Gossip-6 LAYER4 Secure Communication Protocol

1. Need secure communication:
   - Establish shared AES256 key: Diffie Hellman key exchange,
   - Need to trust the source (signature),
   - Since there will be no CA, we need Proof of Work for the identity,
   - We are already given an out-of-band public key sharing mechanism.

2. For Alice (client):
   - Let handshake message $\boldsymbol{m = DHE_{pub}^{Alice} \mid RSA_{pub}^{Alice} \mid nonce}$ such that $\boldsymbol{scrypt_C(m) < k}$ for some predetermined $\boldsymbol{k} \in \mathbb{Z}^+$ and Scrypt hash function with configuration $\boldsymbol{C}$,
   - Sign the digest $\mathbf{SHA3\_256}(\boldsymbol{m})$ with $\boldsymbol{RSA_{priv}^{Alice}}$ as $\boldsymbol{s} = \boldsymbol{Sign}_{\boldsymbol{RSA_{priv}^{Alice}}}(\mathbf{SHA3\_256}(\boldsymbol{m}))$,
   - Add the signature to the message as $\boldsymbol{m^{Alice} = m \mid s}$ and send $\boldsymbol{m^{Alice}}$ to Bob (server).
3. For Bob (server):
   - Upon receiving $\boldsymbol{m^{Alice}}$, get the fields of the message as $\boldsymbol{m \mid s}$,
   - Check validity first as $\boldsymbol{scrypt_C(m) < k}$, if not valid then discard connection,
   - From $\boldsymbol{m}$, get the fields $\boldsymbol{DHE_{pub}^{Alice} \mid RSA_{pub}^{Alice} \mid nonce}$,
   - Make sure $\boldsymbol{RSA_{pub}^{Alice}}$ is a known and trusted public key*****,
   - Verify signature as $\boldsymbol{Verify}_{\boldsymbol{RSA_{pub}^{Alice}}}(\boldsymbol{m, s})$, if not valid then discard connection,
   - Let handshake message $\boldsymbol{m' = DHE_{pub}^{Bob} \mid RSA_{pub}^{Bob}}$,
   - Sign the digest $\mathbf{SHA3\_256}(\boldsymbol{m'})$ with $\boldsymbol{RSA_{priv}^{Bob}}$ as $\boldsymbol{s'} = \boldsymbol{Sign}_{\boldsymbol{RSA_{priv}^{Bob}}}(\mathbf{SHA3\_256}(\boldsymbol{m'}))$,
   - Add the signature to the message as $\boldsymbol{m^{Bob} = m' \mid s'}$ and send $\boldsymbol{m^{Bob}}$ to Alice (client).
4. For Alice again (client):
   - Upon receiving $\boldsymbol{m^{Bob}}$, get the fields of the message as $\boldsymbol{m' \mid s'}$,
   - From $\boldsymbol{m'}$, get the fields $\boldsymbol{DHE_{pub}^{Bob} \mid RSA_{pub}^{Bob}}$,
   - Make sure $\boldsymbol{RSA_{pub}^{Bob}}$ is a known and trusted public key*****,
   - Verify signature as $\boldsymbol{Verify}_{\boldsymbol{RSA_{pub}^{Bob}}}(\boldsymbol{m', s'})$, if not valid then discard connection.

5. Now that both Alice and Bob have $\boldsymbol{DHE_{pub}^{Alice}}$ and $\boldsymbol{DHE_{pub}^{Bob}}$, they can both calculate the shared secret as $\boldsymbol{AES256_{key} = DHE\left(DHE_{priv}^{Alice}, DHE_{pub}^{Bob}\right) = DHE\left(DHE_{pub}^{Alice}, DHE_{priv}^{Bob}\right)}$. After 1 round trip, the secure communication has been established. Each message following the handshakes will be encrypted with the $\boldsymbol{AES256_{key}}$.
6. Note that only the client side who is initiating the secure communication has to pay for the proof of work effort. Hence for malicious nodes who try to abnormally increase their out-degree, the price to pay increases linearly.

*****: We can check if a public key is known due to the out-of-band hostkey sharing mechanism.