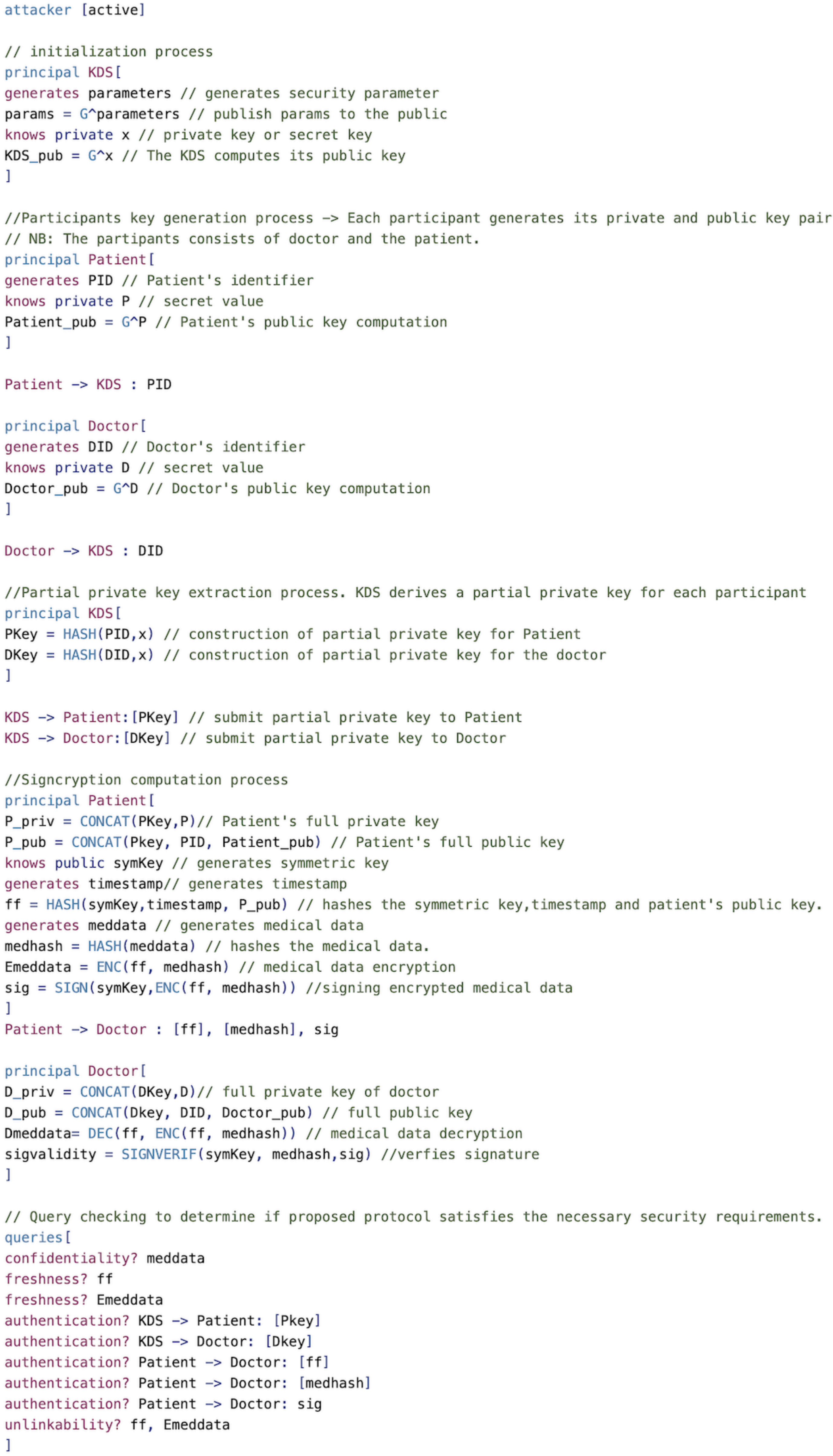

```
attacker [active]

// initialization process
principal KDS[
generates parameters // generates security parameter
params = G^parameters // publish params to the public
knows private x // private key or secret key
KDS_pub = G^x // The KDS computes its public key
]

//Participants key generation process -> Each participant generates its private and public key pair
// NB: The partipants consists of doctor and the patient.
principal Patient[
generates PID // Patient's identifier
knows private P // secret value
Patient_pub = G^P // Patient's public key computation
]

Patient -> KDS : PID

principal Doctor[
generates DID // Doctor's identifier
knows private D // secret value
Doctor_pub = G^D // Doctor's public key computation
]

Doctor -> KDS : DID

//Partial private key extraction process. KDS derives a partial private key for each participant
principal KDS[
PKey = HASH(PID,x) // construction of partial private key for Patient
DKey = HASH(DID,x) // construction of partial private key for the doctor
]

KDS -> Patient:[PKey] // submit partial private key to Patient
KDS -> Doctor:[DKey] // submit partial private key to Doctor

//Signcryption computation process
principal Patient[
P_priv = CONCAT(PKey,P)// Patient's full private key
P_pub = CONCAT(Pkey, PID, Patient_pub) // Patient's full public key
knows public symKey // generates symmetric key
generates timestamp// generates timestamp
ff = HASH(symKey,timestamp, P_pub) // hashes the symmetric key,timestamp and patient's public key.
generates meddata // generates medical data
medhash = HASH(meddata) // hashes the medical data.
Emeddata = ENC(ff, medhash) // medical data encryption
sig = SIGN(symKey,ENC(ff, medhash)) //signing encrypted medical data
]
Patient -> Doctor : [ff], [medhash], sig

principal Doctor[
D_priv = CONCAT(DKey,D)// full private key of doctor
D_pub = CONCAT(Dkey, DID, Doctor_pub) // full public key
Dmeddata= DEC(ff, ENC(ff, medhash)) // medical data decryption
sigvalidity = SIGNVERIF(symKey, medhash,sig) //verfies signature
]

// Query checking to determine if proposed protocol satisfies the necessary security requirements.
queries[
confidentiality? meddata
freshness? ff
freshness? Emeddata
authentication? KDS -> Patient: [Pkey]
authentication? KDS -> Doctor: [Dkey]
authentication? Patient -> Doctor: [ff]
authentication? Patient -> Doctor: [medhash]
authentication? Patient -> Doctor: sig
unlinkability? ff, Emeddata
]
```