Algorithm 1 Sustainable Backdoor Attack based on Steganographic Algorithm

Input: original image P and the string S that needs to be encrypted into P,current epoch e, start epoch E_s, attack num E_a, end epoch E_e, client set C, selected client set C_n, adversary client set C_{adv}, global model G, local model θ, central server C_s, aggregate algrithm $PartFedAvg$, benign datasets D ,poisoned datasets D_p, benign learning rate η_b, poison learning rate η_p, PartFedAvg gradient removal scale $\mathcal{R}\%$

Output: a global model with high accuracy, stealth and robust backdoor and high accuracy in main-task

Define encoder U-Net and decoder STN
while train U-Net and STN **do**
  sample = Concatenate(Normalize(Totensor(P)),Totensor(S))
  P_{res} = U-Net(sample)
  $P_{trigger} = P + P_{res}$
  S_{decode} = STN($P_{trigger}$)
  $Loss = w_1 * P_{trigger} - P_{org} + w_2 * L_{LPIPS} + w_3 *$ CrossEntropyLoss(S, S_{decode}) $+ w_4 * D_{fake}$
  Update U-Net and STN by $Loss$
C_s select n clients by random into C_n , C_s build a global model G , C_s send G to each client in C_n
for $e < E_e$ and $e < E_s + E_a$ **do**
  for the k-th client C_e^k in C_n **do**
    if $C_e^k \in C_{adv}$ **then**
      poisoned dataset D_p = U-Net(D) + D
      Download G as local model L and train by D_p,
      Compute gradient by D_p on batch B_i of size ℓ
      $g_{e+1}^p = \frac{1}{\ell}\sum_{i=1}^{\ell} \nabla_\theta \mathcal{L}(\theta_{C_e^k}, D_p)$
      for $Value(g_{e+1}^p[x,y])$ in g_{e+1}^p **do**
        if $Value(g_{e+1}^p[x,y]) \subseteq top_{5\%}(Value(g_{e+1}^p[x,y]))$ **then**
          Set $g_{e+1}^p[x,y] = 0$
      Update $\theta_{C_{e+1}^k} = \theta_{C_e^k}^k - \eta_p g_{e+1}^p$
      Upload $\theta_{C_{e+1}^k}$ to C_s
    else if client $C_e^k \notin C_{adv}$ **then**
      Download G as local model L and train by D,
      Compute gradient by D_b on batch B_i of size ℓ
      $g_{e+1}^b = \frac{1}{\ell}\sum_{i=1}^{\ell} \nabla_\theta \mathcal{L}(\theta_{C_e^k}, D)$
      Update $\theta_{C_{e+1}^k} = \theta_{C_e^k}^k - \eta_p g_{e+1}^b$
      Upload $\theta_{C_{e+1}^k}$ to C_s
  C_s recieve $\sum_1^k \theta_{C_{e+1}^k}$ and randomly set $\mathcal{R}\%$ of update $\sum_1^k \theta_{C_{e+1}^k}$ to zero
  Generate update U_{e+1} for G_{e+1}
  $G_{e+1} = G_e - U_{e+1}$
for epoch $<$ E_e and epoch $>$ $E_s + E_a$ **do**
  Conduct normal federated learning client training, upload model updates, and update the global model.
return Final global model G with backdoor