Получение ключей для шифрования:

Для начала нужно сгенерировать keyAES и keyMAC - это те ключи которыми мы будем шифровать данные при передаче:  
  
String keyAES = Security.getKey();  
String keyMAC = Security.getKey();  
  
 , затем эти ключи нужно зашифровать с помощю публичных ключей которые будут изначально вшиты в приложение (в скрипте это keyMod и keyExp):  
  
BigInteger keyMod = new BigInteger("4128129803496668451833756170118141868851614705806314994203614389510613404539543352564468242326067417323817062813109724033546102916918446969287233378526867");  
(изначально вшитый ключ в приложение)  
  
BigInteger keyExp = new BigInteger("3127925502487553883617082248057751761281245073620963837458066168905990700262438767821460201762431802142515638993587131725483191226686428040565351502696063");  
(изначально вшитый ключ в приложение)  
  
String encryptedKeyAES = Security.encryptRSA(keyAES, keyExp, keyMod);  
String encryptedKeyMAC = Security.encryptRSA(keyMAC, keyExp, keyMod);  
  
, так же генерируется параметр sequence(последовательность запросов, любое число которое затем при каждом запросе должно увеличиваться на один, это что бы в дальнейшем понимать что запрос пришол именно от приложения) и шифруем sequence сразу с помощью сгенерированым keyAES а затем полученный результат шифруем еще и с помощью сгенерированым keyMAC тоесть:  
  
String sequence = Integer.toString(Security.getSequence());  
String sequenceAES = Security.encrypt(sequence, keyAES);  
String sequenceMAC = Security.encrypt(sequenceAES, keyMAC);  
  
  
  
  
  
  
  
  
  
  
  
  
, затем отправляем следующий запрос на http://tapmoney.testmy.tk/rest/keyex - POST с action=setKeys

{  
 "sequence": sequenceAES,  
 "sequence\_MAC": sequenceMAC,  
 "action": "setKeys",  
 "encryptedKeyAES": encryptedKeyAES,  
 "encryptedKeyMAC": encryptedKeyMAC  
}

Пример реального запроса  
{

"sequence": "ue4LFmhuRK/BjH6ipO3PBQ==",

"sequence\_MAC": "WmdE+Fi32KuQM5w/GgCT+wMIvbzbA70I/KhMSPh+tBk=",

"action": "setKeys",

"encryptedKeyAES": "1548281295457413279429040521798673006578849460062704214555845517838594238320368536014207981800303392710450847788952163378498346253053265949175394148659954|1374533098252786818697288965866943387323686140324361533491598167966873076694591850923734576935766155182845976400920283703871479048145725720319455318680137|1464074923475818565407461418364410106779596452741889843278137590431402920357277492446860846162679717733780807007343729213652696639165659695940683325652336|1524450486697110329043010399978188664898889405235730158698166417573060515366544856444097381835695554912315049055037084962997124124483776806851270035070707|1171051314655048611835997142606646170165205550264734453581738993830554526716589105279246313731427365657787637008914330560368375162315806314216462150935102|2463371469547441169896091287895945220373283223053511430585349208431597347091109260266282437579622318790514482231839840668915126330709192446428210911509889|1286996349744524032080162586270560135050295407775810708365101364551921806135052840793866354677966619575311897274509717291312478853509158576737917490037963|3774837279661753914568091719958184547515559136594377127998238363916572446651723723012020913240687817503565579057070948988367852294873870271251075423458788|290905651403217254067672475802334145738779504046634623288790443606248636809143146931581017055946069642191469051828368099175270423196996616067544183361927|2986852577657821229977315309823886325946721415846819880351627002833486560363872782731503873157932546078875488027663986720329106858344648942358003712274082|1468509310790721241318893346156082811917314515571352676219897335523052075928884063142200266204544558770385640989582160673545623144853017754861181539615233|357582978299475556709203100541857009295557053272225771784333557869038776609377562061019397720262025754680793722256829840271386970697725003383468016498715|363212522962346823336508560249234612598550488553736985317878530177175324538808326708505539762096364114007953061265289467029811726722911791554589763637896|726516216580919147888863728216099834576327978517967399818985633120787823385516109424570554852186878087886565232759693774509779486497247022530497966313670|1916748449800289913099969265178155567431766181885123755969958190528052082216980753005819062396690450272632104916081999796614951423609131677245437290662766|1387208745721781789900182812335004346104160824019781730161394215522546103314575289495045148079522889502308366602398904378651560774040212945281432752270578",

