HorusEye: A Realtime IoT Malicious Traffic Detection Framework using Programmable Switches

Yutao Dong[1,2)], Qing Li[2*)], **Kaidong Wu**[1,2)], Ruoyu Li[1,2)], Dan Zhao[2)], Gareth Tyson[3)], Junkun Peng[1,2)], Yong Jiang[1,2)], Shutao Xia[1,2)] and Mingwei Xu[4)]

1) *Tsinghua Shenzhen International Graduate School*
2) *Peng Cheng Laboratory*
3) *Hong Kong University of Science and Technology (GZ)*
4) *Tsinghua University*

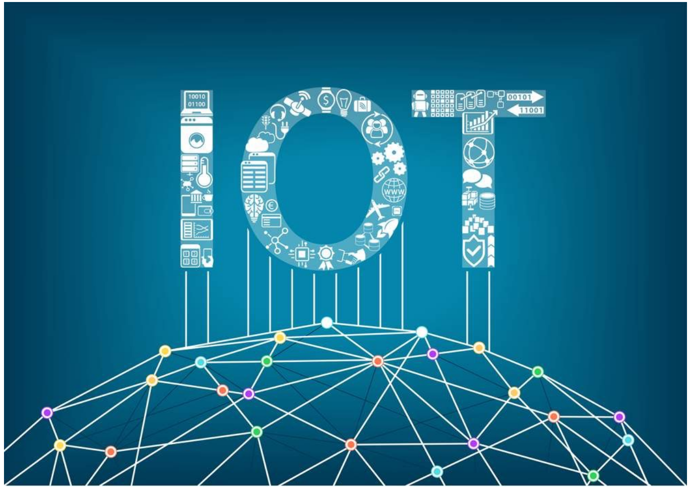
香港科技大學
THE HONG KONG UNIVERSITY OF SCIENCE AND TECHNOLOGY

Contens

Background

The number of Internet of Things (IoT) connections is increasing dramatically.

- It is expected to reach 8.01 billion in 2025 and video streaming is the dominant traffic in IoT traffic. [1]

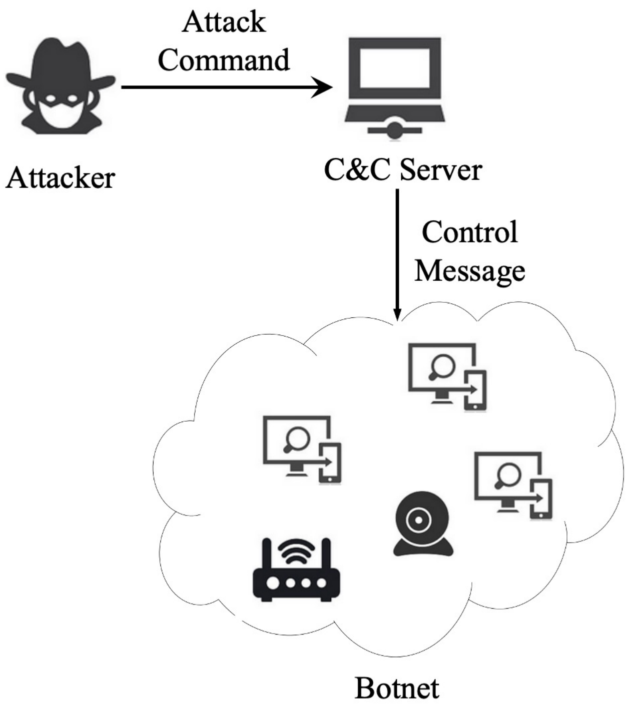


[1] Intelligence G. The mobile economy 2020[EB/OL]. 2020. https://www.gsmaintelligence.com

Background

IoT security issues remain severe.

- Exposed to the outdoors and vulnerable to attacks through physical connections.
- Unable to deploy complex security mechanisms due to their limited hardware.
- Attackers can get unauthorized access and transform IoT devices into part of a botnet.



Background

How to achieve low-cost and real-time anomaly detection?

