Network and Distributed System Security (NDSS) Symposium 2023

Detecting Unknown Encrypted Malicious Traffic in Real Time via Flow Interaction Graph Analysis

Eunbee Hwang

서울대학교
SEOUL NATIONAL UNIVERSITY

Network Convergence & Security Lab

CONTENTS

Introduction

Keywords

Detecting Unknown Encrypted Malicious Traffic in Real Time via Flow Interaction Graph Analysis

Chuanpu Fu*, Qi Li†‡, Ke Xu*‡
*Department of Computer Science and Technology, Tsinghua University
†Institute for Network Sciences and Cyberspace, Tsinghua University ‡Zhongguancun Lab

***Abstract*—Nowadays traffic on the Internet has been widely encrypted to protect its confidentiality and privacy. However, traffic encryption is always abused by attackers to conceal their malicious behaviors. Since the encrypted malicious traffic has similar features to benign flows, it can easily evade traditional detection methods. Particularly, the existing encrypted malicious traffic detection methods are supervised and they rely on the prior knowledge of known attacks (e.g., labeled datasets). Detecting unknown encrypted malicious traffic in real time, which does not require prior domain knowledge, is still an open problem.**

existing malicious traffic detection methods. Different from plain-text malicious traffic, the encrypted traffic has similar features to benign flows and thus can evade existing machine learning (ML) based detection systems as well [2], [3], [62]. Particularly, the existing encrypted traffic detection methods are supervised, i.e., relying on the prior knowledge of known attacks, and can only detect attacks with known traffic patterns. They extract features of specific known attacks and use labeled datasets of known malicious traffic for model training [2],

Introduction

Keywords

1

Detecting Unknown Encrypted Malicious Traffic in Real Time via Flow Interaction Graph Analysis

Chuanpu Fu*, Qi Li†‡, Ke Xu*‡
*Department of Computer Science and Technology, Tsinghua University
†Institute for Network Sciences and Cyberspace, Tsinghua University ‡Zhongguancun Lab

***Abstract*—Nowadays traffic on the Internet has been widely encrypted to protect its confidentiality and privacy. However, traffic encryption is always abused by attackers to conceal their malicious behaviors. Since the encrypted malicious traffic has similar features to benign flows, it can easily evade traditional detection methods. Particularly, the existing encrypted malicious traffic detection methods are supervised and they rely on the prior knowledge of known attacks (e.g., labeled datasets). Detecting unknown encrypted malicious traffic in real time, which does not require prior domain knowledge, is still an open problem.**

existing malicious traffic detection methods. Different from plain-text malicious traffic, the encrypted traffic has similar features to benign flows and thus can evade existing machine learning (ML) based detection systems as well [2], [3], [62]. Particularly, the existing encrypted traffic detection methods are supervised, i.e., relying on the prior knowledge of known attacks, and can only detect attacks with known traffic patterns. They extract features of specific known attacks and use labeled datasets of known malicious traffic for model training [2],

Introduction

Keywords

Detecting Unknown Encrypted Malicious Traffic in Real Time via Flow Interaction Graph Analysis

Chuanpu Fu*, Qi Li†‡, Ke Xu*‡
*Department of Computer Science and Technology, Tsinghua University
†Institute for Network Sciences and Cyberspace, Tsinghua University ‡Zhongguancun Lab

***Abstract*—Nowadays traffic on the Internet has been widely encrypted to protect its confidentiality and privacy. However, traffic encryption is always abused by attackers to conceal their malicious behaviors. Since the encrypted malicious traffic has similar features to benign flows, it can easily evade traditional detection methods. Particularly, the existing encrypted malicious traffic detection methods are supervised and they rely on the prior knowledge of known attacks (e.g., labeled datasets). Detecting unknown encrypted malicious traffic in real time, which does not require prior domain knowledge, is still an open problem.**

existing malicious traffic detection methods. Different from plain-text malicious traffic, the encrypted traffic has similar features to benign flows and thus can evade existing machine learning (ML) based detection systems as well [2], [3], [62]. Particularly, the existing encrypted traffic detection methods are supervised, i.e., relying on the prior knowledge of known attacks, and can only detect attacks with known traffic patterns. They extract features of specific known attacks and use labeled datasets of known malicious traffic for model training [2],

Introduction

Keywords

- Unknown Encrypted
  - Encrypted malicious traffic detection is not well addressed
    - Similar features to benign flow
    - Diverse traffic patterns
  - The existing encrypted traffic detection methods are ***supervised***
    - Unable to detect encrypted malicious traffic with unknown patterns
    - Incapable of detecting both attacks constructed with and without encrypted traffic

Introduction

Keywords

- Unknown Encrypted
  - Encrypted malicious traffic detection is not well addressed
    - Low-rate
    - Diverse traffic patterns
  - The existing encrypted traffic detection methods are ***supervised***
    - Unable to detect encrypted malicious traffic with unknown patterns
    - Incapable of detecting both attacks constructed with and without encrypted traffic
- Real Time
  - Encrypted malicious traffic involves multiple attack steps with ***different flow interactions*** among attackers and victims
    - The interaction patterns are distinct from benign flow interaction patterns
  - A ***graph*** to capture various flow interaction patterns
    - The dependence explosion problem
  - Reduce the density of the graph inspired by the ***flow size distribution*** study

Introduction

Keywords

- The comparison with the existing methods of malicious traffic detection

Data Source Categories	Data Sources	Typical Methods	Data for Detection		Design Goals			Detection Performance	
			Unlabeled Datasets	Multi-Flow Features	Generic Detection	Realtime Detection	Unknown Attacks	Low Latency	High Throughput
Encrypted Traffic	Protocol Headers	TLS Extensions [16]	✗	✗	✗	✗	✗	✗	✓
		HTTPS Headers [3]	✗	✗	✗	✗	✗	✗	✗
	Related Flows	Time Series [76]	✗	✗	✗	✗	✗	✗	✗
		TLS Handshakes [2]	✗	✗	✗	✗	✗	✗	✗
		Flow Statistics [90]	✓	✗	✗	✓	✗	✗	✓
Plain-text and Encrypted Traffic	Network Logs	Intrusion Events [20]	✓	✗	✗	✗	✓	✗	✗
		Sampled Connections [8]	✓	✓[1]	✗	✓	✗	✗	✓
	Traffic Features	Per-Packet Features [56]	✓	✗	✗	✗	✓	✓	✗
		Per-Flow Features [5]	✗	✗	✗	✓	✗	✓	✗
		Flow Interaction Graph	✓	✓	✓	✓	✓	✓	✓

[1] Existing multi-flow features can only represent the features of specific flows, which cannot be used to represent complicated interaction patterns among various flows.

Introduction

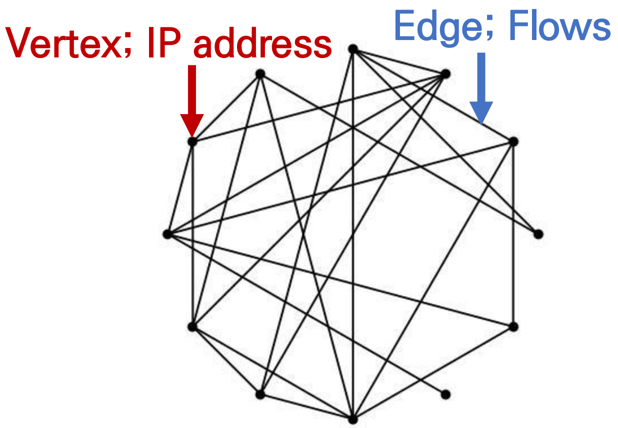
HyperVision

A real time detection system that aims

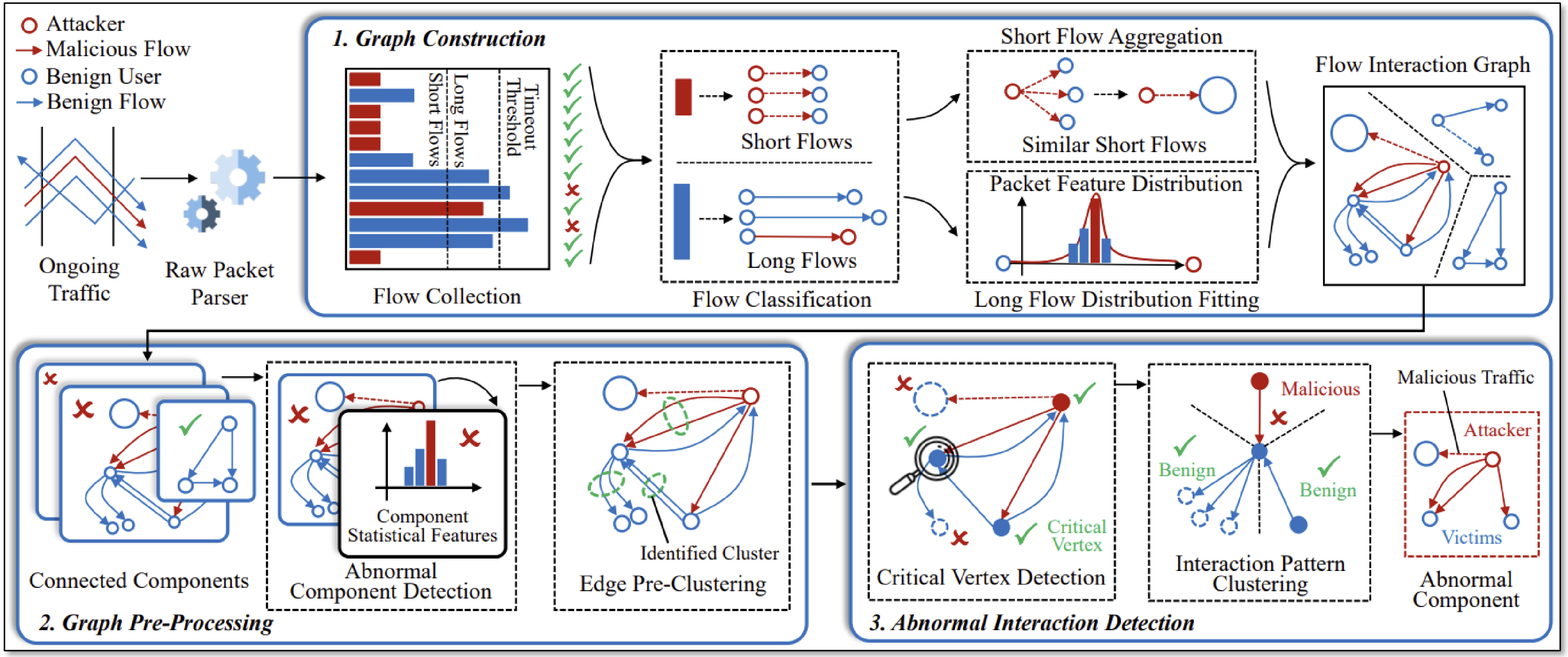
to capture footprints of encrypted malicious traffic by analyzing interaction patterns among flows

- Design Goals of HyperVison
  - Generic detection
  - Real time high-speed traffic processing
  - Unsupervised

