

A Hard-Label Black-Box Evasion Attack against ML-based Malicious Traffic Detection Systems

Zixuan Liu, Yi Zhao, Zhuotao Liu, Qi Li, Chuanpu Fu, Guangmeng Zhou, Ke Xu

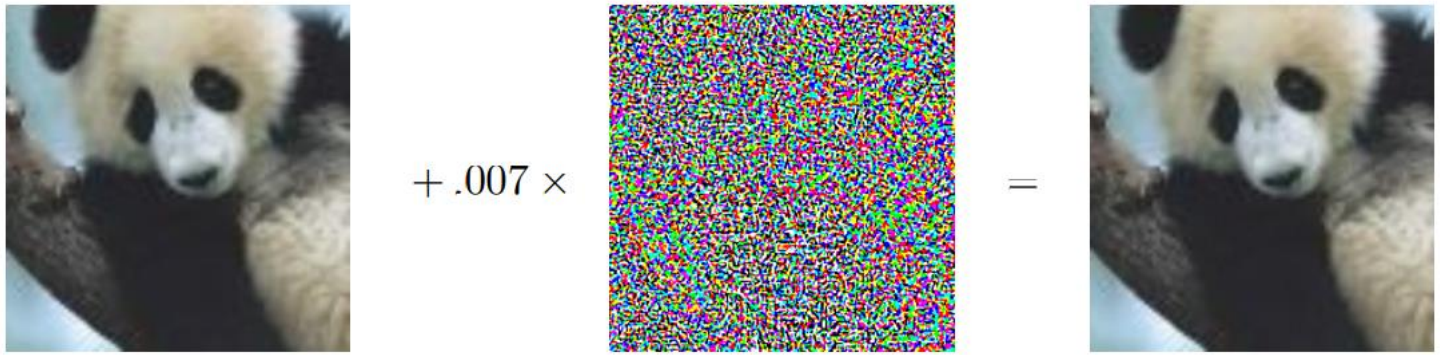
Tsinghua University
Beijing Institute of Technology
Zhongguancun Lab

1 Background: ML-based Traffic Analysis

- The use of ML techniques for traffic analysis has been rapidly increasing.
  - Traffic Detection
  - Traffic Classification
  - Flow Correlation
  - …
- Models are trained either on statistical traffic features or on packet contents, **and the former is more common**.
  - K-FP, USENIX Security 16
  - DeepLog, CCS 18
  - KitSune, CCS 18
  - DeepCor, CCS 18
  - Whisper, CCS 21
  - Flowlens, NDSS 21
  - HyperVision, NDSS 23
  - NetBeacon, USENIX Security 23
  - pVoxel, CCS 23
  - Exosphere, CCS 24
  - tFusion, CCS 25
  - …

1 Background: Adversarial (Evasion) Attack

- However, ML techniques have been shown to have vulnerabilities in security, which are susceptible to ***adversarial attacks***.

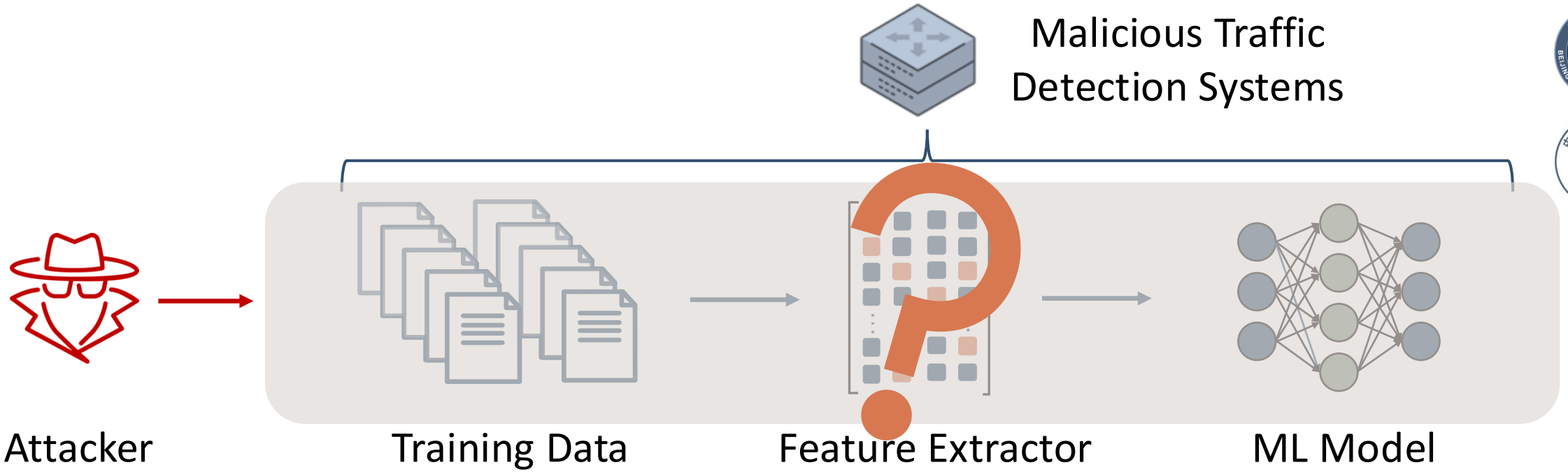


Panda Tiny Perturbations Predicted: Gibbon

An adversarial attack is the deliberate introduction of subtle perturbations into input data to deceive an ML model into making incorrect predictions.

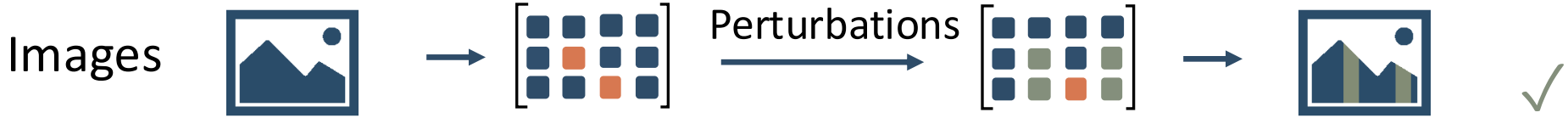
- Question: Do ML-based traffic analysis models easily fall victim to adversarial attacks?