Rule-based Detection Systems

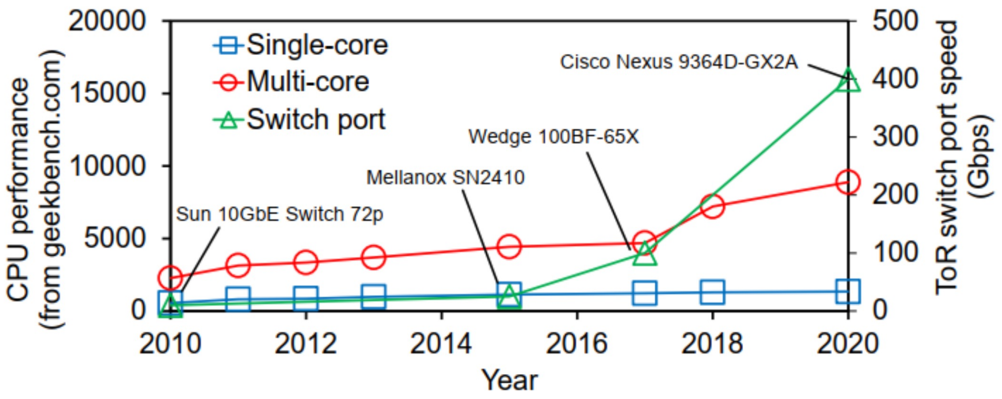
- ✅ Pros: available for scenarios requiring high throughput, packet-level schemes are low-cost.
- ❌ Cons: fail to detect unseen attacks and can be easily bypassed.

Unsupervised Learning-based Systems

- ✅ Pros: great generalizability, may discern unseen attacks.
- ❌ Cons: designed to deploy on the control plane without sufficient throughput, uploading traffic is high cost.

Background

New perspective on anomaly detection: Programmable Switches

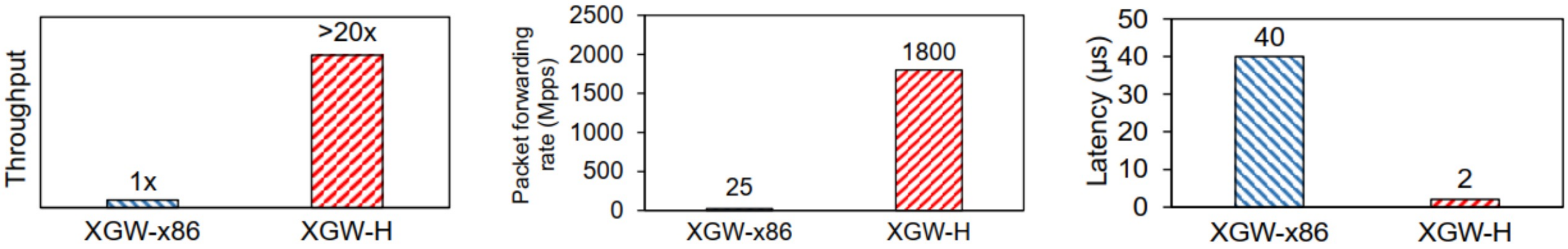


The performance of the CPU on the control plane for processing and forwarding packets can't keep up with the surge in traffic, but the switch can.

Figure: Sigcomm-2021 Sailfish: Accelerating Cloud-Scale Multi-Tenant Multi-Service Gateways with Programmable Switches

Background

New perspective on anomaly detection: Programmable Switches



(a) Throughput (b) Packet forwarding rate (c) Latency

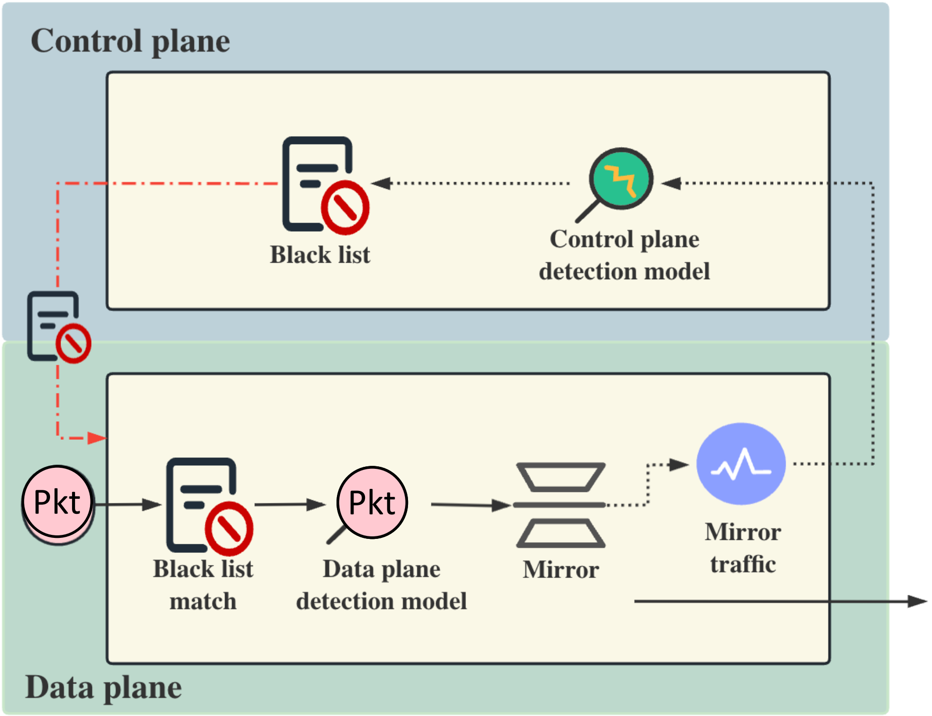
Programmable switches can achieve higher throughput, lower latency, and faster packet forwarding rate than control plane at equal cost.