- Graph in HyperVision



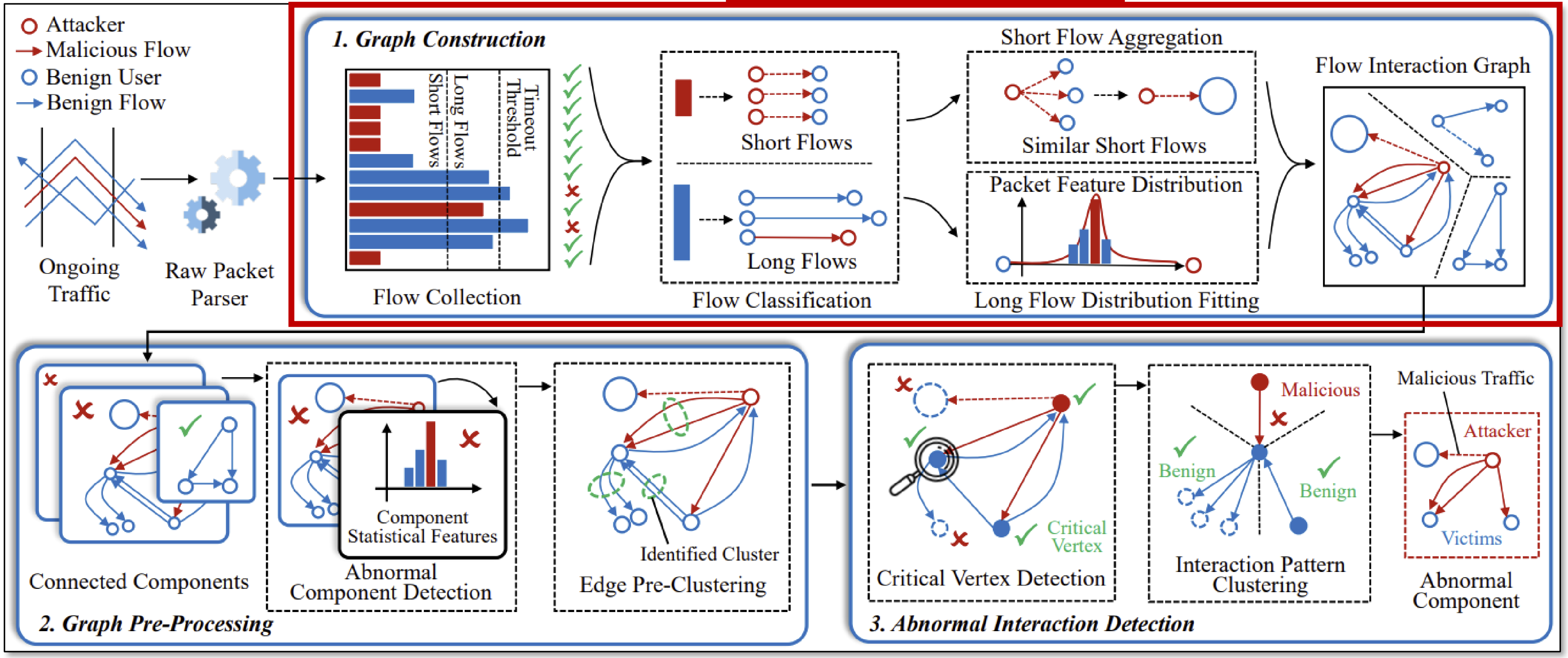
Overview



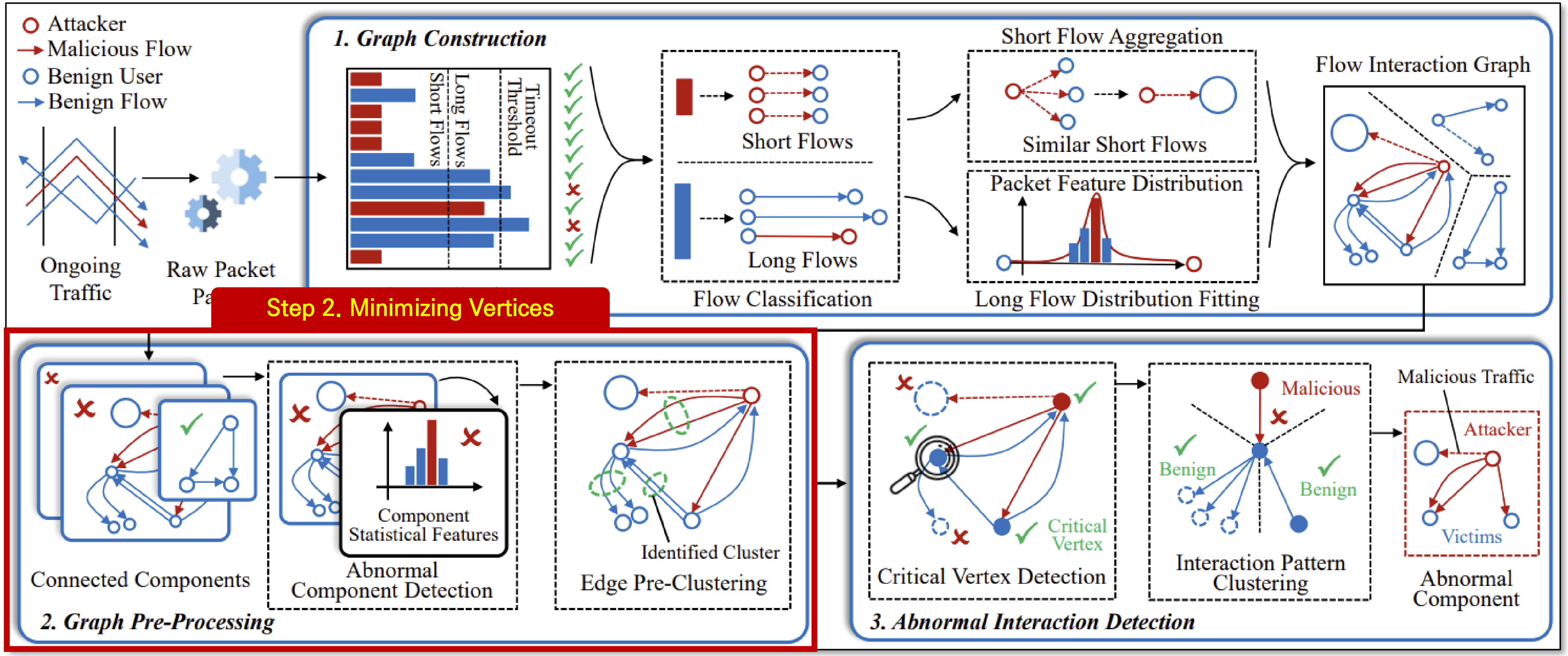
Attacker
Malicious Flow
Benign User
Benign Flow
Ongoing Traffic
Raw Packet Parser
1. Graph Construction
Short Flows
Long Flows
Timeout Threshold
Flow Collection
Short Flows
Long Flows
Flow Classification
Short Flow Aggregation
Similar Short Flows
Packet Feature Distribution
Long Flow Distribution Fitting
Flow Interaction Graph
Connected Components
Component Statistical Features
Abnormal Component Detection
Identified Cluster
Edge Pre-Clustering
2. Graph Pre-Processing
Critical Vertex
Critical Vertex Detection
Malicious
Benign
Benign
Interaction Pattern Clustering
Malicious Traffic
Attacker
Victims
Abnormal Component
3. Abnormal Interaction Detection

Overview

Step 1. Minimizing Edges

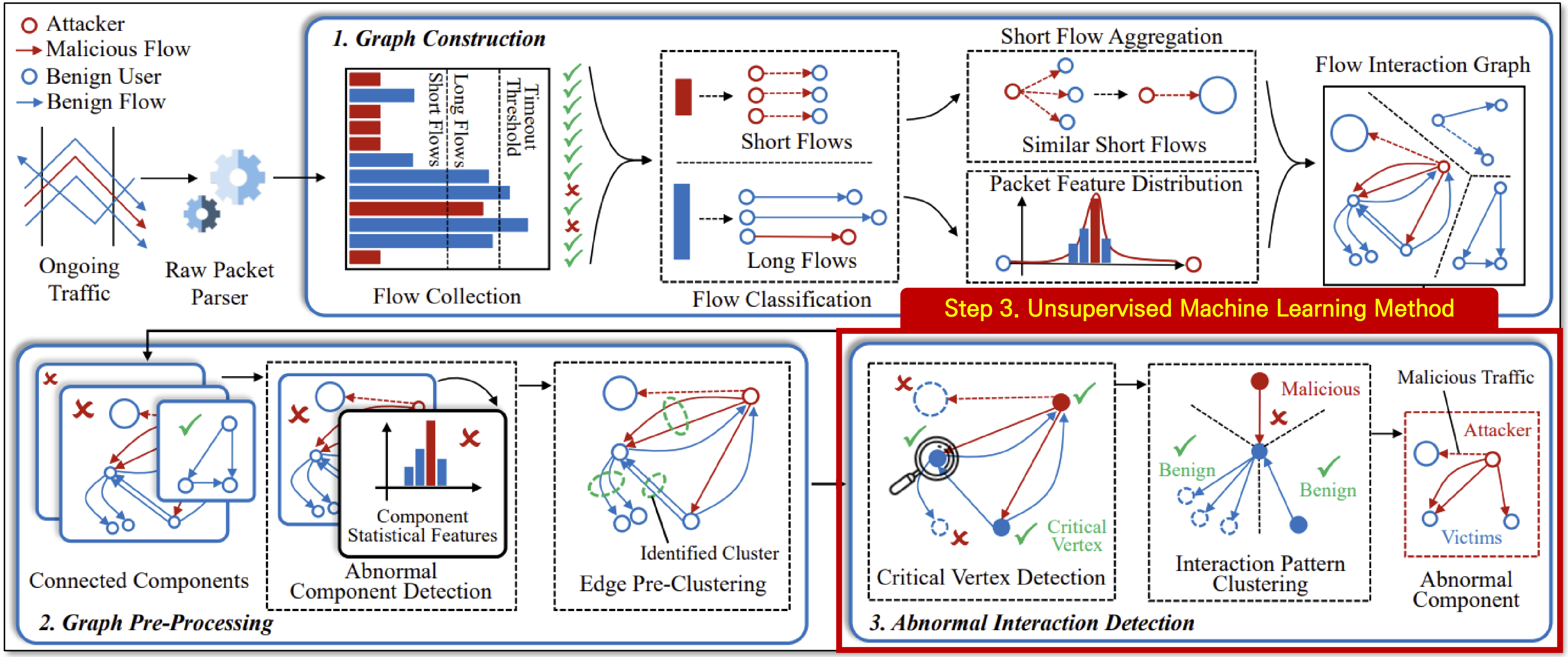


Overview



Attacker
Malicious Flow
Benign User
Benign Flow
Ongoing Traffic
Raw Packet
1. Graph Construction
Short Flows
Long Flows
Timeout Threshold
Short Flows
Long Flows
Flow Classification
Short Flow Aggregation
Similar Short Flows
Packet Feature Distribution
Long Flow Distribution Fitting
Flow Interaction Graph
Step 2. Minimizing Vertices
Connected Components
Component Statistical Features
Abnormal Component Detection
Identified Cluster
Edge Pre-Clustering
2. Graph Pre-Processing
Critical Vertex
Critical Vertex Detection
Malicious
Benign
Benign
Interaction Pattern Clustering
Malicious Traffic
Attacker
Victims
Abnormal Component
3. Abnormal Interaction Detection

Overview



Attacker
Malicious Flow
Benign User
Benign Flow
Ongoing Traffic
Raw Packet Parser
1. Graph Construction
Short Flows
Long Flows
Timeout Threshold
Flow Collection
Short Flows
Long Flows
Flow Classification
Short Flow Aggregation
Similar Short Flows
Packet Feature Distribution
Flow Interaction Graph
Step 3. Unsupervised Machine Learning Method
Connected Components
Component Statistical Features
Abnormal Component Detection
Identified Cluster
Edge Pre-Clustering
2. Graph Pre-Processing
Critical Vertex
Critical Vertex Detection
Malicious
Benign
Benign
Interaction Pattern Clustering
Malicious Traffic
Attacker
Victims
Abnormal Component
3. Abnormal Interaction Detection

Graph Construction

Objective of Graph Construction and Flow Classification

To efficiently analyze the flows on the internet,
need to avoid the dependency explosion among flows during the graph construction