2 Challenge: Black-Box

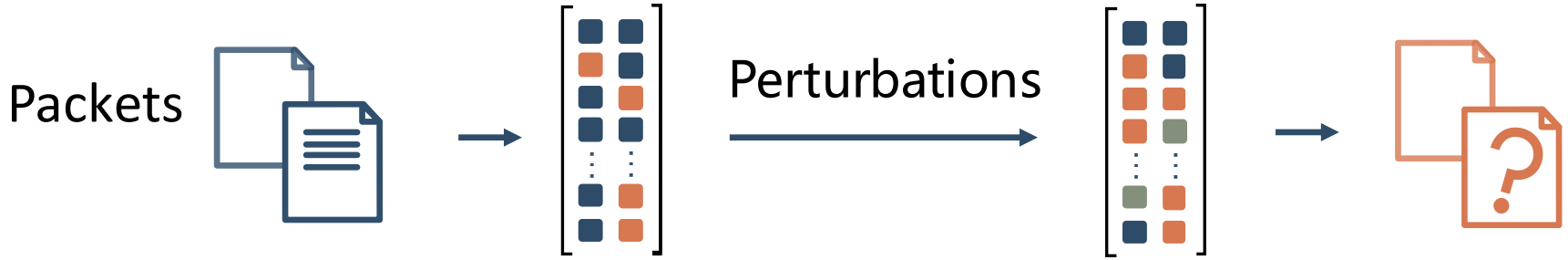


- Attackers cannot access any internal **implementation details** (i.e., data and model) of target detection systems in real-world scenarios.
- They may only observe binary **pass/fail feedback** on their traffic, formulating a strict hard-label setting.

2 Challenge: Domain Constraints



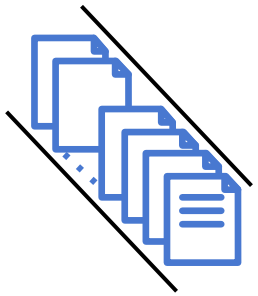
Images: Pixels can be arbitrarily perturbed in the feature space without destroying visual semantics



Traffic: Arbitrary modifications often break protocol constraints and invalidate the malicious payload.

- Furthermore, the attack must be protocol-agnostic and task-agnostic to seamlessly support diverse malicious flows.

NetMasquerade : A Hard-Label Black-Box Evasion Attack against ML-based Malicious Traffic Detection Systems



Enormous Public Benign Traffic

Benign Traffic Pattern Mimicking

- We establish a tailored pre-trained model called Traffic-BERT to capture diverse benign traffic patterns.
- Traffic-BERT learns to reconstruct realistic per-packet attributes from public datasets through a Mask-Fill task.

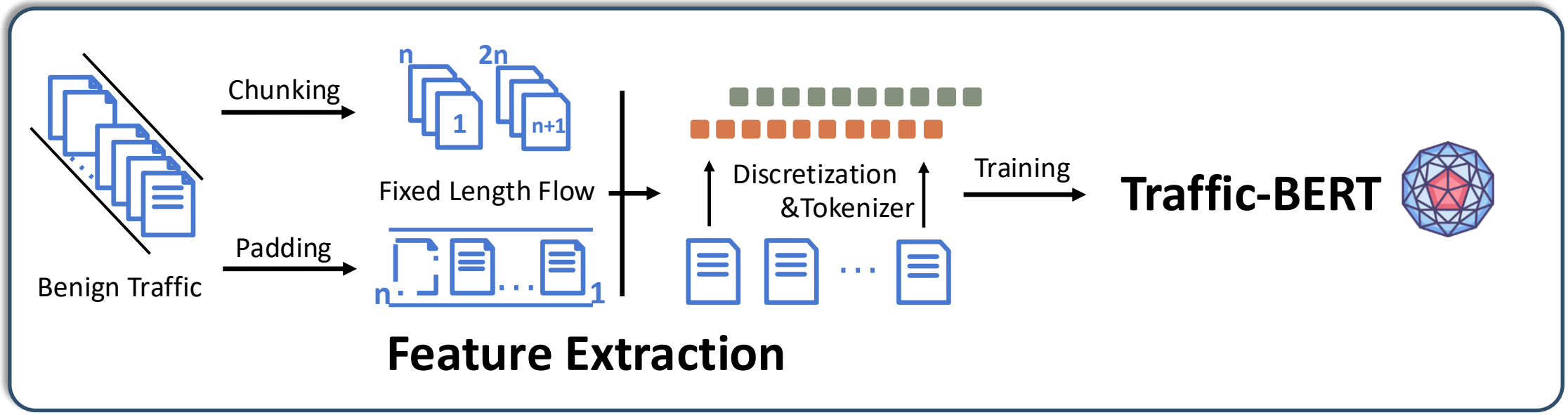
Malicious Traffic

Adversarial Traffic Generation

- A reinforcement learning agent selects the optimal packet modification positions to ensure minimal changes.
- Traffic-BERT fills these positions with benign features guided solely by the target system's pass/fail feedback.

