OSS Signature

1 Definition

Let p and q be large primes, and $n = pq$ be an integer. The Ong-Schnorr-Shamir (OSS) signature scheme picks the public key as $pk = (n, k)$, where $k \in \mathbb{Z}_n^*$.

For a message $m \in \mathbb{Z}_n$, the OSS signature on m is defined as a pair (s_1, s_2), such that:

$$m = s_1^2 + ks_2^2 \pmod{n}\,. \tag{1}$$

Notice that this signature scheme is randomized, and as a result, there are many valid signatures per message.

2 The Challenge

2.1 Public Key

Consider an OSS scheme with public key defined as below:

$n =$ 2500274667302321444325561141500416362281316785205092385845552903020388697784043599163302407984573633515046878453012330196146411149280295126398484303916405643083212516312338380571370725051525407316942470700935934175980600339259413829445816790283048415267269627431354100237607618387226623028533881755311042227789544502242340725234158378271966455460048583165349676235272729414041086200783924103482624640993740831758601610032011833930449356830887537971732449772775019089885401420289579878112986724053038732356771155724435920171857416377914076962270289293799763739963281375904672591324215387939414520243914582434260953073 3.

$k =$ 2336233940242237981777232731506150276159349436384664018037299234755236443232922075603023111295777861881007866676297457724722570908933442259178288474958438563294188505531828849508165141304162518369355323325283724376209882806408939697679678482951269169045612152838677815144527548924147182432558423831193984554191240398529121342514545372949097551884354228278567426169882675738157503841583622094450689491350012893308429179005946544584044089533943130399909686111345275200036473836443710543908066577077043624148656313271740759377649851220681388590008903680642931352487614677914819589706304021769517023773347344641866877991 6.

2.2 Known Message-Signature Pairs

For the given public key, assume we have two messages $m = 53$ and $m' = 97$, as well as valid OSS signatures $\sigma = (s_1, s_2)$ and $\sigma' = (s_1', s_2')$ on them:

$s_1 =$ 4996965626677818470899497036192572928953383468049981492215052947486968576469160069531045847406602867300998349717580001316746485225762698993261684213353602996748343182092335509307573088247624968954427896794412611830427704686799860673116498727830792282242951843221232059541413610254765610041160440513577441311377402794483664718403091720812660740195547201741143700725549004 4

9652735056977331343710554733124151571652560430149226865775558619093858626594736752292181369860
3847151625881131098032431190975920711377012137155839366204869205993663573232084527465728306732
531071378794321276304228712618173043121942149396615492478.

s_2 = 9339264669983291900327820828270098016379861097267077744514708712834641173057650202534710250765922313741944462789942785478170212231431727070626211785081581216011869498431598197306917945956602966039228580868647572114697339953682091890878192717124657001319847969571764996539075871196135114129827099959218903704028739118783540539731484487013859564619154565781303034838767726419327010072827695144834838584650689215227715966068273968350448794571951839296494120274908108873597642627135960002648616376004036368059928510282972523844522428329769439534207771320850592847756802766651250916352633401119949553102145237888877557923.

s'_1 = 1175310634525428873736158851314723717775040092854996360770705416694272018453287016466331325452687686428711409140325577346012670344636409621516735011752251223036346079688292673964033478609454101792743606188403296247585587072876334286528231228257044624899061913318090722801397648963960930687927282321711032264600151060684222917055844683612220607835138296825397086414359711046196517585083481766587553803379995751918462862144530248422810026343142703208941166513773384841689779724344806887348570177578834182034693594611742417484784962639166222305695072504372190763599000981437545373635947182501581706733483980623543554544 8.

s'_2 = 12316166200662617325765815544099575807764413310634795505343707038403232764360600618225429373635295166934848622912606605137676618701432674218527915857291619046154607273795107010842999973856769548732268370712874750839053527808464550788859449105335277743795696144339058053122452964971592298330255373330218368022944268493068838443426306443778736080987978564424372156971785476907882272963659885645066539763165385373011125956153655244095555806255476415567481015699996383149028800951684269545606335042830832959051917038675106732802995597508462654646750537646918088323007809730266045627946103737352610491304987610016877095 8.

2.3 The Task

Two get the flag, enter any valid signature $\sigma'' = (s''_1, s''_2)$ on the message $m'' = m \times m' = 5141$, such that (1) holds for the given public key.

For your convenience, the signature verification algorithm is implemented as `Verify.py`. Insaide this file, you can also find the public key mentioned above, as well as the message-signature pairs.

The same verification algorithm is implemented on our servers. Enter any valid signature on m'', and get the flag!