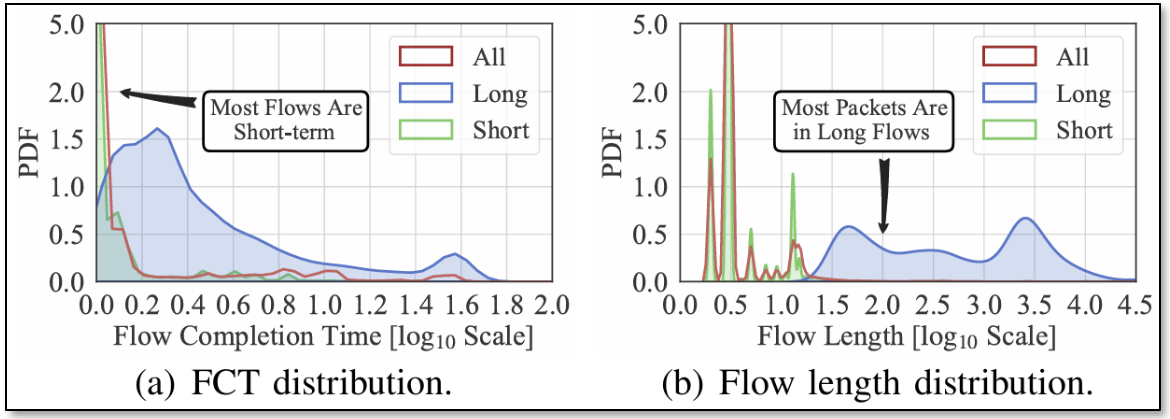
- Flow Classification
  - Eliminate timeout threshold flows
  - Classify the collected flows into short and long
    - Short flows 〈 Flow line
    - Long flows 〉 Flow line
  - Obtain per-packet features
    - Protocols, lengths, arrival intervals

Hyper-Parameter	Description	Value
PKT_TIMEOUT	Flow completion time threshold.	10.0s
FLOW_LINE	Flow classification threshold.	15
AGG_LINE	Flow aggregation threshold.	20

Graph Construction

Flow Classification

- The real-world flow features distribution of short and long flows
  - 5.52% flows have Flow Completion Time (FCT) 〉2.0s
  - 93.7% packets in the dataset are long flows
  - 97.64% proportion of short flows
  - 2.36% proportion of long flows



(a) FCT distribution. (b) Flow length distribution.

- The proportion difference inspired that different flow collection strategies are needed

Graph Construction

Short Flow Aggregation

Short flow aggregation to represent similar flows using one edge after the classification

- **Most short flows have almost the same per-packet feature sequence**
  - e.g. Repetitive SSH cracking

- **Requirements for short flow aggregation**
  - The flows have the same source and/or destination addresses
  - The flows have the same protocol type
  - The number of the flows is large enough
    - The threshold AGG_LINE

Graph Construction

Short Flow Aggregation

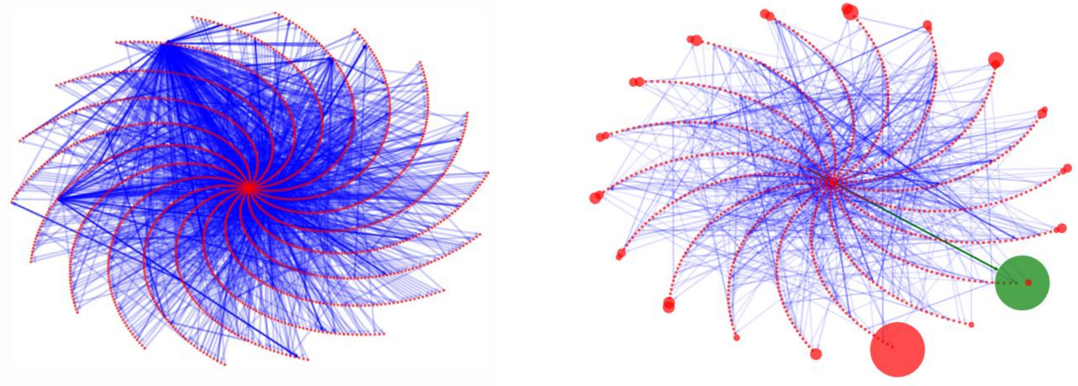
- **An edge for the short flow preserves one feature sequence and four tuples**
  - Per-packet features
    - Protocols, lengths, arrival intervals
  - Four tuples
    - Source and destination addresses, port numbers

- **Four types of edges associated with short flows exist on the graph**
  - Source address aggregated
  - Destination address aggregated
  - Both address aggregated
  - Without aggregation

Graph Construction

Short Flow Aggregation

- **Short flow aggregation to reduce the dense graph**
  - The diameter of a vertex indicates the number of addresses denoted by the vertex
  - The color indicates the repeated edges
  - The algorithm reduces 93.94% vertices and 94.04% edges
  - The edge highlighted in green indicates short flows exploiting a vulnerability



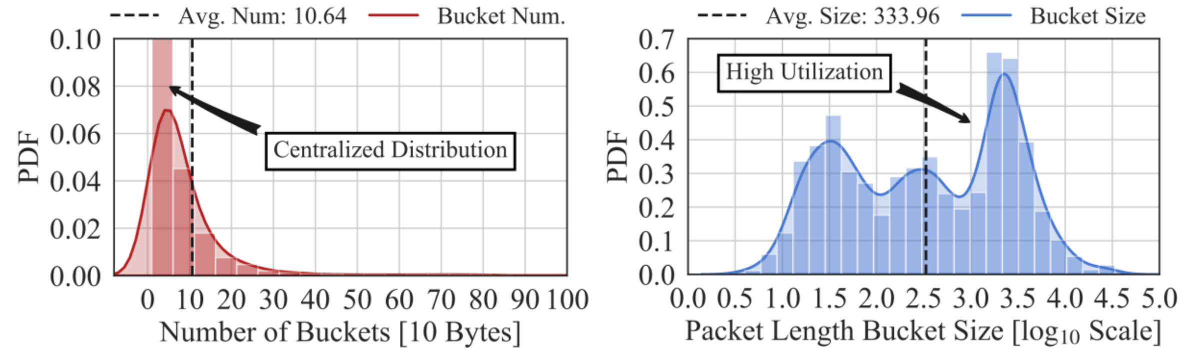
(a) Traditional flows as edges. (b) Short flow aggregation.

Graph Construction

Feature Distribution Fitting For Long Flows

- **Histogram is used to represent the per-packet feature distributions of a long flow**
  - A histogram to avoid preserving long per-packet feature sequences
  - A hash table for each per-packet feature sequence in each long flow

- **Most packets in the long flows have similar packet lengths and arrival intervals**
  - On average, only 11 buckets were used to fit the distribution of packet length, most of the buckets collected more than 200 packets



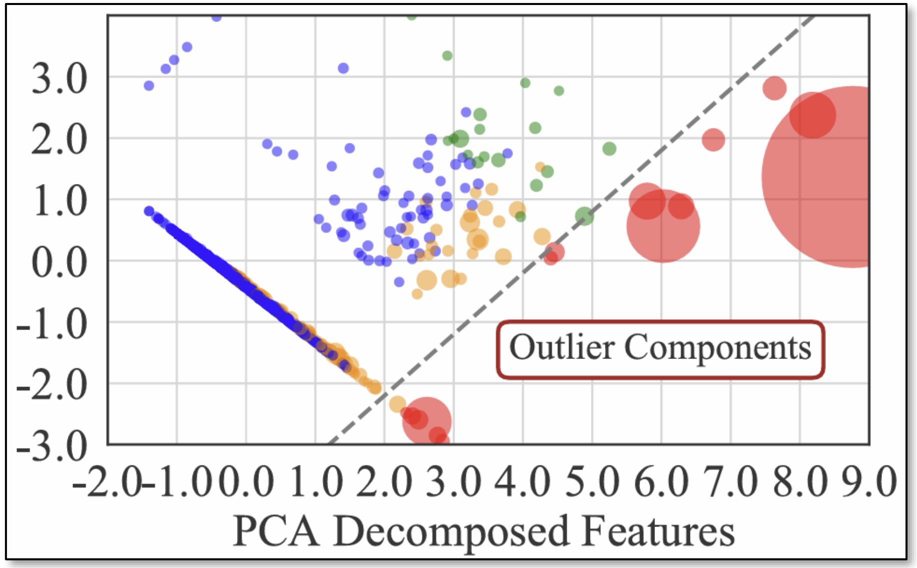
(a) Number of packet length buckets. (b) Maximum bucket size.

Graph Pre-Processing

Connectivity Analysis

- Split the graph by the components
  - Most components contain few edges with similar interaction patterns

- Five features to profile the components
  - The number of long flows
  - The number of short flows
  - The number of edges denoting short flows
  - The number of bytes in long flows
  - The number of bytes in short flows



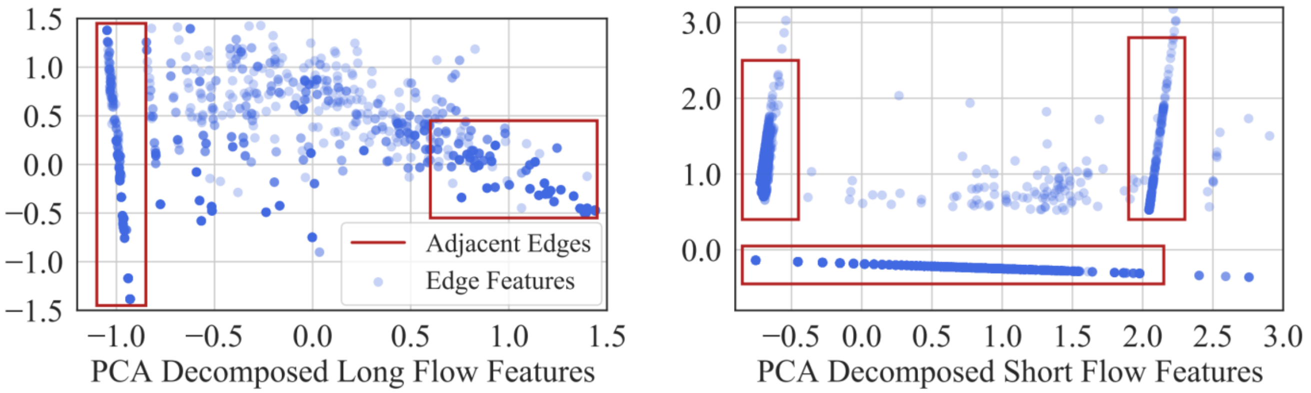
- DBSCAN for density based clustering

Graph Pre-Processing

Edge Pre-Clustering

- The abnormal components in the graph have massive vertices and edges
  - Graph Neural Network (GNN) for real time is impossible

- Extract eight and four graph structural features for the edges associated with short and long flow



(a) Adjacent long flows. (b) Adjacent short flows.