Figure: Sigcomm-2021 Sailfish: Accelerating Cloud-Scale Multi-Tenant Multi-Service Gateways with Programmable Switches

General Idea and Key Challenges

General Idea

Preliminary screening of anomaly traffic at the switch, and reporting suspicious traffic to control plane.



General Idea and Key Challenges

Key Challenges

- It is challenging to extract and maintain the required flow features on the limited switch memory (e.g., 120 Mb SRAM);
- It is difficult to deploy an unsupervised model with both high anomaly recall and offloading capabilities on a programmable switch that only supports simple instructions and has limited resources.
- It is challenging to achieve a low false-positive rate using a high throughput deep model, since the control plane is a major throughput bottleneck.

Design of HorusEye

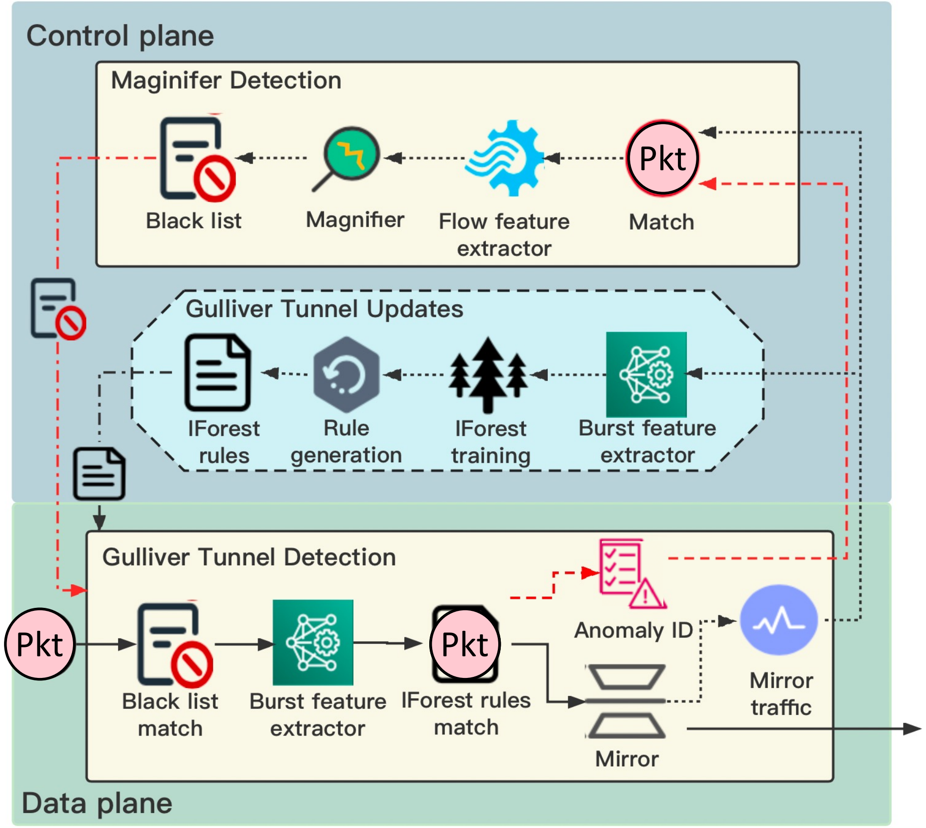
HorusEye

- **Gulliver Tunnel** (Data Plane):

  preliminary screening, alleviate the burden of the control plane.

