678916457162721515189503215316946651201768374073000

> HASHING server_secret TO VERIFY decrypted_rsa_signature...
- hashed_server_secret: 56351960410102610520438991678916457162721515189503215316946651201768374073000

> GENERATING client_dhe_private_key...
- client_dhe_private_key: 79489025611265391718795658341332673438633016826410513334303316818491128617269

> GENERATING client_secret...
- client_secret:175455261981818624121533262378517623417327796294030708966473193285524896638516233418388848195698391 5
360598992617840116311371838673876730750324639951067042644843643272123675524835690279975391433505710949680210047363 67
0444867602232694517008908140670646539670906136951999240111247526266682349738034055049907412 08
> SENDING client_secret TO SERVER...

> GENERATING client_dhe_public_key...
- client_dhe_public_key: 164468092777587291650412771861280027008479851475178884361032385255242083902238204232790322 7
263155003881066824599606415294461732535139976543281222474604503120412763261264803149160448670492899941396513719853 44
861528287685210702015340018268114004592260436682048846773534973964199247467329487171731039653795380608

> GENERATING client_shared_hash...
- client_shared_hash:18462048162738253098679962358581919089445664491245012739745017612685085042992

> SENDING client_shared_hash TO SERVER...

> READING server_shared_hash FROM SERVER...
- server_shared_hash:18462048162738253098679962358581919089445664491245012739745017612685085042992

---- END OF HANDSHAKE: ----
```

SERVER DATA EXCHANGE:



```
---- DATA EXCHANGE: ----

> STARTING FIRST DATA EXCHANGE:

> GENERATING server_message...
- server_message: aaaaaaaaaaaaaaaabbbbbbbbbbbbbbbbccccccccccccccccdddddddddddddddd

> ENCRYPTING server_message TO PROVIDE CONFIDENTIALITY...
- encrypted_message: 5405342953171705043b24320a500a055501323255320c0b263220235118360a0e510f3a34340c253208355a3a1a5b0
22c291e3c4b1629210a3c251452371409

> GENERATING server_mac TO PROVIDE INTEGRITY...
- server_mac: /2OqErKoO01HWAZRplHIsQ==

> SENDING encrypted_message AND server_mac TO CLIENT...

> FIRST DATA EXCHANGE COMPLETE.

> STARTING SECOND DATA EXCHANGE:

> READING encrypted_message AND client_mac FROM CLIENT...
- encrypted_message: 5001302d57131301003f20360e540e01510536365136080f22362427551c320e0a550b3e30300821360c315e3e1e5f0
620251230471a252d063029185e3b1805
- client_mac: jz82w0u0Ch5hG8GAt3qkJg==

> DECRYPTING encrypted_message...
- decrypted_message: eeeeeeeeeeeeeeeeffffffffffffffffgggggggggggggggghhhhhhhhhhhhhhhh

> GENERATING server_mac TO VERIFY decrypted_message INTEGRITY...
- server_mac: jz82w0u0Ch5hG8GAt3qkJg==

> SECOND DATA EXCHANGE COMPLETE.

---- END OF DATA EXCHANGE: ----
```

CLIENT DATA EXCHANGE:



```
---- DATA EXCHANGE: ----

> STARTING FIRST DATA EXCHANGE:

> READING encrypted_message AND server_mac FROM SERVER...
- encrypted_message: 5529572b2636581b501514265353062d160c132721371706140e27153217101411141a510431075a53201329171b2c0
f030b0f2d16170609545715083444f5c31
- server_mac: 64uL29NQrrTftMJyQJ0rPw==

> DECRYPTING encrypted_message...
- decrypted_message: aaaaaaaaaaaaaaaabbbbbbbbbbbbbbbbccccccccccccccccdddddddddddddddd

> GENERATING client_mac TO VERIFY server_message INTEGRITY...
- client_mac: 64uL29NQrrTftMJyQJ0rPw==

> FIRST DATA EXCHANGE COMPLETE.

> STARTING SECOND DATA EXCHANGE:

> GENERATING client_message...
- client_message: eeeeeeeeeeeeeeeeffffffffffffffffgggggggggggggggghhhhhhhhhhhhhhhh

> ENCRYPTING client_message TO PROVIDE CONFIDENTIALITY...
- encrypted_message: 512d532f22325c1f5411102257570229120817232533130210 0a23113613141015101e550035035e5724172d131f280
b0f0703211a1b0a05585b19043843503d

> GENERATING client_mac TO PROVIDE INTEGRITY...
- client_mac: AXHtA+oNprP1SmCWc92//A==

> SENDING encrypted_message AND client_mac TO SERVER...

> SECOND DATA EXCHANGE COMPLETE.

---- END OF DATA EXCHANGE: ----
```