- Most edges are adjacent to massive similar edges in the feature space

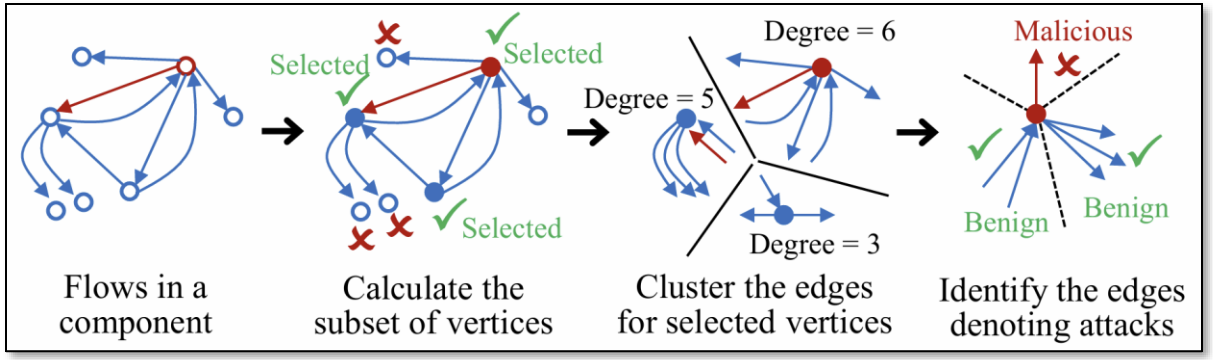
- DBSCAN for a pre-clustering

Malicious Traffic Detection

Identifying Critical Vertices

- **Cluster edges connected to the same critical vertex and detects outliers as malicious traffic**
  - Clustering all edges directly is not efficient to learn the interaction patterns of the traffic

- **Select a subset of all vertices in the connected component according to the following conditions**
  - The source and/or destination vertices of each edge in the component are in the subset
  - The number of selected vertices in the subset is minimized



Malicious Traffic Detection

Identifying Critical Vertices

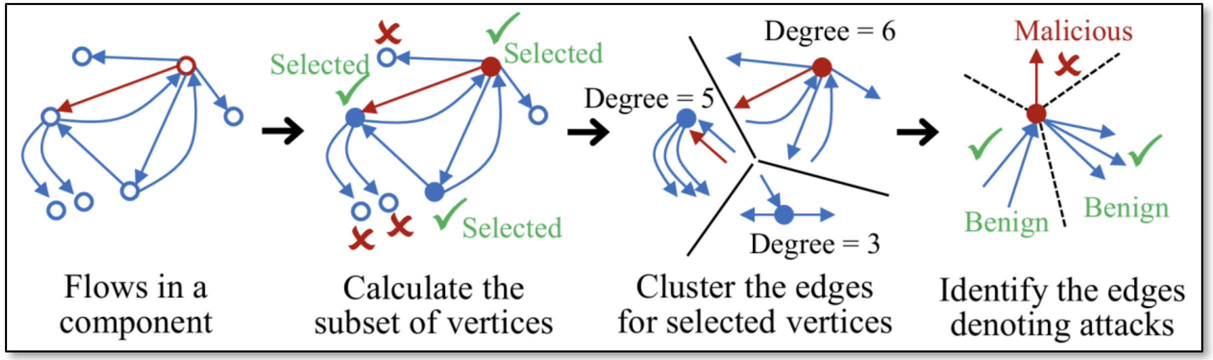
- **Finding such a subset of vertices is an optimization problem and equivalent to the *vertex cover problem*, which was proved to be NP Complete (NPC)**
  - All edges and vertices on each component were selected to solve the problem
  - Vertex cover problem was reformulated to Satisfiability Modulo Theories (SMT) problem
    - SMT can be effectively solved by using Z3 SMT solver

- **NPC can be solved in real time due to massive edge pre-clustering**

Malicious Traffic Detection

Edge Feature Clustering for Detection

- **To identify abnormal interaction patterns cluster the edges connected to each critical vertex**
  - Use the structural features and the flow features extracted from the per-packet feature sequences
  - Use the lightweight K-Means algorithm to cluster the edges
  - Calculate the clustering loss that indicates the degree of maliciousness for malicious flow detection



Theoretical Analysis

To analyze the information preserved
in the graph of HyperVision for graph learning based detection

- **Analysis**
  - Used metrics
    - The amount of information
    - The scale of data
    - The density of information
  - Typical types of flow recording modes
    - Idealized mode that records and stores the whole per-packet feature sequence
    - Event based mode
    - Sampling based mode

- **Key Results**
  - HyperVision maintains more information using the graph than the existing methods
  - HyperVision maintains near-optimal information using the graph
  - HyperVision has higher information density than the existing methods

Experimental Evaluation

Datasets

- **Background traffic**
  - Real world backbone network traffic datasets from the vantage-G of WIDE MAWI project in AS2500, Tokyo, Japan, Jan. ~ Jun. 2020

- **Malicious traffic**
  - Traditional brute force attack
    - To verify its generic detection
  - Encrypted flooding traffic
  - Encrypted web malicious traffic
  - Malware generated encrypted traffic

- **Metrics**
  - F1
    - F1 combines precision and recall into a single metric
  - AUC
    - AUC measures the performance of a binary classification model by plotting the true positive rate against the false positive rate

Experimental Evaluation

Overview of Accuracy Evaluation

- **HyperVision shows the highest accuracy**
  - Average F1 ranging between 0.927 and 0.978
  - Average AUC ranging between 0.974 and 0.993
  - HyperVision shows 35% and 13% improvements over the best accuracy of the baselines

Method	Metric	Traditional Attacks	Flooding Enc. Traffic	Enc. Web Attacks	Malware Traffic	Overall
Jaqen	AUC	0.913▼7%	0.782▼19%	N/A[1]	N/A	0.867▼12%
	F1	0.819▼16%	0.495▼46%	N/A	N/A	0.705▼26%
FlowLens	AUC	0.939▼4%	0.757▼22%	0.685▼30%	0.768▼22%	0.752▼36%
	F1	0.799▼18%	0.651▼29%	0.384▼59%	0.411▼57%	0.451▼41%
Whisper	AUC	0.951▼3%	0.932▼4%	0.958▼2%	0.648▼34%	0.752▼23%
	F1	0.705▼27%	0.461▼50%	0.546▼42%	0.357▼62%	0.407▼57%
Kitsune	AUC	0.748▼24%	-[2]	0.759▼22%	-	0.751▼23%
	F1	0.419▼57%	-	0.366▼61%	-	0.402▼58%
DeepLog	AUC	0.716▼27%	0.621▼26%	0.767▼22%	0.653▼34%	0.666▼32%
	F1	0.513▼47%	0.508▼45%	0.572▼40%	0.628▼34%	0.597▼37%
H.V.	AUC	0.988▲8%	0.974▲4%	0.985▲2%	0.993▲29%	0.988▲13%
	F1	0.978▲19%	0.927▲42%	0.957▲67%	0.970▲54%	0.960▲36%

[1] The results are N/A because Jaqen is designed for detection of volumetric attacks.
[2] - means that the average AUC is lower than 0.60, which is nearly the result of random guessing.

서 울 대 학 교
SEOUL NATIONAL UNIVERSITY

Network Convergence & Security Lab

Experimental Evaluation

Accuracy Evaluation

- Traditional Brute Force Attack

TABLE IV. DETECTION ACCURACY OF HYPERVISION AND THE BASELINES ON TRADITIONAL BRUTE FORCE ATTACKS.

Method	Metric	Brute Scanning							Amplification Attack							Source Spoofing DDoS			
		ICMP	NTP	SSH	SQL	DNS	HTTP	HTTPS	NTP	DNS	CharG.	SSDP	RIPv1	Mem.	CLDAP	SYN	RST	UDP	ICMP
Jaqen	AUC	0.9478	0.9989	0.9706	0.9851	0.9989	0.9774	0.9988	0.9822	0.9590	0.9860	0.9907	0.9011	0.9586	0.9537	0.9976	0.9985	0.9682	0.9995
	F1	0.9710	0.9356	0.9835	0.9924	0.9965	0.9884	0.9299	0.9457	0.8816	0.7986	0.7054	0.6549	0.8500	0.7931	0.9614	0.9236	0.5603	0.9861
FlowLens	AUC	0.9906	0.9021	0.9961	0.9993	0.9985	0.9874	0.9226	0.9784	0.8001	0.9998	0.9907	0.9833	0.9786	0.9993	0.9912	0.9918	0.9999	0.6351
	F1	0.9181	0.6528	0.8899	0.9996	0.9992	0.9936	0.9572	0.9794	0.7127	0.9991	0.8918	0.9889	0.9691	0.9986	0.8638	0.8173	0.9990	0.2632
Whisper	AUC	0.9499	0.9796	0.9562	0.9811	0.9832	0.9658	0.9827	0.9125	0.9645	0.8489	0.9662	0.9761	0.8954	0.9402	0.9563	0.9658	0.8956	0.9489
	F1	0.7004	0.7585	0.8869	0.7022	0.6748	0.7182	0.7489	0.8248	0.8435	0.4686	0.6195	0.6396	0.6956	0.8620	0.7587	0.8778	0.4857	0.4192
Kitsune	AUC	0.4522	0.7252	- [2]	0.7439	0.7228	0.7380	0.9614	0.7340	0.9994	0.9998	0.9989	0.4343	0.3993	0.7592	0.6210	0.4086	0.8534	0.7913
	F1	- [1]	0.3459	-	0.5033	0.4923	0.4798	0.4878	0.4461	0.5031	0.4609	0.4360	-	-	0.3838	0.3361	-	0.4539	0.4153
DeepLog	AUC	0.6717	0.8232	0.8377	0.6518	0.8261	0.6617	0.5545	0.7475	0.7428	0.7462	0.7458	0.7487	0.7480	0.7483	0.7564	0.2470	0.7012	0.7521
	F1	0.3566	0.4178	0.5266	0.2695	0.4050	0.2668	0.3653	0.5108	0.7201	0.5705	0.4313	0.3368	0.3321	0.3424	0.6074	-	0.4370	0.3428
H.V.	AUC	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9998	0.9989	0.9998	0.9969	0.9999	0.9999	0.9999	0.9996	0.9928
	F1	0.9939	0.9928	0.9960	0.9932	0.9831	0.9808	0.9892	0.9998	0.9998	0.9992	0.9956	0.9984	0.9983	0.9996	0.9993	0.9571	0.9981	0.9295

[1] We highlight the best accuracy in ● and the worst accuracy in ●. We mark - for the F1 when the AUC is lower than 0.50, which is the accuracy of random guessing.
[2] Kitsune did not finish the detection within 90 min (i.e., meaningless for defenses). And H.V. is short for HyperVision.

Experimental Evaluation

Accuracy Evaluation

- Traditional Brute Force Attack

TABLE IV. DETECTION ACCURACY OF HYPERVISION AND THE BASELINES ON TRADITIONAL BRUTE FORCE ATTACKS.

Method	Metric	Brute Scanning							Amplification Attack							Source Spoofing DDoS			
		ICMP	NTP	SSH	SQL	DNS	HTTP	HTTPS	NTP	DNS	CharG.	SSDP	RIPv1	Mem.	CLDAP	SYN	RST	UDP	ICMP
Jaqen	AUC	0.9478	0.9989	0.9706	0.9851	0.9989	0.9774	0.9988	0.9822	0.9590	0.9860	0.9907	0.9011	0.9586	0.9537	0.9976	0.9985	0.9682	0.9995
	F1	0.9710	0.9356	0.9835	0.9924	0.9965	0.9884	0.9299	0.9457	0.8816	0.7986	0.7054	0.6549	0.8500	0.7931	0.9614	0.9236	0.5603	0.9861
FlowLens	AUC	0.9906	0.9021	0.9961	0.9993	0.9985	0.9874	0.9226	0.9784	0.8001	0.9998	0.9907	0.9833	0.9786	0.9993	0.9912	0.9918	0.9999	0.6351
	F1	0.9181	0.6528	0.8899	0.9996	0.9992	0.9936	0.9572	0.9794	0.7127	0.9991	0.8918	0.9889	0.9691	0.9986	0.8638	0.8173	0.9990	0.2632
Whisper	AUC	0.9499	0.9796	0.9562	0.9811	0.9832	0.9658	0.9827	0.9125	0.9645	0.8489	0.9662	0.9761	0.8954	0.9402	0.9563	0.9658	0.8956	0.9489
	F1	0.7004	0.7585	0.8869	0.7022	0.6748	0.7182	0.7489	0.8248	0.8435	0.4686	0.6195	0.6396	0.6956	0.8620	0.7587	0.8778	0.4857	0.4192
Kitsune	AUC	0.4522	0.7252	- [2]	0.7439	0.7228	0.7380	0.9614	0.7340	0.9994	0.9998	0.9989	0.4343	0.3993	0.7592	0.6210	0.4086	0.8534	0.7913
	F1	- [1]	0.3459	-	0.5033	0.4923	0.4798	0.4878	0.4461	0.5031	0.4609	0.4360	-	-	0.3838	0.3361	-	0.4539	0.4153
DeepLog	AUC	0.6717	0.8232	0.8377	0.6518	0.8261	0.6617	0.5545	0.7475	0.7428	0.7462	0.7458	0.7487	0.7480	0.7483	0.7564	0.2470	0.7012	0.7521
	F1	0.3566	0.4178	0.5266	0.2695	0.4050	0.2668	[illegible]	[illegible]	[illegible]	[illegible]	0.4313	0.3368	0.3321	0.3424	0.6074	-	0.4370	0.3428
H.V.	AUC	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9998	0.9989	0.9998	0.9969	0.9999	0.9999	0.9999	0.9996	0.9928
	F1	0.9939	0.9928	0.9960	0.9932	0.9831	0.9808	0.9892	0.9998	0.9998	0.9992	0.9956	0.9984	0.9983	0.9996	0.9993	0.9571	0.9981	0.9295

[1] We highlight the best accuracy in ● and the worst accuracy in ●. We mark - for the F1 when the AUC is lower than 0.50, which is the accuracy of random guessing.
[2] Kitsune did not finish the detection within 90 min (i.e., meaningless for defenses). And H.V. is short for HyperVision.