- **Magnifier** (Control Plane) :

  further investigating, reduce the false positive rate.



Design of HorusEye

Gulliver Tunnel: Burst Feature Extractor

- Burst-level Feature:

  Identifying IoT behavior and reducing resource occupancy

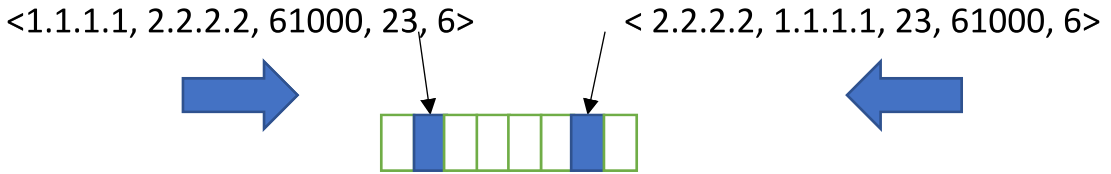
  - Total packet number & size
  - Burst segmentation

Burst feature extractor

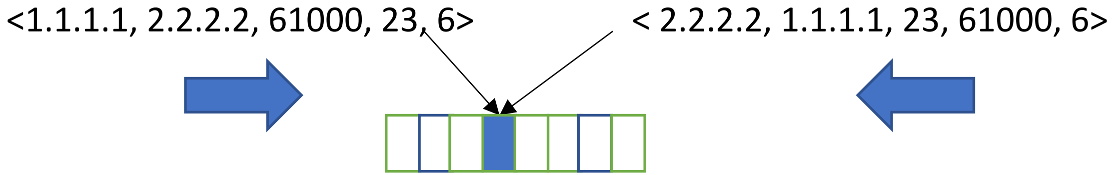
Design of HorusEye

Gulliver Tunnel: Burst Feature Extractor

- Bi-hash:
  - Five tuple hash ❌



$$hash(src\_IP, dst\_IP, src\_Port, dst\_Port, protocol)$$



  - Bi-hash **O(1)**✅

$$hash(dst\_IP, dst\_Port, protocol) \oplus hash(src\_IP, src\_Port, protocol)$$

Design of HorusEye

Gulliver Tunnel: Burst Feature Extractor

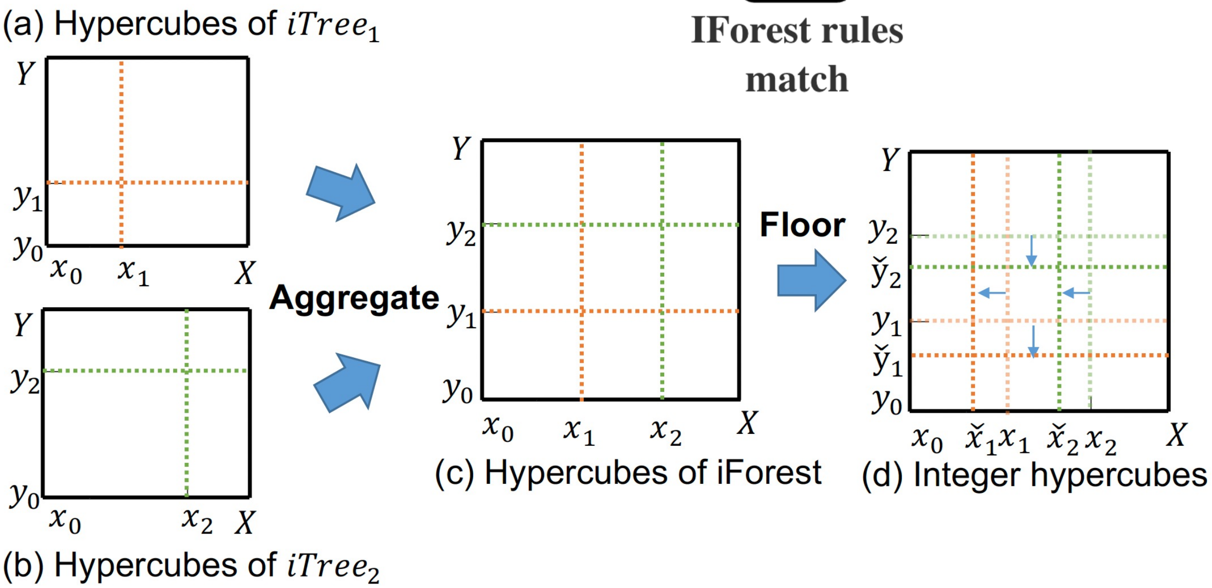
- Double hash table:
  - If the value conflicts in the first hash table, the algorithm executes the hash function on the first hash value and allocates it to the second hash table.
  - Design hash-check to judge whether the first hash table conflicts.

