A. Algorithm outline

Algorithm 1 Pseudocode for ESTI

Input: A clean model f_C and a poisoned model f_P with randomly initialized parameters θ. Dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$ with C classes.The clean data pool $\mathcal{D}_C$ and the poisoned data pool $\mathcal{D}_P$. $\mathcal{L}$ is SCE loss. **Output:** Deployment model f_T

Initialize clean model parameters θ_C randomly;
$\mathcal{D}_C \leftarrow$ clean seed samples;
for $I = 1$ to 2 **do**
  train f_C on D_C
  Calculate $\mathcal{L}(f_C)$ for the entire $\mathcal{D}$;
  Calculate the split threshold τ_{KDE};
  Update D_C with samples from D where $L(f_C)$ losses is lower than τ_{KDE}.
  Update D_P with samples from D where $L(f_C)$ losses is greater than τ_{KDE}.
  Initialize poisoned model parameters θ randomly;
  train f_P on D_P ;
  Calculate $\mathcal{L}(f_P)$ for the entire $\mathcal{D}$;
  Calculate the split threshold τ_{KDE};
  $\mathcal{D}_C \leftarrow$ samples from $\mathcal{D}$ where $\mathcal{L}(f_P)$ losses is greater than τ_{KDE};
  $\mathcal{D}_P \leftarrow$ samples from $\mathcal{D}$ where $\mathcal{L}(f_P)$ losses is lower than τ_{KDE};
end for
Initialize poisoned model parameters θ_P randomly;
train f_P on D_P with high learning rate;
Calculate $\mathcal{L}(f_P)$ for the entire $\mathcal{D}$;
Calculate the split threshold τ_{KDE};
$\mathcal{D}_C \leftarrow$ samples from $\mathcal{D}$ where $\mathcal{L}(f_P)$ losses is greater than τ_{KDE};
Initialize poisoned model parameters θ randomly;
train f_P on D_P with low learning rate;
Calculate $\mathcal{L}(f_P)$ for the entire $\mathcal{D}$;
Calculate the split threshold τ_{KDE};
$\mathcal{D}_P \leftarrow$ samples from $\mathcal{D}$ where $\mathcal{L}(f_C)$ losses is lower than τ_{KDE};
initialize f_T parameters θ as $\theta_T \leftarrow \theta_C$
for $I = 1$ to 2 **do**
  $\mathcal{D}'_P \leftarrow$ samples in $\mathcal{D}$ label flipping to y_{trap};
  train f_T on $\mathcal{D}_C \cup \mathcal{D}'_P$
  Calculate $\mathcal{L}(f_P)$ for the entire $\mathcal{D}$;
  Calculate the split threshold τ_{KDE};
  $\mathcal{D}_C \leftarrow$ samples from $\mathcal{D}$ where $\mathcal{L}(f_T)$ losses is lower than τ_{KDE};
  $\mathcal{D}_P \leftarrow$ samples from $\mathcal{D}$ where $\mathcal{L}(f_T)$ losses is greater than τ_{KDE};
end for

B. Implementation details

In this study, we trained over 100 epochs on a unified dataset composed of CIFAR-10[18] and GTSRB[29] images, resized to 32x32 pixels. For Tiny-ImageNet[9] we trained the sample epochs and resize its images to 64*64 pixels. The model is optimized using Stochastic Gradient Descent(SGD) with a starting learning rate of 0.01, momentum of 0.9, and weight decay set at 5×10^{-4}. The learning rate follows a cosine annealing schedule with a period of 100 epochs, and the training utilizes a batch size of 128. For the loss functions, Symmetric Cross-Entropy(SCE) loss is used. The entire process is implemented in PyTorch and conducted on a system equipped with an RTX 4090 GPU, ensuring a consistent and reproducible environment.

C. Attack Settings

C.0.1. Settings for BadNets.

As suggested by Gu et al. [12], we implement a 3×3 square as the trigger pattern in the bottom right corner of the images in both CIFAR-10 and GTSRB datasets.

C.0.2. Settings for WaNet.

For CIFAR-10 and GTSRB, we set the noise rate $\rho_n = 0.2$, control grid size $k = 4$, and warping strength $k = 0.5$.

C.0.3. Settings for IAB.

We re-implemented the Input-Aware Backdoor (IAB) attack as detailed in [25]. This process involved developing a trigger generator based on the original study's descriptions. We then used this generator to create poisoned samples in advance, facilitating the execution of a poisoning-based backdoor attack.

C.0.4. Settings for LC.

For the LC[32] attack, we implemented a grid-based strategy by positioning triggers at all four corners of the input image, with a reduced amplitude setting of 1.

C.0.5. Settings for Blended.

For the Blended[8] attack, we employ two variants: Blended-H and Blended-R. For Blended-H, both the training and testing phases use an α of 0.2. For Blended-R, the α is 0.2 during training and 0.5 during testing.

C.0.6. Settings for SIG.

For the SIG[2] attack, specific parameters are configured to adjust the attack's intensity and targeting precision. We set $\Delta = 40$ and $f = 6$.

C.0.7. Settings for SSBA.

For the SSBA attack, we follow the implementation guidelines as recommended as described in the settings of [20].