0.992 ~ 0.999 AUC

Experimental Evaluation

Accuracy Evaluation

- Traditional Brute Force Attack

TABLE IV. DETECTION ACCURACY OF HYPERVISION AND THE BASELINES ON TRADITIONAL BRUTE FORCE ATTACKS.

Method	Metric	Brute Scanning							Amplification Attack							Source Spoofing DDoS			
		ICMP	NTP	SSH	SQL	DNS	HTTP	HTTPS	NTP	DNS	CharG.	SSDP	RIPv1	Mem.	CLDAP	SYN	RST	UDP	ICMP
Jaqen	AUC	0.9478	0.9989	0.9706	0.9851	0.9989	0.9774	0.9988	0.9822	0.9590	0.9860	0.9907	0.9011	0.9586	0.9537	0.9976	0.9985	0.9682	0.9995
	F1	0.9710	0.9356	0.9835	0.9924	0.9965	0.9884	0.9299	0.9457	0.8816	0.7986	0.7054	0.6549	0.8500	0.7931	0.9614	0.9236	0.5603	0.9861
FlowLens	AUC	0.9906	0.9021	0.9961	0.9993	0.9985	0.9874	0.9226	0.9784	0.8001	0.9998	0.9907	0.9833	0.9786	0.9993	0.9912	0.9918	0.9999	0.6351
	F1	0.9181	0.6528	0.8899	0.9996	0.9992	0.9936	0.9572	0.9794	0.7127	0.9991	0.8918	0.9889	0.9691	0.9986	0.8638	0.8173	0.9990	0.2632
Whisper	AUC	0.9499	0.9796	0.9562	0.9811	0.9832	0.9658	0.9827	0.9125	0.9645	0.8489	0.9662	0.9761	0.8954	0.9402	0.9563	0.9658	0.8956	0.9489
	F1	0.7004	0.7585	0.8869	0.7022	0.6748	0.7182	0.7489	0.8248	0.8435	0.4686	0.6195	0.6396	0.6956	0.8620	0.7587	0.8778	0.4857	0.4192
Kitsune	AUC	0.4522	0.7252	- [2]	0.7439	0.7228	0.7380	0.9614	0.7340	0.9994	0.9998	0.9989	0.4343	0.3993	0.7592	0.6210	0.4086	0.8534	0.7913
	F1	- [1]	0.3459	-	0.5033	0.4923	0.4798	0.4878	0.4461	0.5031	0.4609	0.4360	-	-	0.3838	0.3361	-	0.4539	0.4153
DeepLog	AUC	0.6717	0.8232	0.8377	0.6518	0.8261	0.6617	0.5545	0.7475	0.7428	0.7462	0.7458	0.7487	0.7480	0.7483	0.7564	0.2470	0.7012	0.7521
	F1	0.3566	0.4178	0.5266	0.2695	0.4050	0.2668	0.3653	0.5108	0.7201	0.5705	0.4313	0.3368	0.3321	0.3424	0.6074	-	0.4370	0.3428
H.V.	AUC	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	[illegible]	[illegible]	[illegible]	[illegible]8	0.9989	0.9998	0.9969	0.9999	0.9999	0.9999	0.9996	0.9928
	F1	0.9939	0.9928	0.9960	0.9932	0.9831	0.9808	0.9892	0.9998	0.9998	0.9992	0.9956	0.9984	0.9983	0.9996	0.9993	0.9571	0.9981	0.9295

0.929 ~ 0.999 F1

[1] We highlight the best accuracy in ● and the worst accuracy in ●. We mark - for the F1 when the AUC is lower than 0.50, which is the accuracy of random guessing.
[2] Kitsune did not finish the detection within 90 min (i.e., meaningless for defenses). And H.V. is short for HyperVision.

Experimental Evaluation

Accuracy Evaluation

- Traditional Brute Force Attack

TABLE IV. DETECTION ACCURACY OF HYPERVISION AND THE BASELINES ON TRADITIONAL BRUTE FORCE ATTACKS.

H.V. shows 56.3% AUC Improvement

Method	Metric	Brute Scanning							Amplification Attack							Source Spoofing DDoS			
		ICMP	NTP	SSH	SQL	DNS	HTTP	HTTPS	NTP	DNS	CharG.	SSDP	RIPv1	Mem.	CLDAP	SYN	RST	UDP	ICMP
Jaqen	AUC	0.9478	0.9989	0.9706	0.9851	0.9989	0.9774	0.9988	0.9822	0.9590	0.9860	0.9907	0.9011	0.9586	0.9537	0.9976	0.9985	0.9682	0.9995
	F1	0.9710	0.9356	0.9835	0.9924	0.99[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]549	0.8500	0.7931	0.9614	0.9236	0.5603	0.9861
FlowLens	AUC	0.9906	0.9021	0.9961	0.9993	0.9985	0.9874	0.9226	0.9784	0.8001	0.9998	0.9907	0.9833	0.9786	0.9993	0.9912	0.9918	0.9999	0.6351
	F1	0.9181	0.6528	0.8899	0.9996	0.9992	0.9936	0.9572	0.9794	0.7127	0.9991	0.8918	0.9889	0.9691	0.9986	0.8638	0.8173	0.9990	0.2632
Whisper	AUC	0.9499	0.9796	0.9562	0.9811	0.9832	0.9658	0.9827	0.9125	0.9645	0.8489	0.9662	0.9761	0.8954	0.9402	0.9563	0.9658	0.8956	0.9489
	F1	0.7004	0.7585	0.8869	0.7022	0.6748	0.7182	0.7489	0.8248	0.8435	0.4686	0.6195	0.6396	0.6956	0.8620	0.7587	0.8778	0.4857	0.4192
Kitsune	AUC	0.4522	0.7252	- [2]	0.7439	0.7228	0.7380	0.9614	0.7340	0.9994	0.9998	0.9989	0.4343	0.3993	0.7592	0.6210	0.4086	0.8534	0.7913
	F1	- [1]	0.3459	-	0.5033	0.4923	0.4798	0.4878	0.4461	0.5031	0.4609	0.4360	-	-	0.3838	0.3361	-	0.4539	0.4153
DeepLog	AUC	0.6717	0.8232	0.8377	0.6518	0.8261	0.6617	0.5545	0.7475	0.7428	0.7462	0.7458	0.7487	0.7480	0.7483	0.7564	0.2470	0.7012	0.7521
	F1	0.3566	0.4178	0.5266	0.2695	0.4050	0.2668	0.3653	0.5108	0.7201	0.5705	0.4313	0.3368	0.3321	0.3424	0.6074	-	0.4370	0.3428
H.V.	AUC	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9998	0.9989	0.9998	0.9969	0.9999	0.9999	0.9999	0.9996	0.9928
	F1	0.9939	0.9928	0.9960	0.9932	0.9831	0.9808	0.9892	0.9998	0.9998	0.9992	0.9956	0.9984	0.9983	0.9996	0.9993	0.9571	0.9981	0.9295

[1] We highlight the best accuracy in ● and the worst accuracy in ●. We mark - for the F1 when the AUC is lower than 0.50, which is the accuracy of random guessing.
[2] Kitsune did not finish the detection within 90 min (i.e., meaningless for defenses). And H.V. is short for HyperVision.

Experimental Evaluation

Accuracy Evaluation

- Traditional Brute Force Attack

TABLE IV. DETECTION ACCURACY OF HYPERVISION AND THE BASELINES ON TRADITIONAL BRUTE FORCE ATTACKS.

Method	Metric	Brute Scanning							Amplification Attack							Source Spoofing DDoS			
		ICMP	NTP	SSH	SQL	DNS	HTTP	HTTPS	NTP	DNS	CharG.	SSDP	RIPv1	Mem.	CLDAP	SYN	RST	UDP	ICMP
Jaqen	AUC	0.9478	0.9989	0.9706	0.9851	0.9989	0.9774	0.9988	0.9822	0.9590	0.9860	0.9907	0.9011	0.9586	0.9537	0.9976	0.9985	0.9682	0.9995
	F1	0.9710	0.9356	0.9835	0.9924	0.9965	0.9884	0.9299	0.9457	0.8816	0.7986	0.7054	0.6549	0.8500	0.7931	0.9614	0.9236	0.5603	0.9861
FlowLens	AUC	0.9906	0.9021	0.9961	0.9993	0.9985	0.9874	0.9226	0.9784	0.8001	0.9998	0.9907	0.9833	0.9786	0.9993	0.9912	0.9918	0.9999	0.6351
	F1	0.9181	0.6528	0.8899	0.9996	0.99[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]889	0.9691	0.9986	0.8638	0.8173	0.9990	0.2632
Whisper	AUC	0.9499	0.9796	0.9562	0.9811	0.9832	0.9658	0.9827	0.9125	0.9645	0.8489	0.9662	0.9761	0.8954	0.9402	0.9563	0.9658	0.8956	0.9489
	F1	0.7004	0.7585	0.8869	0.7022	0.6748	0.7182	0.7489	0.8248	0.8435	0.4686	0.6195	0.6396	0.6956	0.8620	0.7587	0.8778	0.4857	0.4192
Kitsune	AUC	0.4522	0.7252	- [2]	0.7439	0.7228	0.7380	0.9614	0.7340	0.9994	0.9998	0.9989	0.4343	0.3993	0.7592	0.6210	0.4086	0.8534	0.7913
	F1	- [1]	0.3459	-	0.5033	0.4923	0.4798	0.4878	0.4461	0.5031	0.4609	0.4360	-	-	0.3838	0.3361	-	0.4539	0.4153
DeepLog	AUC	0.6717	0.8232	0.8377	0.6518	0.8261	0.6617	0.5545	0.7475	0.7428	0.7462	0.7458	0.7487	0.7480	0.7483	0.7564	0.2470	0.7012	0.7521
	F1	0.3566	0.4178	0.5266	0.2695	0.4050	0.2668	0.3653	0.5108	0.7201	0.5705	0.4313	0.3368	0.3321	0.3424	0.6074	-	0.4370	0.3428
H.V.	AUC	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9998	0.9989	0.9998	0.9969	0.9999	0.9999	0.9999	0.9996	0.9928
	F1	0.9939	0.9928	0.9960	0.9932	0.9831	0.9808	0.9892	0.9998	0.9998	0.9992	0.9956	0.9984	0.9983	0.9996	0.9993	0.9571	0.9981	0.9295

[1] We highlight the best accuracy in ● and the worst accuracy in ●. We mark - for the F1 when the AUC is lower than 0.50, which is the accuracy of random guessing.
[2] Kitsune did not finish the detection within 90 min (i.e., meaningless for defenses). And H.V. is short for HyperVision.

H.V. shows 11.6% AUC Improvement

Experimental Evaluation

Accuracy Evaluation

- Traditional Brute Force Attack

TABLE IV. DETECTION ACCURACY OF HYPERVISION AND THE BASELINES ON TRADITIONAL BRUTE FORCE ATTACKS.