Design of HorusEye

Gulliver Tunnel: IForest Rule Generation

- Merge hypercubes of iTrees
- Label Consistency
- Integer boundary fine-tuning

IForest rules match



Design of HorusEye

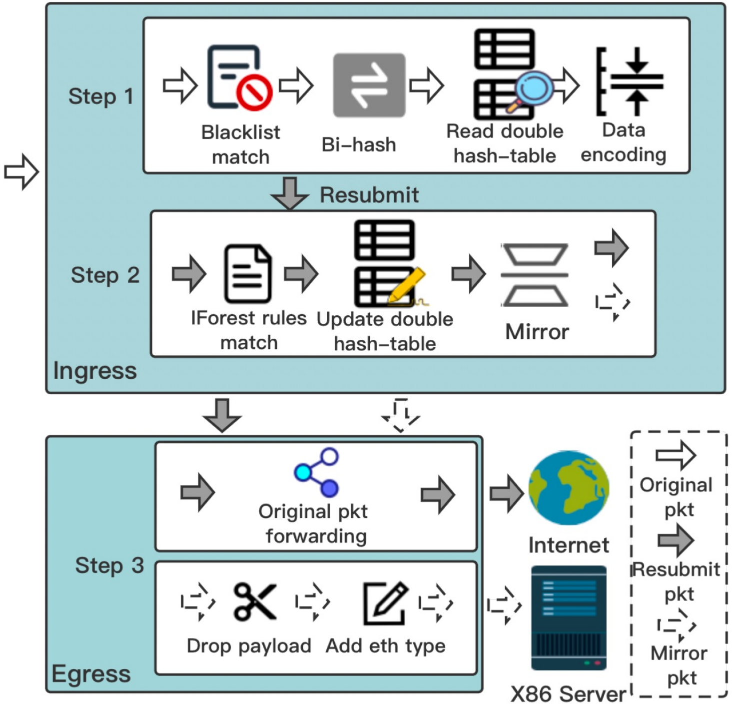
Gulliver Tunnel: Implementation on Programmable Switch

- Ingross:

  flow identification, burst feature extraction, storage and anomaly detection.

- Egross:

