APPENDIX

A. Details of Implementations

We present the details of the flow classification and short flow aggregation algorithm in Algorithm 1 and 2, respectively. The features used for edge pre-clustering and clustering are shown in Table V. And Table VI shows the hyper-parameters used in HyperVision and the recommended values.

TABLE V
THE FEATURES OF EDGES USED IN HYPERVISION.

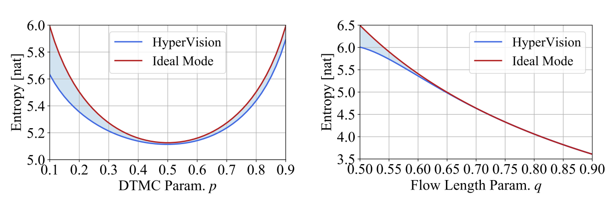
Edge	Group	Data	Description
Edge Denoting Short Flows	structural	bool	Denoting short flows with the same source address.
		bool	Denoting short flows with the same source port.
		bool	Denoting short flows with the same destination address.
		bool	Denoting show flows with the same destination port.
		int	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.

TABLE VI
RECOMMENDED HYPER-PARAMETER 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

TABLE VII
THE INTEGRAL OF THE DENSITY IN THE FEASIBLE REGION.

Per-Packet Features	Packet Length	Time Interval	Protocol Type
$\iint_{\mathcal{F}} \mathcal{D}_{\text{Ideal}}(p,q)\mathrm{d}p\mathrm{d}q$	1.011▼32.10%	0.918▼32.00%	0.795▼32.51%
$\iint_{\mathcal{F}} \mathcal{D}_{\text{Samp.}}(p,q)\mathrm{d}p\mathrm{d}q$	0.965▼35.17%	0.963▼28.66%	0.800▼32.08%
$\iint_{\mathcal{F}} \mathcal{D}_{\text{Eve.}}(p,q)\mathrm{d}p\mathrm{d}q$	0.588▼60.51%	0.588▼56.44%	0.588▼50.08%
$\iint_{\mathcal{F}} \mathcal{D}_{\text{H.V.}}(p,q)\mathrm{d}p\mathrm{d}q$	1.489▲47.27%	1.350▲35.51%	1.178▲48.18%



(a) Fix q and leave p as variable. (b) Fix p and leave q as variable.

Fig. 20. HyperVision approaches the idealized flow recording mode on information entropy.

TABLE VIII
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	100	313	197	100%
		CrossfireM.	and the ratio of decony servers	200	313	278	100%
		CrossfireL.	(HTTPS) in [10].	500	313	503	100%
		LrDoS 0.2	We use the traffic of an encrypted	1	1	5.57	100%
		LrDoS 0.5	video application and the settings	1	1	3.25	100%
		LrDoS 1.0	in WAN experiments [58]	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 [11]. 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 [32].	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 [32]. And we use the number of the reflectors in [55].	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 [11]. 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.

Algorithm 1: Secure flow classification.

```
Input: Per-packet features: PktInfo, the hash table for flow collecting:
       FlowHashTable.
Output: Classified flows: ShortFlow and LongFlow.
time_now := PktInfo[0].time, last_check := time_now.
for pkt in PktInfo do
    // Aggregate packets into flows.
    if Hash(pkt) not in FlowHashTable then
        FlowHashTable adds an entry for pkt.
    FlowHashTable[Hash(pkt)] appends pkt.
    if time_now − last_check > JUDGE_INTERVAL then
        for flow in FlowHashTable do
            // Judge the completion of flows.
            if time_now − flow[−1].time > PKT_TIMEOUT then
                // Classify the flow via the number of packets.
                if flow.size < FLOW_LINE then
                    ShortFlow adds flow.
                else
                    LongFlow adds flow.
                FlowHashTable clears the states of flow.
        last_check ← time_now. // Record the time of checking.
    time_now ← pkt.time. // Update the timer.
```

Algorithm 2: Short flow aggregation.

```
Input: Short flows: ShortFlow.
Output: Constructed edges: ShortEdge.
Initialize ProtoHashTable as an empty table.
// Select candidate protocols for the aggregation.
for flow in ShortFlow do
    // Calculate the protocol mask of a short flow.
    flow_proto := (flow[0].proto|...|...|flow[−1].proto).
    if Hash(flow_proto) not in ProtoHashTable then
        ProtoHashTable adds an entry for flow_proto.
    Append flow to ProtoHashTable[Hash(flow_proto)].
// Perform the source aggregation.
for flows in ProtoHashTable with same protocols do
    SrcAddrTable collects the flows with same sources in flows.
    for sflow in SrcAddrTable do
        // The flows can be aggregated and denoted by one edge.
        if sflow.size > AGG_LINE then
            edge.features := sflow[0].features.
            edge.source := sflow[0].source.
            if an unique destination in sflow then
                // Source and destination aggregation.
                edge.destination saves the unique destination.
            else
                // Source aggregation only.
                Record each destination in sflow.
            Add the constructed edge to ShortEdge.
            SrcAddrTable evicts sflow.
    DstAddrTable collects flows with same destinations.
    Inspect the flows with the same destinations similarly.
    // Process short flows which cannot be aggregated.
    ShortEdge adds flows in SrcAddrTable and DstAddrTable.
```

B. Details of Experiments

1) Details of Datasets: We present the detailed properties of the 80 newly collected datasets in Table VIII, including the number of attackers and victims, the packet rates of attack flows, and the ratios of encrypted traffic. All the datasets are collected and labeled using the same method as MAWI datasets [41] and CIC datasets [72], [73]. Moreover, Table IX shows the performances on existing datasets.