Method	Metric	Brute Scanning							Amplification Attack							Source Spoofing DDoS			
		ICMP	NTP	SSH	SQL	DNS	HTTP	HTTPS	NTP	DNS	CharG.	SSDP	RIPv1	Mem.	CLDAP	SYN	RST	UDP	ICMP
Jaqen	AUC	0.9478	0.9989	0.9706	0.9851	0.9989	0.9774	0.9988	0.9822	0.9590	0.9860	0.9907	0.9011	0.9586	0.9537	0.9976	0.9985	0.9682	0.9995
	F1	0.9710	0.9356	0.9835	0.9924	0.9965	0.9884	0.9299	0.9457	0.8816	0.7986	0.7054	0.6549	0.8500	0.7931	0.9614	0.9236	0.5603	0.9861
FlowLens	AUC	0.9906	0.9021	0.9961	0.9993	0.9985	0.9874	0.9226	0.9784	0.8001	0.9998	0.9907	0.9833	0.9786	0.9993	0.9912	0.9918	0.9999	0.6351
	F1	0.9181	0.6528	0.8899	0.9996	0.9992	0.9936	0.9572	0.9794	0.7127	0.9991	0.8918	0.9889	0.9691	0.9986	0.8638	0.8173	0.9990	0.2632
Whisper	AUC	0.9499	0.9796	0.9562	0.9811	0.9832	0.9658	0.9827	0.9125	0.9645	0.8489	0.9662	0.9761	0.8954	0.9402	0.9563	0.9658	0.8956	0.9489
	F1	0.7004	0.7585	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	.8620	0.7587	0.8778	0.4857	0.4192
Kitsune	AUC	0.4522	0.7252	- [2]	0.7439	0.7228	0.7380	0.9614	0.7340	0.9994	0.9998	0.9989	0.4343	0.3993	0.7592	0.6210	0.4086	0.8534	0.7913
	F1	- [1]	0.3459	-	0.5033	0.4923	0.4798	0.4878	0.4461	0.5031	0.4609	0.4360	-	-	0.3838	0.3361	-	0.4539	0.4153
DeepLog	AUC	0.6717	0.8232	0.8377	0.6518	0.8261	0.6617	0.5545	0.7475	0.7428	0.7462	0.7458	0.7487	0.7480	0.7483	0.7564	0.2470	0.7012	0.7521
	F1	0.3566	0.4178	0.5266	0.2695	0.4050	0.2668	0.3653	0.5108	0.7201	0.5705	0.4313	0.3368	0.3321	0.3424	0.6074	-	0.4370	0.3428
H.V.	AUC	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9999	0.9998	0.9989	0.9998	0.9969	0.9999	0.9999	0.9999	0.9996	0.9928
	F1	0.9939	0.9928	0.9960	0.9932	0.9831	0.9808	0.9892	0.9998	0.9998	0.9992	0.9956	0.9984	0.9983	0.9996	0.9993	0.9571	0.9981	0.9295

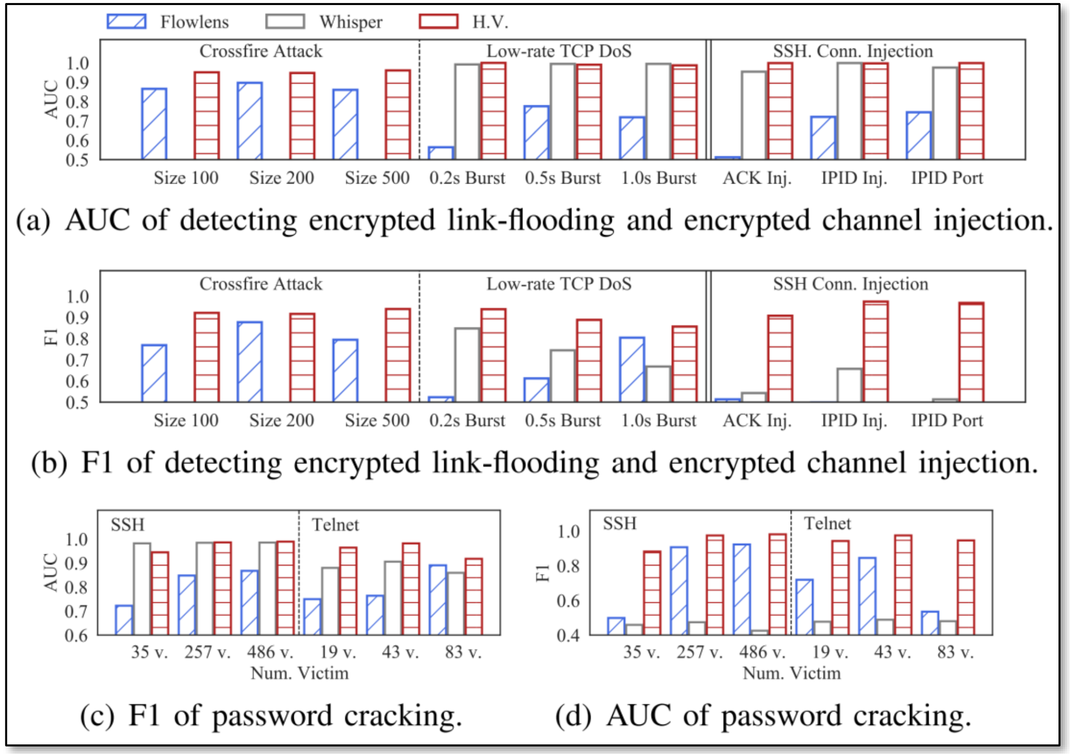
[1] We highlight the best accuracy in ● and the worst accuracy in ●. We mark - for the F1 when the AUC is lower than 0.50, which is the accuracy of random guessing.
[2] Kitsune did not finish the detection within 90 min (i.e., meaningless for defenses). And H.V. is short for HyperVision.

Kitsune and DeepLog cannot afford high speed backbone traffic

Experimental Evaluation

Accuracy Evaluation

- Encrypted Flooding Traffic
  - HyperVision achieves 0.856 ~ 0.981 F1 and 0.917 ~ 0.998 AUC
    - 58.7% F1 and 25.3% AUC accuracy improvement over the baselines
  - HyperVision can accurately detect the link flooding traffic
  - HyperVision can identify slow and persisted password attempts for the channels
    - HyperVision maintains the interaction patterns of attackers using the graph



(a) AUC of detecting encrypted link-flooding and encrypted channel injection.

(b) F1 of detecting encrypted link-flooding and encrypted channel injection.

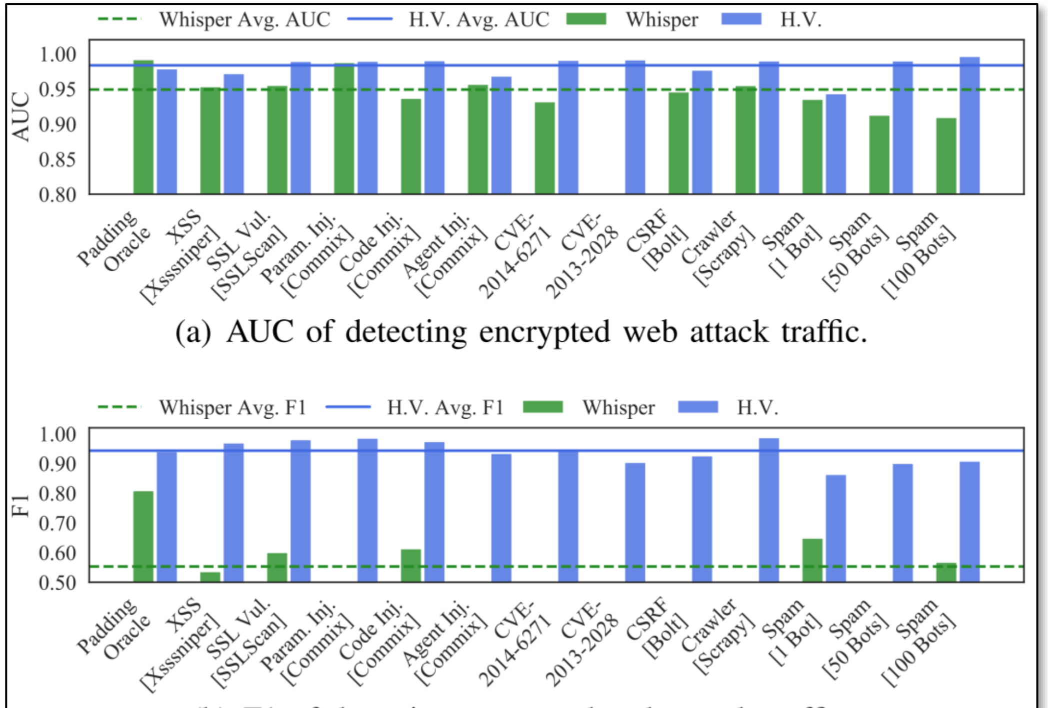
(c) F1 of password cracking.

(d) AUC of password cracking.

Experimental Evaluation

Accuracy Evaluation

- **Encrypted Web Malicious Traffic**
  - HyperVision achieves 0.985 average AUC and 0.957 average F1



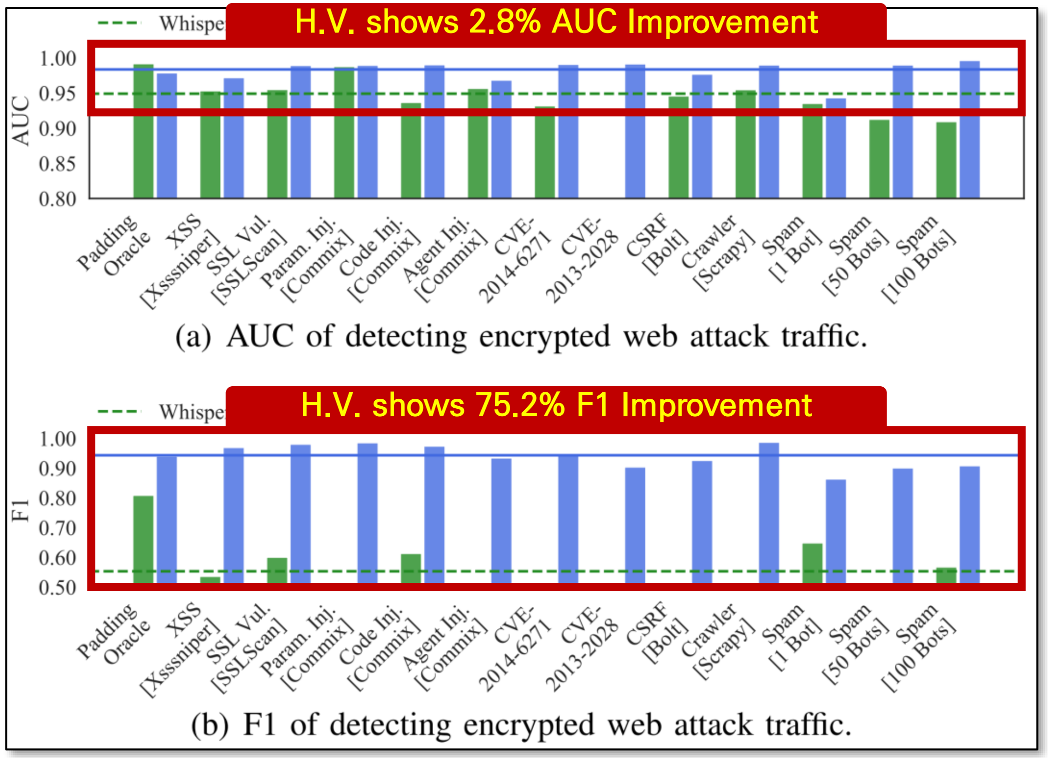
(a) AUC of detecting encrypted web attack traffic.

(b) F1 of detecting encrypted web attack traffic.

Experimental Evaluation

Accuracy Evaluation

- Encrypted Web Malicious Traffic
  - HyperVision achieves 0.985 average AUC and 0.957 average F1



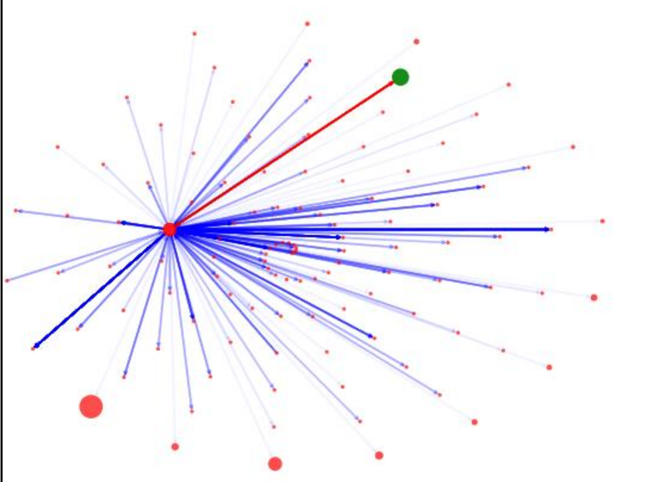
(a) AUC of detecting encrypted web attack traffic.