  packet mirroring and forwarding

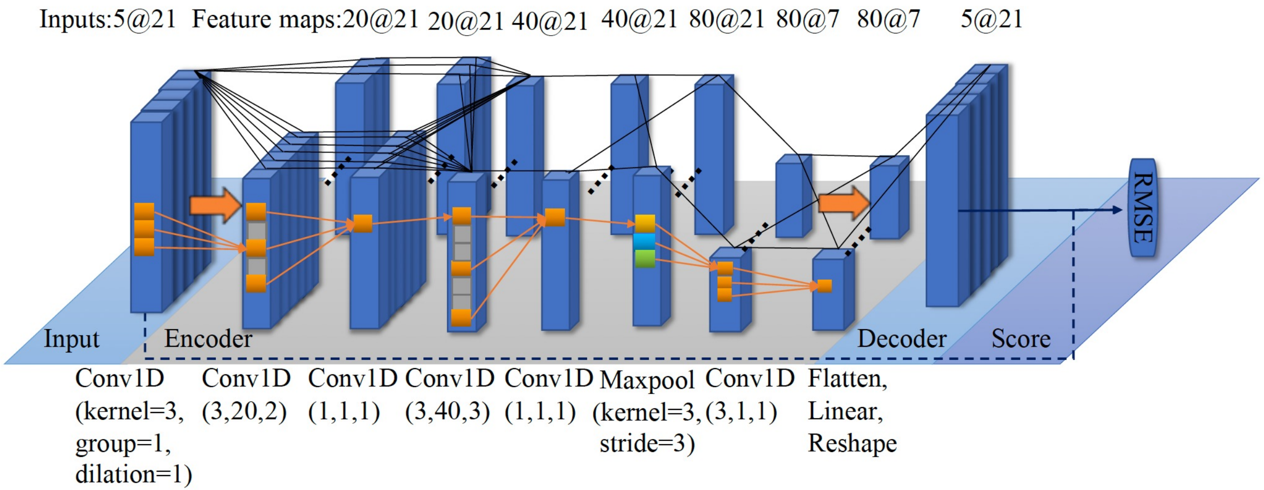


Design of HorusEye

Magnifier: An Asymmetric Lightweight Autoencoder

Magnifier

- ➢ Separable Convolution
- ➢ Dilated Convolution
- ➢ Model quantization



Evaluation

Performance of Rule Generation

- Convert iForest model into rule with nearly 100% label consistency.
- The number of burst-type whitelist rules does not change much.

Num of iTrees

Num of rules

ψ	t	$\#R_{burst}^{TCP}$	$\#R_{burst}^{UDP}$	$\#R_{port}^{TCP}$	$\#R_{port}^{UDP}$	$C(\%)$	#Enum
400	10	11	21	46803	39249	99.46	144947
	50	7	28	26202	34958	99.58	191687
	100	15	17	24866	36264	99.63	237137
	200	19	18	29063	31440	99.62	309197
1000	200	13	37	15549	9132	99.68	402977
2000		10	47	9120	1253	99.63	489392
5000		29	48	3460	125	99.66	612932

$$Consistency(C) = \frac{\sum_{i=1}^{N} \mathrm{I}(\mathrm{iForest}(x_i) = \mathrm{R}(x_i))}{N}$$

Evaluation

Hardware Performance

- The whole Gulliver Tunnel deployment only requires very little resources, i.e., 2.78% of TCAM and 9.90% of SRAM.

Table 3: Resource occupancy

Stage	Table	SRAM(kbit)	TCAM(kbit)
0	8	512	0
1	4	0	0
2	4	128	11
3	15	3456	0
4	18	4480	0
5	11	2944	0
6	4	384	44
7	1	128	22
8	1	128	11
9	0	0	0
10	0	0	0
11	0	0	0
Total	66	12160 (9.90%)	88 (2.78%)

Evaluation

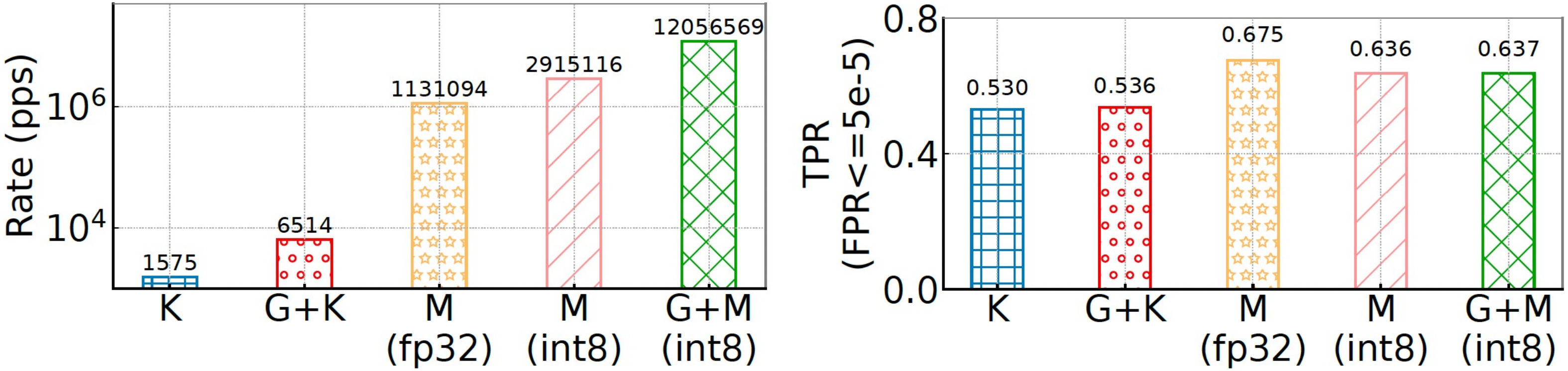
Detection Performance

- HorusEye has higher PR_AUC and TPR (with low FPR) than SOTA.