"encryptedKeyMAC": "1524450486697110329043010399978188664898889405235730158698166417573060515366544856444097381835695554912315049055037084962997124124483776806851270035070707|3611034939733882653409435313222939203232727526433732770049139626144976256356413314201883779758893001070363224869633396320399014066284331905080917805162372|2713360716700088824780118051908184197882375075631664640132494481826863550654293536223133526349552198568761886083544192727702216027822310959701308687855873|3991655153482467050714749106489861832388344873436425122440392999364508840394385787567466630849918426170339247585573624775614672087972697674345224076457919|3428289207929714310351585701732316046835135554326518300962230080232368783102013866849817799065662997654702240546772879888750313741308114691946154189736144|3611034939733882653409435313222939203232727526433732770049139626144976256356413314201883779758893001070363224869633396320399014066284331905080917805162372|2728535310722027382062019280974074864004432616686688326934941485254661023988326872701343228006298104830604122282713500791026318358711021086711546168922877|3226949478450376286989943553106120937121932316256998234524415027664574941362173837862807294271653548263300931769104724807911937528775225048596215564903362|2463371469547441169896091287895945220373283223053511430585349208431597347091109260266282437579622318790514482231839840668915126330709192446428210911509889|1159638600864668756245018791698093881448714391332808284726734617892843669061974330602493241364490257540592681767142467256157305354565981516903188523406329|159240073422684986894401399009374454442652165962183305795352406411647200096615986069604491361046065466232897532223614145029558368146103458262280064967087|1916748449800289913099969265178155567431766181885123755969958190528052082216980753005819062396690450272632104916081999796614951423609131677245437290662766|1248292083231487273006627151855373920722966202035271696481128846923316392431763426455941790844284455175331332020742739649869536136862860057848534079885841|945055186004559802836896381458262709866437139083638937717760830208864952836269417393521846915638705355441031432616670343771477479037521360023942052532876|2517713024332620115679661645630141388194336911192976438276089172150987864609496833553317415993027754598867710524476514105035769595229836754996418612716464|4090949629777024546307209554894986028316444537104652883129458357269074069790667463410300407266727434565108934814788331823995740292538997276318282172612073"

}  
  
  
  
  
  
  
  
В ответ прийдет вот такие данные:  
{  
"string":"ZYPlvPHCUy4XhrnzR\_pFfPkBt-L7TY\_eXhwBJy6RRrSNpMP7rm",  
"string\_MAC":"J8GLaj2o7cukEiFQ8ofBL7JYsK3jZxfsKqaPcLPJuCwQHreoQBSYFTG99ne3MPwot68eFvLYa86JrVy37stK4g==",  
"idSession":134,  
"idSession\_MAC":"QlgIf38XuA8GvHFhsqi5iQ==",  
"KeyMAC":"I5PpL2LfOw93lhbeY9Pbo/WgyO3lZOh4rHB7j6X+5+Y=",  
"KeyMAC\_MAC":"fpcAwFT34qjIcfOrbNMpgFmoyorRuusM1V/TanOma+ylWMqfJqy62GwwkDnwgbLB"}  
  
string и string\_MAC это просто для дополнительной проверки как данные эти параметры не важны  
  
Далее нужно разшифровать string\_MAC и сверить полученный результат с string:  
  
String string\_from\_MAC = Security.decrypt(string\_MAC, keyMAC);  
keyMAC – был сгенерирован выше  
сравниваем теперь string\_from\_MAC с string если совпадают идем дальше  
  
  
idSession – это ID ключей keyAES и keyMAC на сервере, этот параметр нужно всегда передавать в запросе и вместе с ним обязательно idSession\_MAC(зашифрованный idSession с помощю keyMAC)  
Далее нужно разшифровать idSession\_MAC и сверить полученный результат с idSession:  
  
idSession\_from\_MAC = Security.decrypt(idSession\_MAC, keyMAC);  
сравниваем теперь idSession\_from\_MAC с idSession если совпадают идем дальше  
  
  
Далее нужно разшифровать KeyMAC \_MAC и сверить полученный результат с KeyMAC:  
  
KeyMAC\_from\_MAC = Security.decrypt(KeyMAC\_MAC, keyMAC);  
сравниваем теперь KeyMAC \_from\_MAC с KeyMAC если совпадают разшифровываем  
KeyMAC с помощью сгенерированого ранее ключа keyAES :

String KeyMAC\_real = Security.decrypt(KeyMAC, keyAES);

KeyMAC\_real – это будет новый KeyMAC, тоесть KeyMAC будет постоянно генерироваться новый а keyAES будет всегда один и тот же.  
  
  
  
  
  
  
  
Если все проверки выше прошли успешно генерируем данные и отправляем следующий запрос для подтверждения что эти ключи приняты:  
  
генерируем строку:  
  
String string = Security.getString();

Шифруем строку с помощью keyMAC  
  
String stringMAC = Security.encrypt(string, keyMAC);  
  
увеличиваем последовательность запроса на один и шифруем сразу с помощью keyAES, затем полученный результат шифруем с помощью keyMAC:  
  
sequence++  
String sequenceAES = Security.encrypt(sequence, keyAES);

String sequenceMAC = Security.encrypt(sequenceAES, keyMAC);

Берем idSession с ответа – 134 , и шифруем его с помощью ключа keyMAC:  
  
idSession = 134  
String idSessionMAC = Security.encrypt(idSession, keyMAC);  
  
{  
"string\_MAC": stringMAC  
,"sequence": sequenceAES,  
"idSession\_MAC": idSessionMAC,  
"string": string,  
"keysConfirm":1,  
"sequence\_MAC": sequenceMAC,  
"action":"confirm",  
"idSession": idSession  
}  
  
  
  
если ключи успешно подтвердились возвращается ответ в виде:  
  
{  
"keysConfirm":1,  
"KeyMAC":"rRWP5xAN/fKrUHYlBLy8H9Wzpi5NdPS0Lg7Equ0M8zo=",  
"KeyMAC\_MAC":"mODkiKJOp8i7FJiz+41H0mQizJ2wuGvVhFc0DFqk6DOFLvqh/OEP8UrqUA7OijRW"  
}