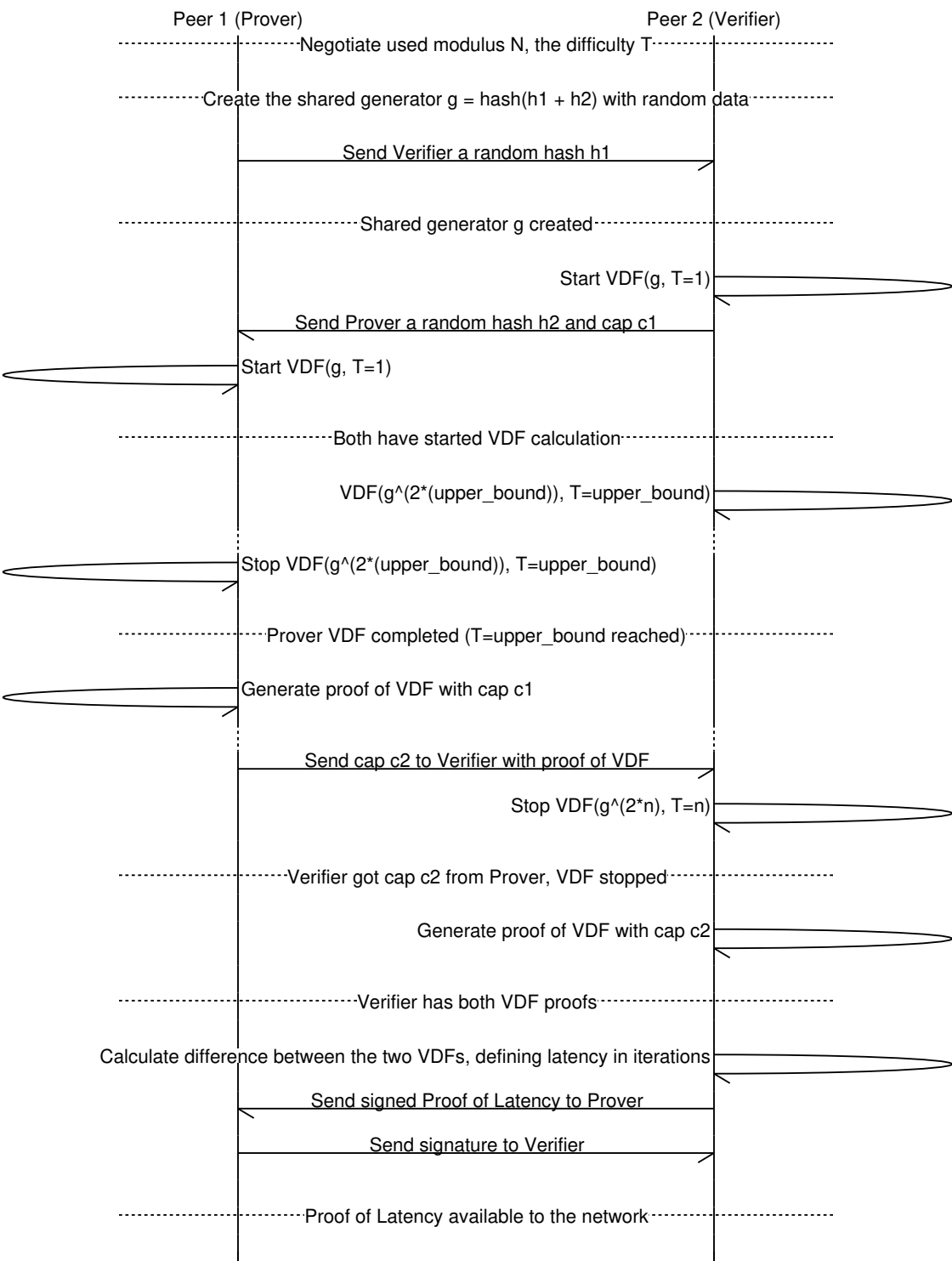
Peer 1 (Prover)
Peer 2 (Verifier)
Negotiate used modulus N, the difficulty T
Create the shared generator g = hash(h1 + h2) with random data
Send Verifier a random hash h1
Shared generator g created
Start VDF(g, T=1)
Send Prover a random hash h2 and cap c1
Start VDF(g, T=1)
Both have started VDF calculation
VDF(g^(2*(upper_bound)), T=upper_bound)
Stop VDF(g^(2*(upper_bound)), T=upper_bound)
Prover VDF completed (T=upper_bound reached)
Generate proof of VDF with cap c1
Send cap c2 to Verifier with proof of VDF
Stop VDF(g^(2*n), T=n)
Verifier got cap c2 from Prover, VDF stopped
Generate proof of VDF with cap c2
Verifier has both VDF proofs
Calculate difference between the two VDFs, defining latency in iterations
Send signed Proof of Latency to Prover
Send signature to Verifier
Proof of Latency available to the network