Dataset	Attack	Kitsune TPR ≤5e-5	Kitsune TPR ≤5e-4	Kitsune PR_{AUC}	Magnifier TPR ≤5e-5	Magnifier TPR ≤5e-4	Magnifier PR_{AUC}	HorusEye TPR ≤5e-5	HorusEye TPR ≤5e-4	HorusEye PR_{AUC}
[5] [18] [26]	Aidra	0.228	0.406	**0.716**	0.370	0.451	0.631	**0.383** ↑ 68.1%	**0.469** ↑ 15.3%	0.657 ↓ 8.33%
	Bashlite	0.605	0.677	**0.818**	0.698	0.730	0.806	**0.713** ↑ 17.7%	**0.735** ↑ 8.58%	0.817 ↓ 0.09%
	Mirai	0.105	0.183	0.949	0.962	**0.966**	0.976	**0.964** ↑ 815%	**0.966** ↑ 428%	**0.980** ↑ 3.23%
	Keylogging	**0.527**	0.527	0.602	**0.527**	**0.528**	0.779	**0.527** -	**0.528** ↑ 0.03%	**0.806** ↑ 33.9%
	Data theft	**0.508**	0.508	0.587	**0.508**	0.508	0.785	**0.508** -	**0.510** ↑ 0.04%	**0.810** ↑ 38.1%
	Service scan	0.217	0.274	0.833	0.318	0.358	0.915	**0.334** ↑ 53.6%	**0.363** ↑ 32.5%	**0.934** ↑ 12.1%
	OS scan	0.367	0.504	0.939	0.461	0.561	0.933	**0.498** ↑ 35.9%	**0.577** ↑ 14.5%	**0.946** ↑ 0.77%
	HTTP DDoS	0.055	0.211	0.779	0.235	0.382	0.927	**0.285** ↑ 421%	**0.408** ↑ 93.8%	**0.942** ↑ 21.0%
	TCP DDoS	0.903	0.936	0.969	**0.959**	**0.971**	**0.989**	0.903 -	0.912 ↓ 2.65%	0.929 ↓ 4.13%
	UDP DDoS	0.904	0.936	0.968	0.959	0.972	0.989	**0.965** ↑ 6.70%	**0.973** ↑ 4.02%	**0.990** ↑ 2.31%
	macro	0.442	0.516	0.816	0.600	0.643	0.873	**0.608** ↑ 37.6%	**0.644** ↑ 24.8%	**0.881** ↑ 7.97%
Ours	Mirai	0.000	0.012	0.636	0.196	0.412	0.842	**0.303** ↑ ∞	**0.424** ↑ 340%	**0.868** ↑ 36.4%
	Service scan	0.918	0.956	0.998	0.989	0.995	0.999	**0.991** ↑ 8.06%	**0.996** ↑ 4.05%	**1.000** ↑ 0.14%
	OS scan	0.617	0.810	0.994	0.943	0.983	**0.999**	**0.968** ↑ 56.9%	**0.985** ↑ 21.7%	**0.999** ↑ 0.54%
	TCP DDoS	0.994	0.996	**1.000**	**0.997**	**0.998**	**1.000**	**0.997** ↑ 0.39%	**0.998** ↑ 0.25%	**1.000** -
	UDP DDoS	0.995	0.997	**1.000**	0.997	**0.998**	**1.000**	**0.998** ↑ 0.27%	**0.998** ↑ 0.15%	**1.000** -
	macro	0.705	0.754	0.925	0.825	0.877	0.968	**0.852** ↑ 20.9%	**0.880** ↑ 16.7%	**0.973** ↑ 5.19%

Evaluation

Throughput and Detection Capability



(a) Thoughput of models (b) Macro TPR at FPR≤5e-5

- HorusEye has excellent throughput and anomaly detection capabilities.

Conclusion

- We propose HorusEye, an unsupervised Internet of Things (IoT) anomaly detection framework. It offloads abnormal traffic detection into the data plane, thereby freeing resources in the control plane to recheck results for higher accuracy.
- In the data plane, we propose a rule generation algorithm for iForest and a new flow feature extraction scheme, which implement the first unsupervised model that can reflect the limited resources of switches.
- On the control plane, we adopt a lightweight unsupervised model and a high-speed inference scheme.

Thank you!

For more details, welcome to follow our paper.

Contact us

Yutao Dong	:	dyt20@tsinghua.org.cn
Qing Li	:	liq@pcl.ac.cn
Kaidong Wu	:	w1450945445@gmail.com