Adversarial Malicious Traffic

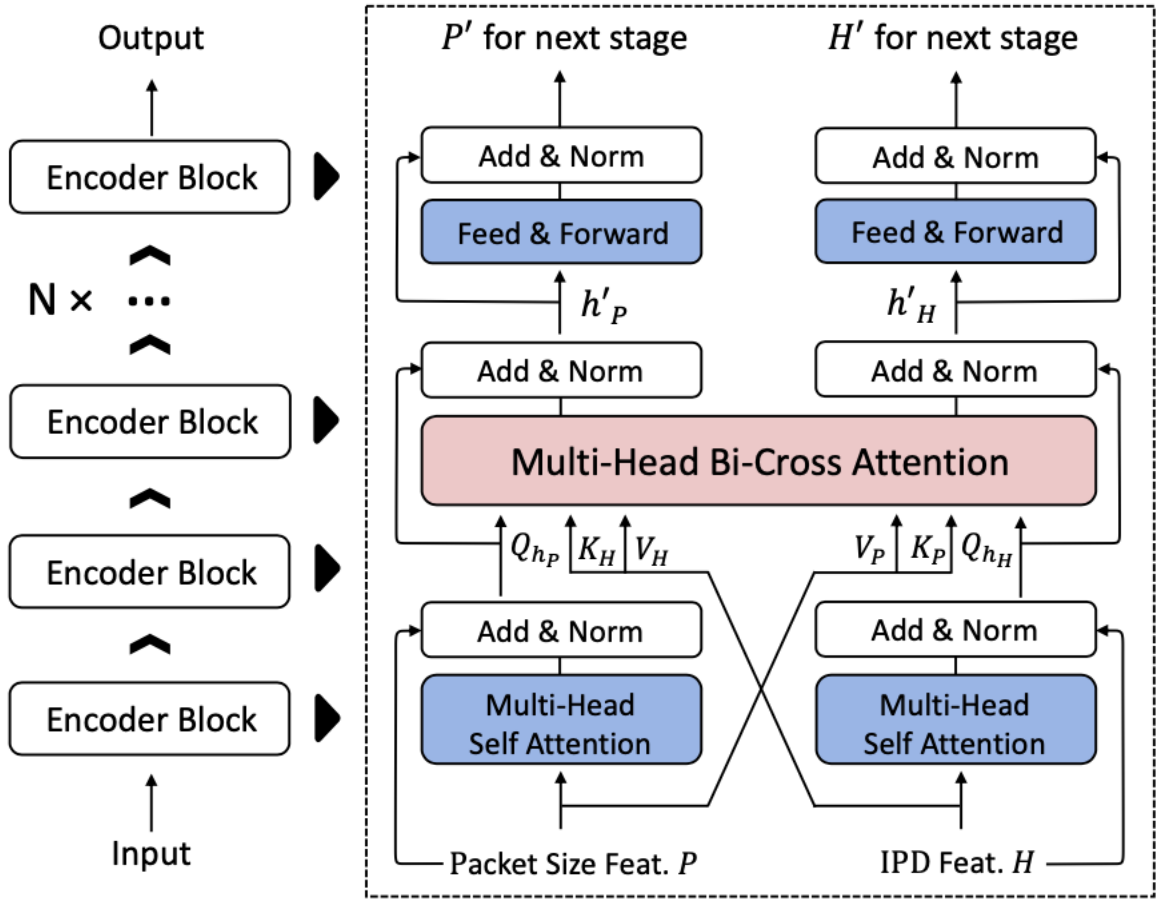
3 Design: Benign Traffic Pattern Mimicking



Flow Standardization: Real-world flow lengths show a long-tail distribution. Based on this observation, we apply padding and chunking to standardize sequence inputs.

Feature Tokenization: We hash continuous delays into logarithmic intervals and convert bimodal packet sizes directly into tokens.

3 Design: Traffic-BERT



Core Design of Traffic-BERT

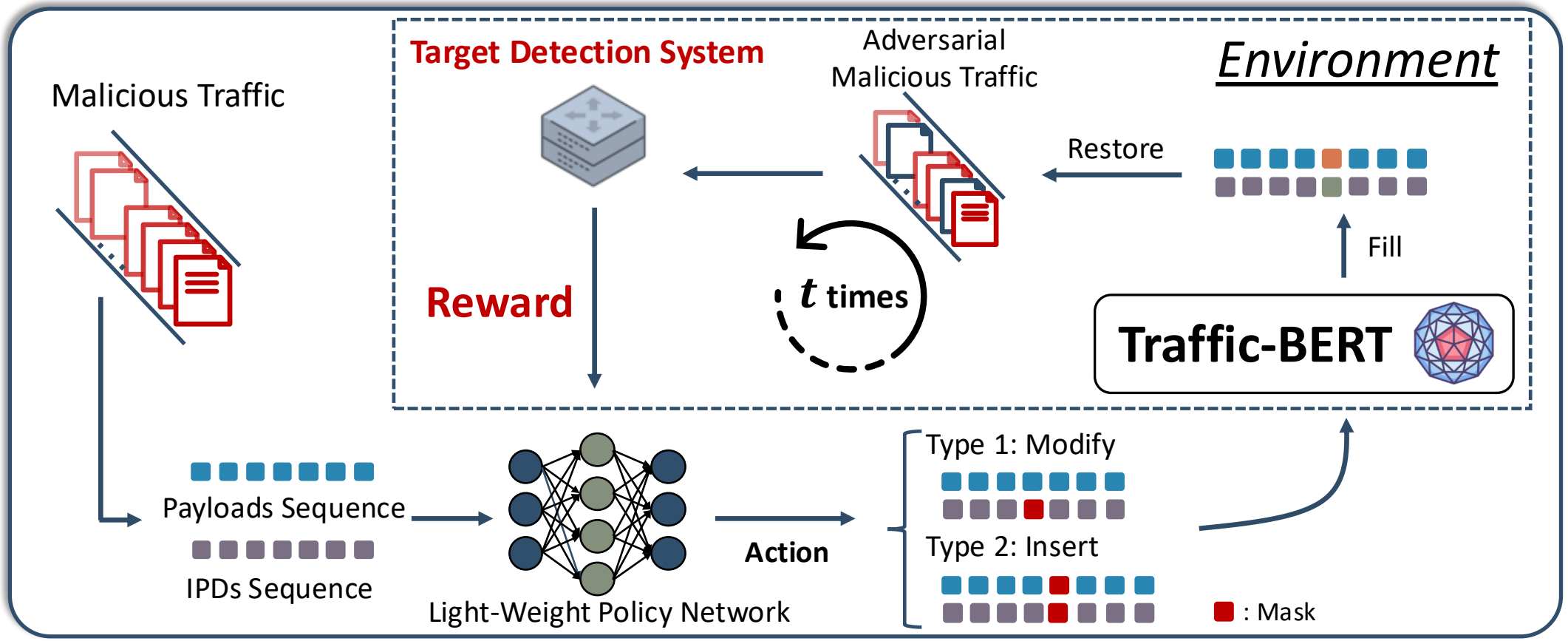
Parallel Processing: Traffic-BERT processes packet sizes and delays as parallel inputs. Self-attention layers first generate the independent hidden states h_P and h_H.