(b) F1 of detecting encrypted web attack traffic.

Experimental Evaluation

Accuracy Evaluation

- **Encrypted Web Malicious Traffic**
  - HyperVision achieves 0.985 average AUC and 0.957 average F1
  - The flow based ML detection cannot detect web encrypted malicious traffic
    - Single flow patterns are almost same to benign web access flows
  - HyperVision can accurately detect the encrypted web malicious traffic, because it captures the traffic from the frequent interactions

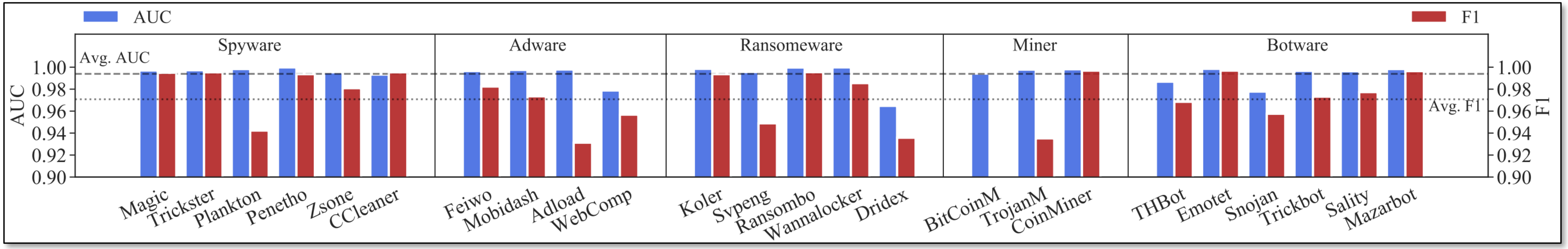


(c) XSS detection.

Experimental Evaluation

Accuracy Evaluation

- Encrypted Malware Traffic



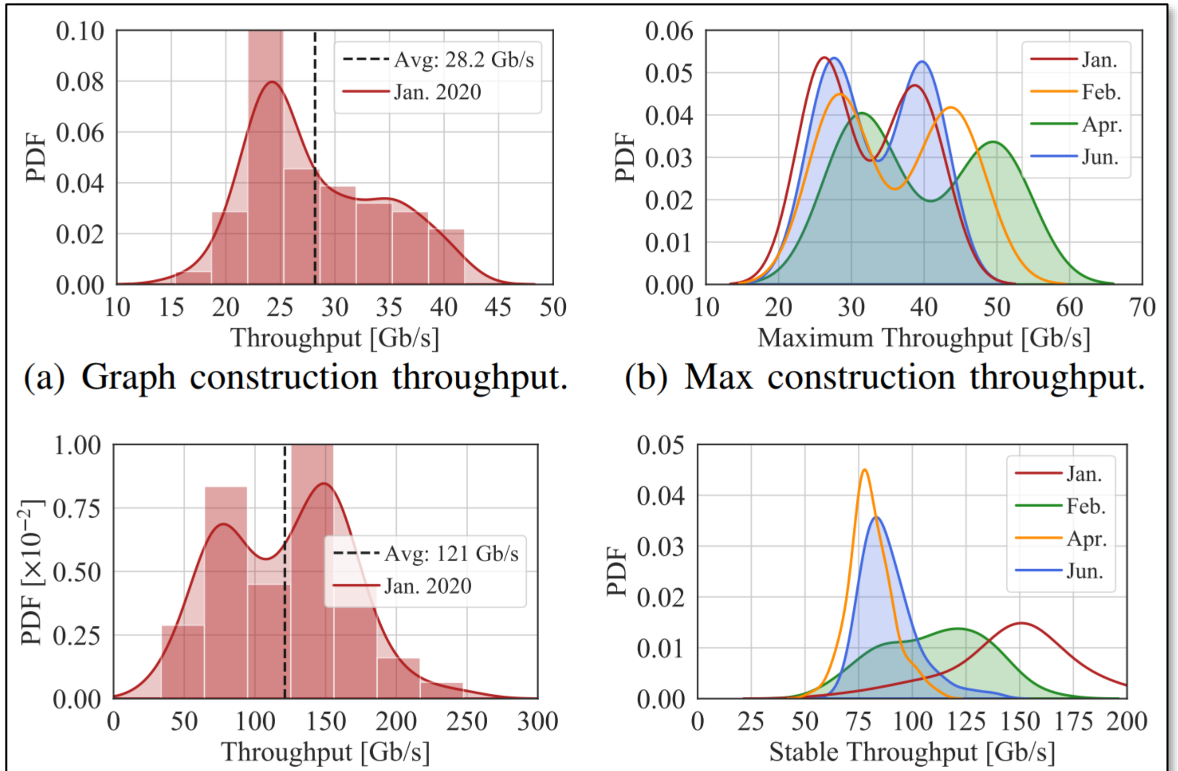
  - Encrypted malware traffic is hard to detect for the baselines, because it is slow and persistent

  - HyperVision accurately detects the malware campaigns at least 0.964 AUC and 0.891 F1

Experimental Evaluation

Performance Results

- Throughput
  - Graph construction throughput
    - 28.21 Gb/s
  - Max construction throughput
    - 32.43 ~ 39.71 Gb/s
  - Graph detection throughput
    - 121.64 Gb/s
  - Stable detection throughput
    - 80.6 ~ 148.9 Gb/s



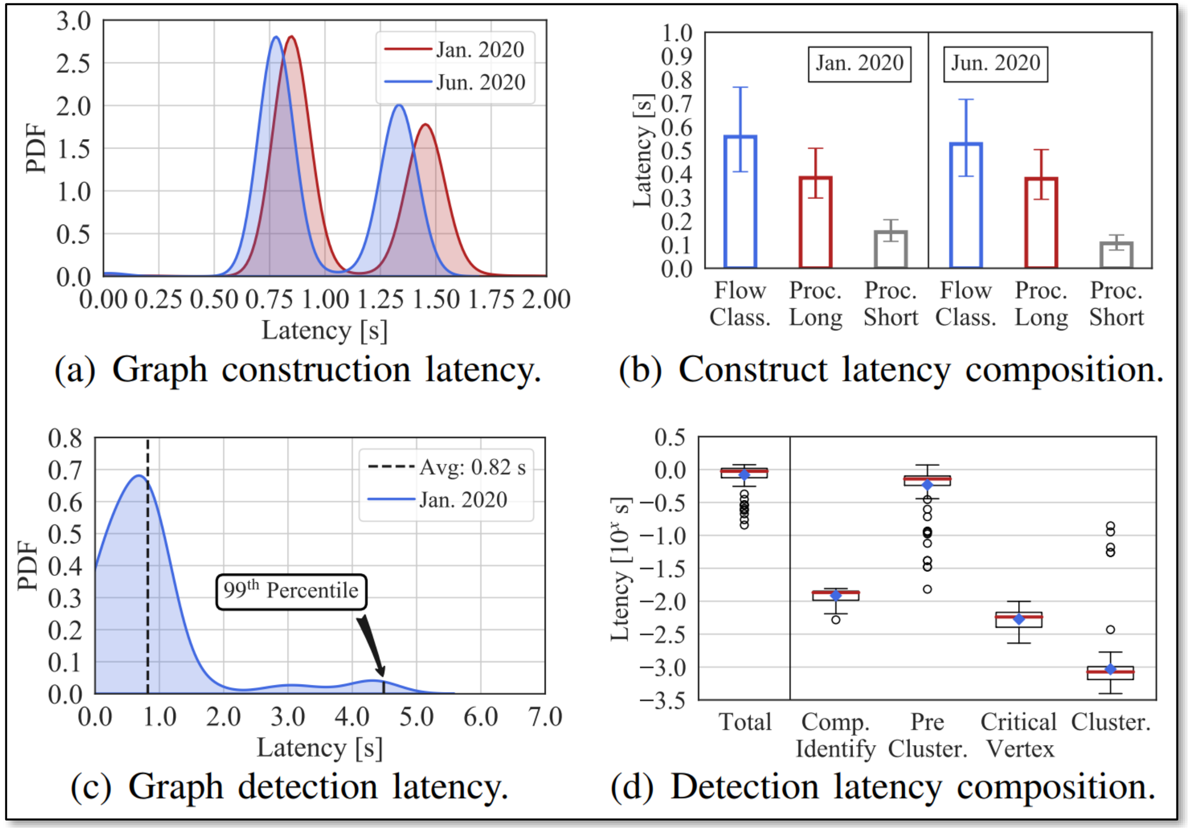
(a) Graph construction throughput. (b) Max construction throughput.

(c) Graph detection throughput. (d) Stable detection throughput.

Experimental Evaluation

Performance Results

- **Latency**
  - HyperVision has 1.09 ~ 1.04s average construction latency with an upper bound of 1.93s
    - The Receive Side Scaling (RSS) on the Intel NIC is unbalanced on the threads
  - Construct latency composition
    - Flow classification 50.95%
    - Short flow aggregation 35.03%
    - Long flow distribution fitting 14.0%

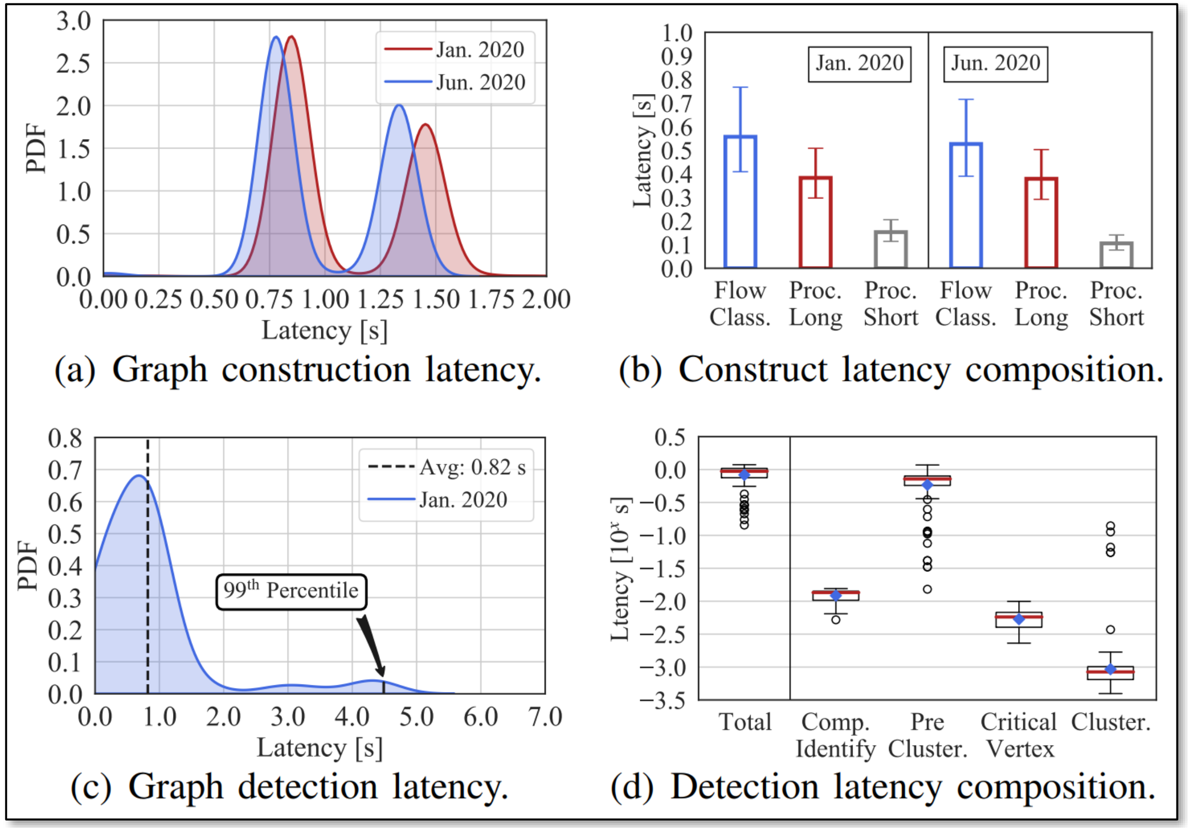


(a) Graph construction latency. (b) Construct latency composition. (c) Graph detection latency. (d) Detection latency composition.

