Kerberos

Ticket Structure:
\[ E_{K(U)} \{ C, S, K_{C,S}, \text{timestamp, lifetime} \} \]

Phase 1
1. The user logs on to the client and the client asks for credentials for the user from Kerberos:
   \[ U \rightarrow C: \quad U (\text{user id}) \]
   \[ C \rightarrow K: \quad (U, tgs) \]

2. Kerberos constructs a ticket for U and tgs and a credential for the user and returns them to the client:
   \[ T_{\text{U,tgs}} = E_{K_{K(tgs)}} \{ U, tgs, K_{U,tgs}, \text{ts, lt} \} \]
   \[ K \rightarrow C: \quad E_{K(U)} \{ T_{U,tgs}, K_{U,tgs}, \text{ts, lt} \} \]
   The client obtains the user's password, P, and computes:
   \[ K'(U) = f(P) \]
   The user is authenticated to the client if and only if \( K'(U) \) decrypts the credential.

Phase 2
3. The client constructs an “ authenticator” for user U and requests from TGS a ticket for server, S:
   \[ A_U = E_{K(U,tgs)} \{ C, \text{ts} \} \]
   \[ C \rightarrow TGS: \quad (S, T_{U,tgs}, A_U) \]

4. The server authenticates the request as coming from C and constructs a ticket with which C may use S:
   \[ T_{C,S} = E_{K(S)} \{ C, S, K_{C,S}, \text{ts, lt} \} \]
   \[ TGS \rightarrow C: \quad E_{K(U,tgs)} \{ T_{C,S}, K_{C,S}, \text{ts, lt} \} \]

Phase 3
5. The client builds an authenticator and send it together with the ticket for the server to S:
   \[ A_C = E_{K(C,S)} \{ C, \text{ts} \} \]
   \[ C \rightarrow S: \quad (T_{C,S}, A_C) \]

6. The server (optionally) authenticates itself to the client by replying:
   \[ S \rightarrow C: \quad E_{K(C,S)} \{ \text{ts} + 1 \} \]