Bi-Cross Attention: The model explicitly fuses these two feature modalities. h_P acts as a query for the delay features. h_H acts as a query for the size features.

$$h'_P = h_P + \mathcal{A}ttn.(Q_{h_P}, K_H, V_H),$$
$$h'_H = h_H + \mathcal{A}ttn.(Q_{h_H}, K_P, V_P).$$

Mask-Fill Training: We train the model by masking tokens in both sequences at the same time.

3 Design: Adversarial Traffic Generation



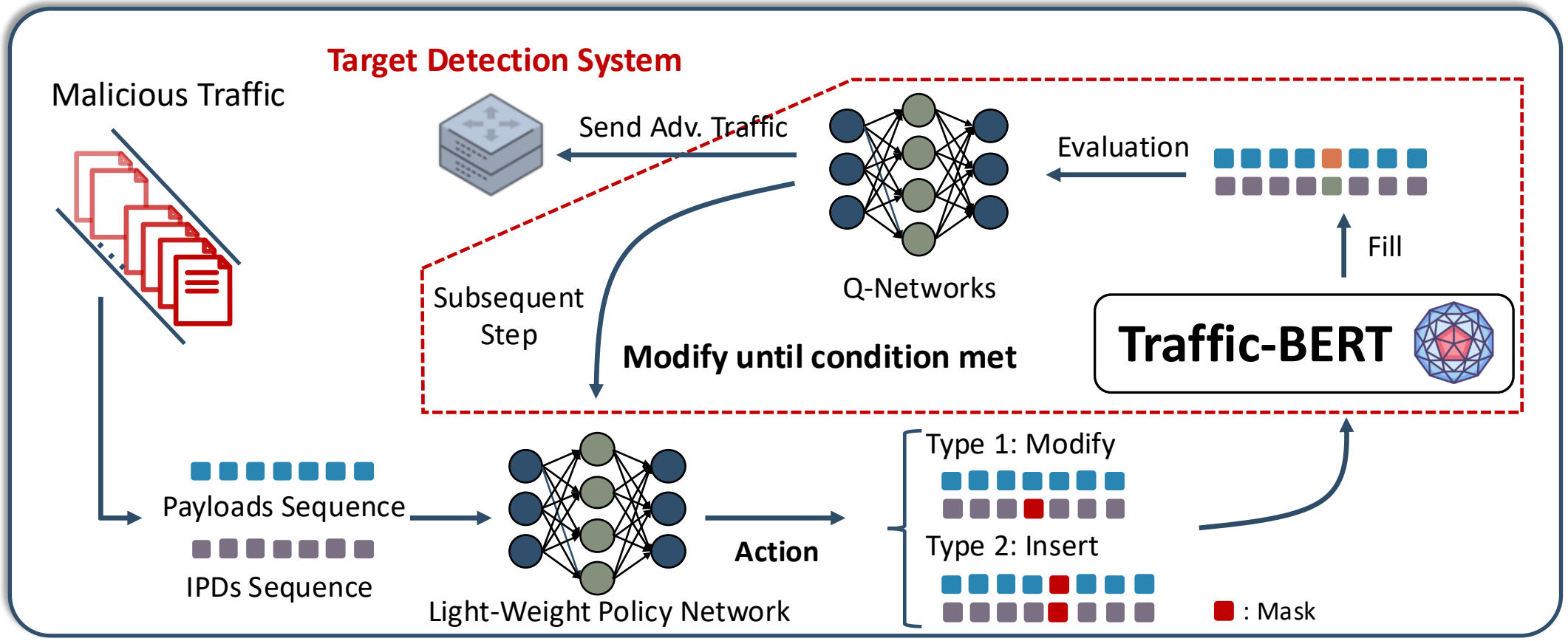
Action: The policy network selects a position to mask (either modify or insert).

Fill: Traffic-BERT fills the masked position with benign features.

Restore: The feature sequence is restored into adversarial traffic.

Update: The agent calculates the reward and updates the policy network.

3 Design: Inference

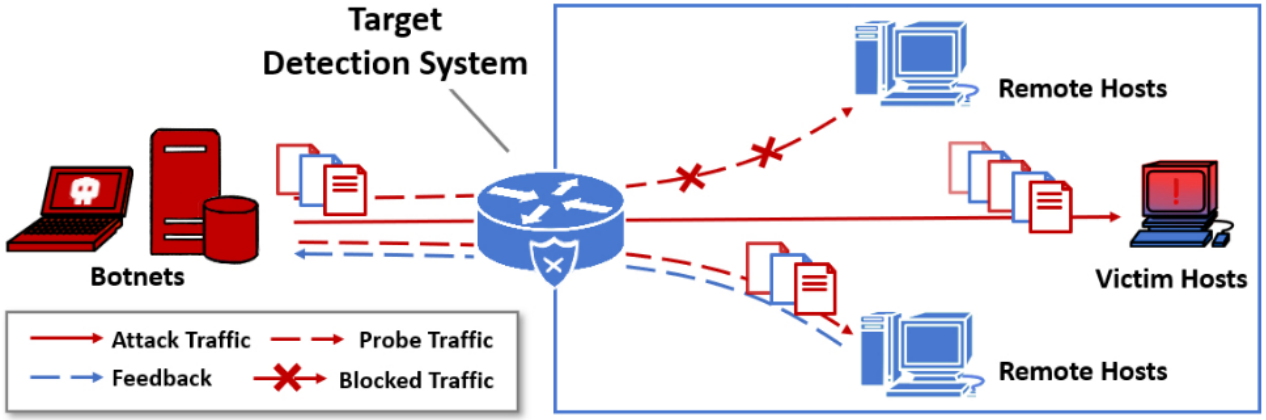


During inference, the agent relies on the trained Q-networks to evaluate actions offline. The modification process terminates once the estimated Q-value reaches a predefined threshold, without interacting with the target environment.