Experimental Evaluation

Performance Results

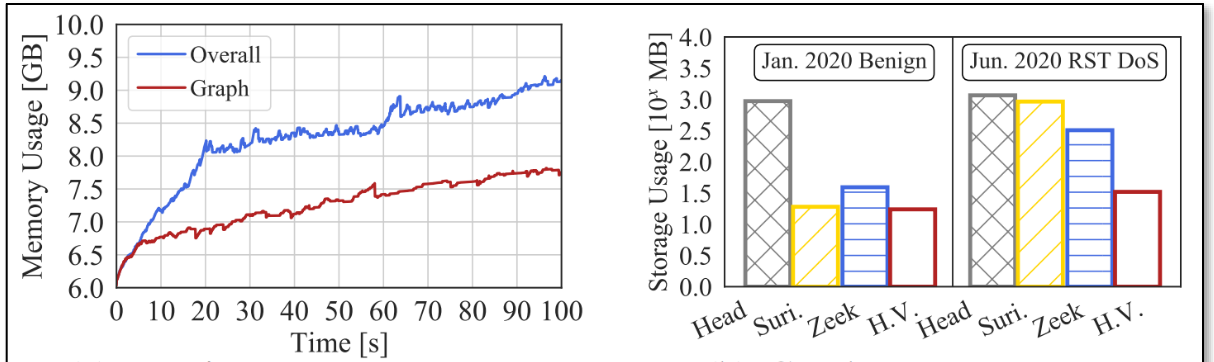
- **Latency**
  - Graph detection latency
    - 0.83s latency on average with a 99th percentile of 4.48s
  - Detection latency composition
    - 75.8% of the latency comes from pre-clustering
      - Pre-clustering step reduces the processing overhead of the subsequent processing



Experimental Evaluation

Performance Results

- Resource Consumption
  - The increasing rate of memory for maintaining the graph is only 13.1 MB/s
  - HyperVision utilizes 1.78 GB memory to maintain the flow interaction patterns extracted from 2.82 TB ongoing traffic
  - Graph storage usages
    - HyperVision achieves 8.99%, 55.7%, 98.1% storage reduction over the baselines



(a) Runtime memory usages. (b) Graph storage usages.

Raw packet header

Suricata

Conclusion

- *HyperVision* is an ML based real time detection system for encrypted malicious traffic with unknown patterns
- *HyperVision* uses two different strategies to represent the interaction patterns generated by short and long flows and aggregates the information of these flows
- *HyperVision* is unsupervised graph learning method to detect the traffic by utilizing the connectivity, sparsity, and statistical features in the graph

Thank you

Appendix

Features of Edges Used in HyperVision

Edge	Group	Data	Description
Edge Denoting Short Flows	structural	bool	Denoting short flows with the same source address.
		bool	Denoting short flows with the same source port.
		bool	Denoting short flows with the same destination address.
		bool	Denoting show flows with the same destination port.
		int	The in-degree of the connected source vertex.
		int	The out-degree of the connected source vertex.
		int	The in-degree of the connected destination vertex.
		int	The out-degree of the connected destination vertex.
	statistical	int	The number of flows denoted by the edge.
		int	The length of the feature sequence associated with the edge.
		int	The sum of packet lengths in the feature sequence.
		int	The mask of protocols in the feature sequence.
		float	The mean of arrival intervals in the feature sequence.
Edge Denoting Long Flows	structural	int	The in-degree of the connected source vertex.
		int	The out-degree of the connected source vertex.
		int	The in-degree of the connected destination vertex.
		int	The out-degree of the connected destination vertex.
	statistical	float	The flow completion time of the denoted long flow.
		float	The packet rate of the denoted long flow.
		int	The number of packets in the denoted long flow.
		int	The maximum bin size for fitting packet length distribution.
		int	The length associated with the maximum bin size.
		int	The maximum bin size for fitting protocol distribution.
		int	The protocol associated with the maximum bin size.

Hyper-Paramter Configuration

Group	Hyper-Parameter	Description	Value
Graph Construction	PKT_TIMEOUT	Flow completion time threshold.	10.0s
	FLOW_LINE	Flow classification threshold.	15
	AGG_LINE	Flow aggregation threshold.	20
Graph Pre-Processing	ϵ	DBSCAN hyper-parameters for clustering components and edges.	4×10^{-3}
	minPoint		40
Traffic Detection	K	K-means hyper-parameter.	10
	T	Loss threshold for malicious traffic.	10.0
	α	Balancing the terms in the loss function.	0.1
	β		0.5
	γ		1.7

Details of Malicious Traffic Datasets

Class		Dataset Label	Description	Att.[1]	Vic.	B.W.[2]	Enc. Ratio
Malware Related Encrypted Traffic	Spyware	Magic.	Magic Hound spyware.	2	479	0.34	0.13%
		Trickster	Encrypted C&C connections.	2	793	0.63	10.0%
		Plankton	Pulling components from CDN.	3	579	59.2	23.8%
		Penetho	Wifi cracking APK spyware.	1	516	3.57	100%
		Zsone	Multi-round encrypted uploads.	1	479	5.98	93.0%
		CCleaner	Unwanted software downloads.	4	466	28.1	4.09%
	Adware	Feiwo	Encrypted ad API calls.	3	1.00K	19.8	100%
		Mobidash	Periodical statistic ad updates.	3	624	6.08	100%
		WebComp.	WebCompanion click tricker.	3	281	8.38	55.2%
		Adload	Static resources for PPI adware.	1	280	1.04	1.09%
	Ransom-ware	Svpeng	Periodical C&C interactions (10s).	2	403	1.21	1.26%
		Koler	Invalid TLS connections.	3	333	2.22	100%
		Ransombo	Executable malware downloads.	5	369	58.6	42.7%
		WannaL.	Wannalocker delivers components.	2	275	7.49	30.3%
		Dridex	Victim locations uploading.	1	429	4.10	100%
	Miner	BitCoinM.	Abnormal encrypted channels.	1	1.54K	0.79	100%
		TrojanM.	Long SSL connections to C&C.	3	1.37K	2.39	89.4%
		CoinM.	Periodical connections to pool.	1	1.40K	0.21	100%
	Botware	THBot	Getting C&C server addresses.	4	103	1.72	2.71%
		Emotet	Communication to C&C servers.	6	1.17K	1.43	68.6%
		Snojan	PPI malware downloading.	3	326	8.94	100%
		Trickbot	Connecting to alternative C&C.	4	347	0.57	100%
		Mazarbot	Long C&C connections to cloud.	3	409	6.13	30.9%
		Sality	A P2P botware.	20	247	2.19	100%
Encrypted Flooding Traffic	Link Flooding	CrossfireS.	We use the botnet cluster sizes and the ratio of decoy servers (HTTPS) in [41].	100	313	197	100%
		CrossfireM.		200	313	278	100%
		CrossfireL.		500	313	503	100%
		LrDoS 0.2	We use the traffic of an encrypted video application and the settings in WAN experiments [44]	1	1	5.57	100%
		LrDoS 0.5		1	1	3.25	100%
		LrDoS 1.0		1	1	1.90	100%
	SSH Inject	ACK Inj.	SSH injection via ACK rate-limits.	1	2	1.78	-
		IPID Inj.	SSH injection via IPID counters.	1	2	0.28	-
		IPID Port	Requires of the SSH injection.	1	1	1.83	-
	Password Cracking	Telnet S.	Telnet servers in AS38635.	1	19	0.63	100%
		Telnet M.	Telnet servers in AS2501.	1	43	1.70	100%
		Telnet L.	Telnet servers in AS2500.	1	83	2.76	100%
		SSH S.	SSH servers in AS9376.	1	35	1.39	100%
		SSH M.	SSH servers in AS2500.	1	257	2.49	100%
		SSH L.	SSH servers in AS2501.	1	486	5.53	100%
Encrypted Web Traffic	Web Attacks	Oracle	TLS padding Oracle.	1	1	3.99	100%
		XSS	Xsssniper XSS detection.	1	1	31.8	100%
		SSLScan	SSL vulnerabilities detection.	1	1	15.0	100%
		Param.Inj.	Commix parameter injection.	1	1	17.1	100%
		Cookie.Inj.	Commix cookie injection.	1	1	39.6	100%
		Agent.Inj.	Commix agent-based injection.	1	1	19.7	100%
		WebCVE	Exploiting CVE-2013-2028.	1	1	2.30	100%
		WebShell	Exploiting CVE-2014-6271.	1	1	11.2	100%
		CSRF	Bolt CSRF detection.	1	1	7.73	100%
		Crawl	A crawler using scrapy.	1	1	29.7	100%
	SMTP SSL	Spam1	Spam using SMTP-over-SSL.	1	1	36.2	100%
		Spam50	Encrypted spam with 50 bots.	50	1	61.7	100%
		Spam100	Brute spam using 100 bots.	100	1	88.9	100%
Traditional Brute Force Attack	Brute Scanning	ICMP	We use the brute force scanning rates identified by darknet in [22]. We reproduce the scan using Zmap which targets the peers and customers of AS 2500.	1	211K	5.61	-
		NTP		1	99.3K	3.87	-
		SSH		1	205K	5.79	-
		SQL		1	112K	3.04	-
		DNS		1	198K	6.61	-
		HTTP		1	93.7K	2.68	-
		HTTPS		1	209K	4.89	-
	Source Spoof	SYN	We use the protocol types and the packet rates in [40].	6.50K	1	11.41	-
		RST		32.5K	1	5.79	-
		UDP		6.50K	1	54.3	-
		ICMP		3.20K	1	0.13	-
	Amplification Attack	NTP	We use the packet rates and the vulnerable protocols observed in [40]. And we use the number of the reflectors in [43].	650	1	95.8	-
		DNS		200	1	82.7	-
		CharGen		200	1	175	-
		SSDP		1.30K	1	7.23	-
		RIPv1		500	1	7.04	-
		Memcache		1.60K	1	63.5	-
		CLDAP		1.30K	1	36.8	-
	Probing Vulnerable Application	Lr. SMTP	We use the sending rates of vulnerable application discovery disclosed by a darknet [22]. We estimate the number of scanners by the number of visible active addresses from the vantage (i.e., realword measurements) and the size of the darknet.	11	158K	7.97	-
		Lr.NetBios		28	444K	17.3	-
		Lr.Telnet		156	1.23M	49.0	-
		Lr.VLC		22	352K	20.5	-
		Lr.SNMP		6	110K	6.51	-
		Lr.RDP		172	1.30M	53.0	-
		Lr.HTTP		94	640K	38.0	-
		Lr.DNS		28	428K	25.0	-
		Lr.ICMP		268	1.82M	63.3	-
		Lr.SSH		72	994K	5.63	-

[1] Att. and Vic. indicate the number of attackers and victims.
[2] B.W. is short for total bandwidth in the unit of Mb/s.

Five Generic Malicious Traffic Detection Methods

- Jaqen
  - Sampling based recording and signature based detection
- FlowLens
  - Sampling based recording and ML based detection
  - Supervised learning
- Whisper
  - Flow-level features and ML based detection
- Kitsune
  - Packet-level features and DL based detection
  - Unsupervised learning
- Deeplog
  - Event based recording and DL based detection