3 Design: Reward

$$r(s_t, a_t) = r_E(s_t, a_t) + \beta \cdot r_D(s_t, a_t) + \gamma \cdot r_M(s_t, a_t)$$

Evasion Reward r_E : The attacker sends probe traffic to remote hosts. The target system's pass/drop action provides strict binary feedback. We use this feedback to calculate the evasion success rate.

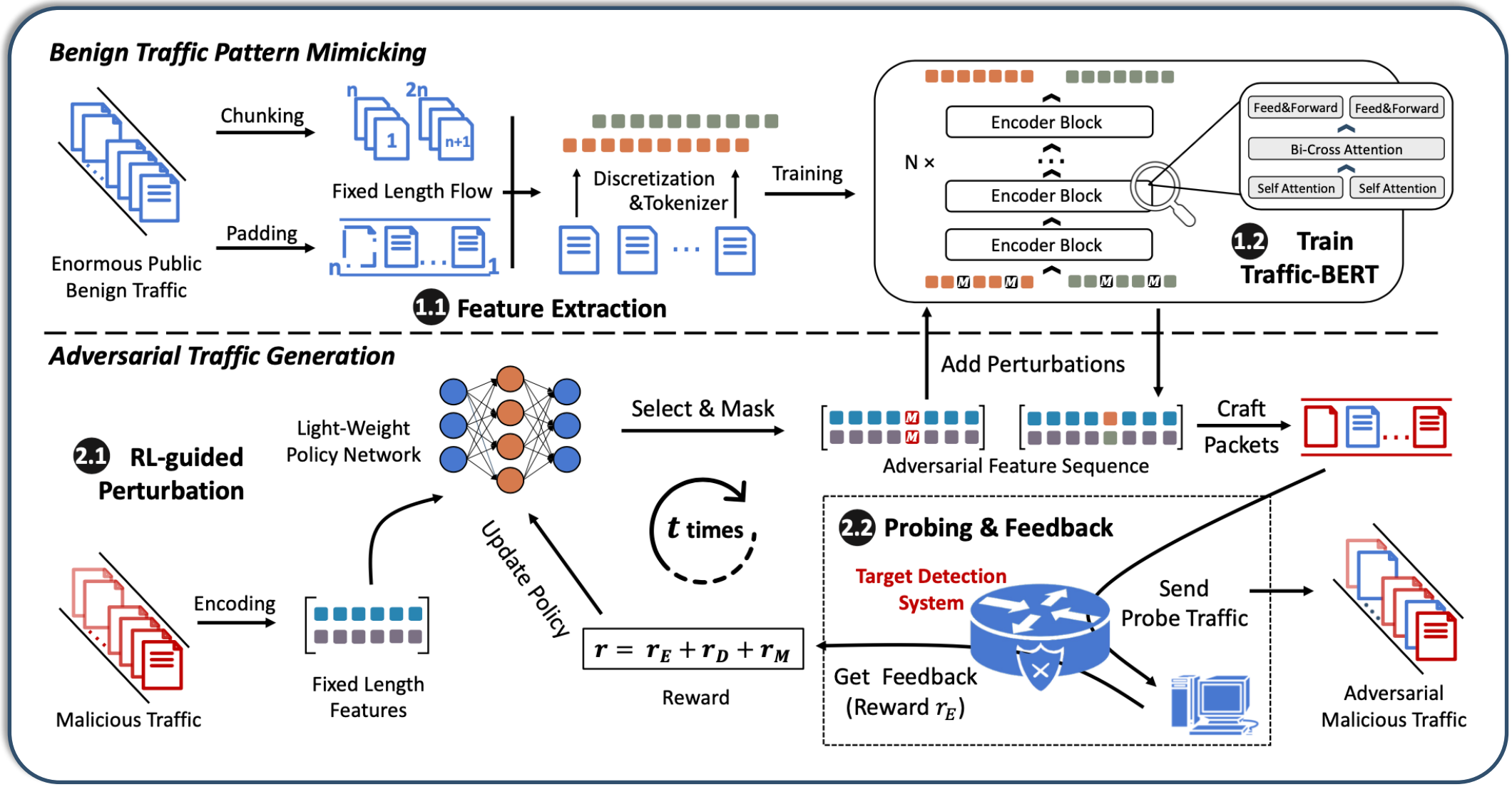


Dissimilarity Penalty r_D : It strictly penalizes each modification step. This enforces minimal perturbations and significantly accelerates the attack.

$$r_D(s_t, a_t) = -1.$$

Effectiveness Penalty r_M : It preserves the original attack semantics. For example, it explicitly maintains the high flow rate for DoS attacks.

3 Overall Design of NetMasquerade



4 Experiments: Setup

- **Malicious Traffic**: We replay **12** kinds of malicious traffic across **4** categories:
  - ✓ *Reconnaissance*: OS Scan, Fuzz Scan.
  - ✓ High-rate DoS: SSDP Flood, SYN DoS.
  - ✓ Botnet: Mirai, Zeus, Storm, Waledac.
  - ✓ Encrypted Web: Webshell, XSS, CSRF, Spam.

- **Target Systems**: We test the effectiveness on **6** top-performing detection systems:
  - ✓ Traditional ML-based: Whisper (CCS'21), FlowLens (NDSS'21), NetBeacon (USENIX Security'23)
  - ✓ DL-based: Vanilla + RNN (Generic Baseline), CICFlowMeter (ICISSP'16), Kitsune (NDSS'18)

4 Experiments: Attack Success Rate

Target System		Methods	Recon.&Scan.		DoS		Botnet				Encrypted Web Attacks				Overall
			Scan	Fuzz.	SSDP	SYN	Mirai	Zeus	Storm	Waledac	Webshell	XSS	CSRF	Spam	
Traditional ML-based systems	Whisper	R.M.	- [1]	0.0100	-	0.2552	0.2324	0.1011	0.2289	0.0585	0.0812	0.0721	0.1717	0.0927	0.1087
		M.I.	0.8907	0.0756	0.1132	0.3346	0.5521	0.5719	0.4590	0.4251	0.6802	0.7010	0.7259	0.7319	0.5218
		T.M.	0.9344	0.9270	0.7712	0.2790	0.6355	0.2551	0.1820	0.3664	0.5839	0.5527	0.6055	0.9072	0.5833
		Amoeba	0.9999	0.9934	0.9999	0.9998	0.9167	0.9254	0.9844	0.8970	0.9999	0.9999	0.9966	0.8381	0.9626
		NetM.	0.9999	0.9965	0.9999	0.9467	0.9988	0.9972	0.9999	0.9355	0.9999	0.9999	0.9999	0.9795	0.9878
	FlowLens	R.M.	-	-	0.1782	0.7660	0.6893	0.0760	0.3846	0.0434	0.0100	-	0.0150	-	0.1802
		M.I.	0.9800	0.1158	0.2375	0.5950	0.9370	0.4941	0.6510	0.3114	0.6391	0.5959	0.6633	0.1313	0.5293
		T.M.	0.0222	0.1525	0.9344	0.9125	0.8591	0.2670	0.8374	0.2899	0.0760	0.0736	0.0036	0.3913	0.4016
		Amoeba	0.9976	0.9442	0.9999	0.9990	0.8776	0.8665	0.9252	0.8000	0.9990	0.9999	0.9295	0.9700	0.9424
		NetM.	0.9999	0.9335	0.9999	0.9995	0.9537	0.9102	0.9990	0.9955	0.9795	0.9999	0.9428	0.9475	0.9717
	NetBeacon	R.M.	-	-	0.5291	0.1823	0.2864	0.0230	-	0.0790	0.6294	0.3916	0.1066	0.1030	0.1942
		M.I.	0.6511	-	0.2285	0.2841	0.5544	0.3455	0.3032	-	0.8781	0.7010	0.6446	0.1134	0.3920
		T.M.	0.6494	0.2435	0.8577	0.4393	0.3047	0.1992	0.4415	0.2180	0.4585	0.5645	0.5294	0.9091	0.4846
		Amoeba	0.9900	0.9999	0.9987	0.9999	0.9999	0.5905	0.6916	0.9727	0.9550	0.9999	0.9894	N/A [2]	0.8490
		NetM.	0.9999	0.9999	0.9999	0.9999	0.9899	0.9449	0.9965	0.9999	0.9999	0.9955	0.9999	0.8448	0.9809
DL-based systems	Vanilla	R.M.	-	0.3660	0.0455	0.5815	0.1163	-	-	0.3299	0.0118	-	0.0050	0.0515	0.1256
		M.I.	0.9510	-	-	0.3355	0.8769	-	0.5415	0.6711	0.6085	0.5353	0.6751	0.1958	0.4492
		T.M.	-	0.0375	0.8600	0.6550	0.0790	0.2232	0.2595	0.1617	0.0492	0.0278	0.0264	0.8636	0.2702
		Amoeba	0.9999	0.9999	0.9999	0.9999	0.8038	0.7156	0.6540	0.2682	0.9975	0.9999	0.9455	0.2538	0.8032
		NetM.	0.9999	0.9985	0.9825	0.9890	0.9817	0.9894	0.9805	0.9687	0.9999	0.9999	0.9999	0.8485	0.9782
	CIC.	R.M.	-	0.0422	0.1100	0.6398	0.5578	0.2467	0.2922	0.0301	0.0151	0.1855	0.4467	0.1031	0.2224
		M.I.	0.2300	0.1367	0.9711	0.5735	0.7111	0.3956	0.5396	0.2122	0.7011	0.6185	0.6598	0.2886	0.5032
		T.M.	0.1444	-	0.9822	0.6520	0.6656	0.1433	0.3026	0.1021	0.0311	0.0445	0.3381	0.6391	0.3371
		Amoeba	0.9999	N/A	0.9999	0.9112	0.9980	0.9999	0.8704	0.8182	0.9800	0.9865	0.9999	N/A	0.7970
		NetM.	0.9999	0.9744	0.9999	0.9959	0.9999	0.9867	0.8898	0.9810	0.9767	0.9999	0.9999	0.7475	0.9626
	Kitsune	R.M.	-	-	0.2379	0.3744	0.2949	0.0360	0.0990	0.2901	-	0.0277	0.0374	-	0.1165
		M.I.	0.3514	0.4484	0.0913	0.1815	0.8109	0.0801	0.4424	0.6334	0.6159	0.4498	0.3493	0.5359	0.4159
		T.M.	0.9760	0.9860	0.7848	0.5590	0.9049	0.4735	0.8318	0.7878	0.8884	0.8965	0.8406	0.6949	0.8020
		Amoeba	0.9339	N/A	0.8949	0.9292	0.9915	0.9449	0.7256	0.4595	0.4355	0.7814	0.7017	N/A	0.6498
		NetM.	0.9049	0.9850	0.8218	0.9333	0.9968	0.9359	0.9911	0.9291	0.9219	0.9231	0.9177	0.7522	0.9177

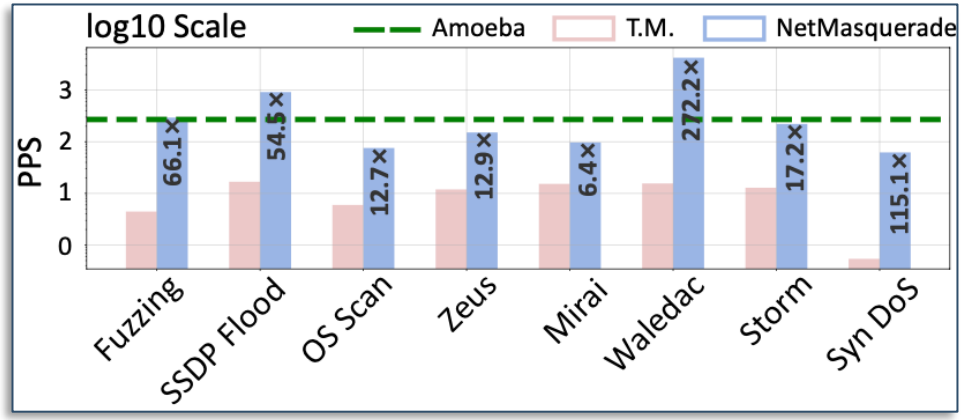
ASR Comparison

- ✓ Achieves an average ASR of **> 96.65%** against 6 advanced detection systems.
- ✓ Outperforms the best baselines by up to **21.88%**, securing the highest ASR in **56/72** evaluated scenarios.

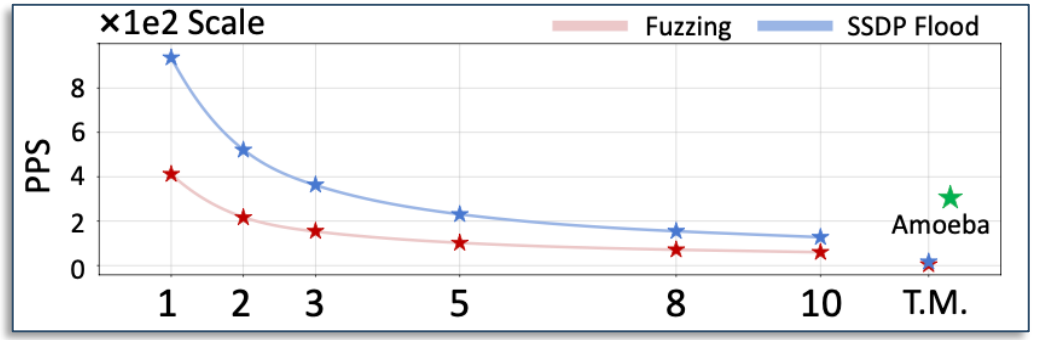
4 Experiments: Efficiency

- **Inference Speed**: NetMasquerade achieves high-throughput adversarial traffic generation, operating **69.6×** faster than baselines on average.

- **Training Overhead**: Stage 1 (Traffic-BERT) is pre-trained offline, allowing the Stage 2 RL policy to converge online within just 1 hour.



Throughput Comparison



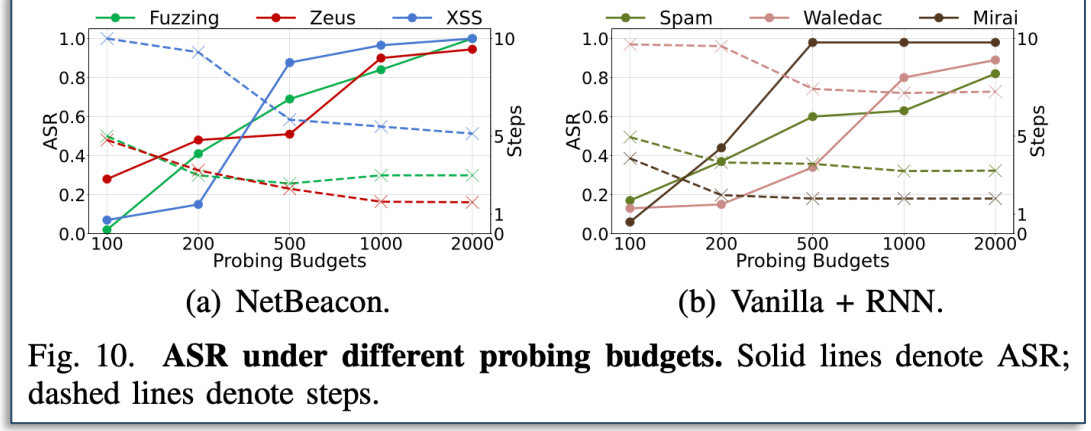
Throughput vs. steps under attack

4 Experiments: Deep Dive

- For scenarios where attackers cannot get enough feedback / control number of queries to avoid triggering security alarms (**limited probes**), NetMasquerade achieves high ASR within just 1,000 to 2,000 probes.

- For scenarios where target systems return unreliable or deceptive feedback to mislead attackers (**noisy feedback**), NetMasquerade remains highly effective at a 15% noise level.

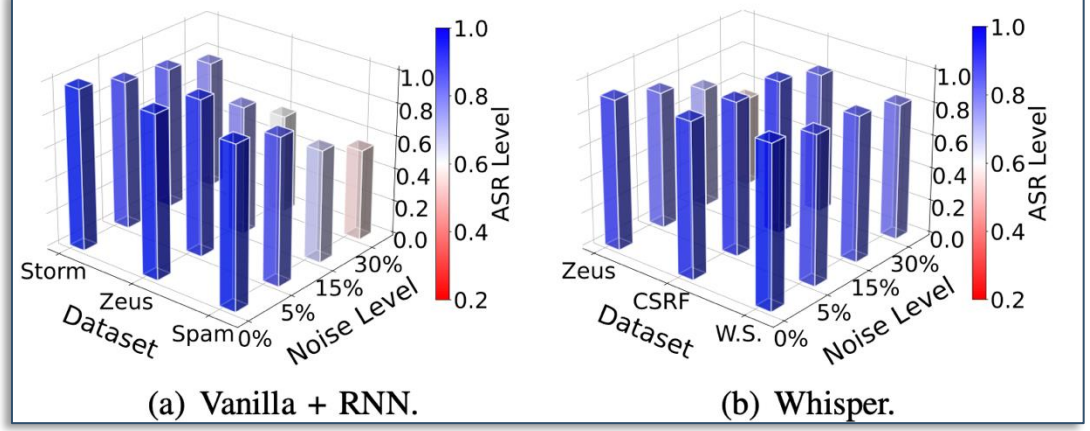
ASR under different probing budgets



(a) NetBeacon. (b) Vanilla + RNN.

Fig. 10. **ASR under different probing budgets.** Solid lines denote ASR; dashed lines denote steps.

ASR under different noise levels



(a) Vanilla + RNN. (b) Whisper.

4 Experiments: Deep Dive

Effect of Penalty Terms: The effectiveness penalty r_M maintains the original malicious intent. For DoS traffic, the bandwidth distribution remains nearly identical and the KL divergence is around 0.01.

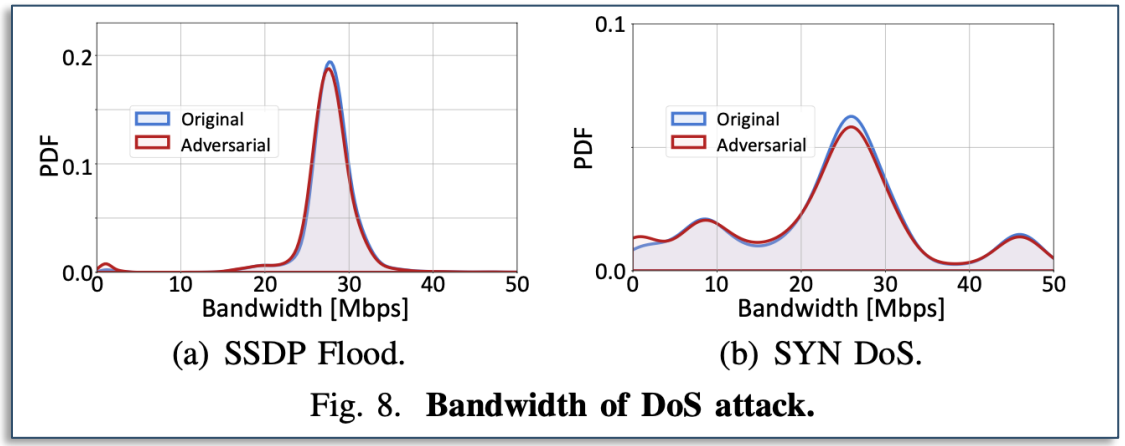


Fig. 8. **Bandwidth of DoS attack.**

Robustness Against Defenses: NetMasquerade maintains a high ASR against advanced feature-space defenses. Future defenses may consider traffic-space adversarial training or dynamic model randomization.

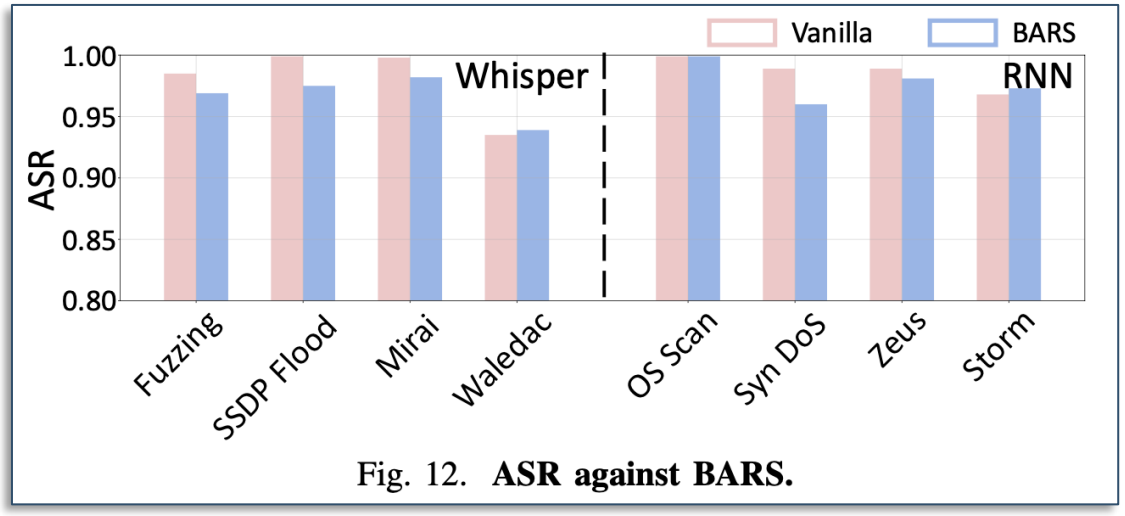


Fig. 12. **ASR against BARS.**

4 Conclusion

Method: We propose NetMasquerade, a practical hard-label black-box evasion attack against ML-based traffic detection systems. We design a two-stage architecture integrating a tailored Traffic-BERT for benign pattern mimicking and an RL agent for adversarial generation.

Performance: NetMasquerade achieves >96.65% average ASR against 6 top-performing detectors across 80 attack scenarios.

Future Defenses: Defenses may consider incorporating traffic-space adversarial training and dynamic model randomization to thwart unrestricted attacks.

Full Paper

Code

A Hard-Label Black-Box Evasion Attack against ML-based Malicious Traffic Detection Systems

liu-zx21@mails.tsinghua.edu.cn

Zixuan Liu, Yi Zhao, Zhuotao Liu, Qi Li, Chuanpu Fu, Guangmeng Zhou, Ke Xu

Tsinghua University, Beijing Institute of Technology, Zhongguancun Lab