C. Details of Theoretical Analysis

1) Analysis of Event based Mode: Let random variable $\mathrm{I}_{\mathrm{Eve.}}$ indicate if the event based mode records an event for a flow denoted by a random variable sequence, $\langle s_1, s_2, \dots, s_L \rangle$, $L \sim G(q)$. And we assume that the mode can merge repetitive events. First, we obtain the probability distribution of the random variable $\mathrm{I}_{\mathrm{Eve.}}$:

$$\begin{aligned}\mathbb{P}[\mathrm{I}_{\mathrm{Eve.}}=1] &= 1-\mathbb{P}[\mathrm{I}_{\mathrm{Eve.}}=0],\\ \mathbb{P}[\mathrm{I}_{\mathrm{Eve.}}=0] &= \sum_{l=1}^{\infty}\mathbb{P}[L=l]\cdot\mathbb{P}[\mathrm{I}_{\mathrm{Eve.}}=0|L=l]\\ &= \sum_{l=1}^{\infty}(1-q)^{l-1}\cdot q\cdot(1-p^s)^l\\ &= \frac{q(1-p^s)}{1-(1-q)(1-p^s)}.\end{aligned} \tag{21}$$

Then, we obtain the entropy of the random variable $\mathrm{I}_{\mathrm{Eve.}}$:

$$\begin{aligned}\mathcal{H}_{\mathrm{Eve.}} = \mathcal{H}[\mathrm{I}_{\mathrm{Eve.}}] =\\ -\mathbb{P}[\mathrm{I}_{\mathrm{Eve.}}=0]\ln\mathbb{P}[\mathrm{I}_{\mathrm{Eve.}}=0]-\mathbb{P}[\mathrm{I}_{\mathrm{Eve.}}=1]\ln\mathbb{P}[\mathrm{I}_{\mathrm{Eve.}}=1].\end{aligned} \tag{22}$$

We observe that $\frac{\partial\mathcal{H}[\mathrm{I}_{\mathrm{Eve.}}]}{\partial q}\approx 0$ when $q>0.5$. Thus, we use the second-order taylor series of q to approach $\mathcal{H}_{\mathrm{Eve.}}$:

$$\mathcal{H}_{\mathrm{Eve.}} = \frac{2q(1-p^s)\ln[\frac{(p^s-1)q}{p^s(q-1)-q}]}{p^s(q-1)-q} = -2\theta\ln\theta, \tag{23}$$

where $\theta=\frac{\zeta}{\eta}$, $\zeta=q-qp^s$, and $\eta=q-p^s(q-1)$. Similarly, we obtain the expected data scale $\mathcal{L}_{\mathrm{Eve.}}$ and the information density $\mathcal{D}_{\mathrm{Eve.}}$:

$$\begin{aligned}\mathcal{L}_{\mathrm{Eve.}} &= \mathbb{P}[\mathrm{I}_{\mathrm{Eve.}}=1] = \frac{p^s}{p^s(1-q)+q} = -\frac{p^s}{\eta},\\ \mathcal{D}_{\mathrm{Eve.}} &= \frac{\mathcal{H}_{\mathrm{Eve.}}}{\mathcal{L}_{\mathrm{Eve.}}} = \frac{2\zeta}{p^s}\cdot\ln\theta.\end{aligned} \tag{24}$$

Here, we complete the analysis for the event based mode.

2) Analysis of Sampling based Mode: We use $X_{\mathrm{Samp.}}$ to denote the random variable to be recorded as the flow information in the sampling based mode which is the sum of the observed per-packet features denoted by the random variable sequence. We can obtain the distribution of $X_{\mathrm{Samp.}}$ as follows:

$$X_{\mathrm{Samp.}} = \sum_{i=1}^{L} s_i, \quad s_i\sim B(s,p)\Rightarrow X_{\mathrm{Samp.}}\sim B(Ls,p). \tag{25}$$

The amount of the information recorded by the sampling based mode is the Shannon entropy of $X_{\mathrm{Samp.}}$. We decompose the entropy as conditional entropy and mutual information:

$$\begin{aligned}\mathcal{H}_{\mathrm{Samp.}} &= \mathcal{H}[X_{\mathrm{Samp.}}]\\ &= \mathcal{H}[X_{\mathrm{Samp.}}|L]+\mathcal{I}(X_{\mathrm{Samp.}};L).\end{aligned} \tag{26}$$

We assume that the mutual information between the sequence length L and the accumulative statistic $X_{\mathrm{Samp.}}$ is close to zero. It implies the impossibility of inferring the statistic from the length of the packet sequence. Then we obtain a lower bound of the entropy as an estimation which is verified to be a tight bound via numerical analysis:

$$\begin{aligned}&\begin{cases}\mathcal{H}_{\mathrm{Samp.}}=\mathcal{H}[X_{\mathrm{Samp.}}|L] & =\sum_{l=1}^{\infty}\mathbb{P}[L=l]\cdot\mathcal{H}[X_{\mathrm{Samp.}}|L=l]\\ \mathcal{H}[X_{\mathrm{Samp.}}|L=l] & =\frac{1}{2}\ln 2\pi elsp(1-p),\end{cases}\\ &\Rightarrow \mathcal{H}_{\mathrm{Samp.}} = \frac{1}{2}\ln 2\pi esp(1-p)+\frac{q}{2}\sum_{l=1}^{\infty}(1-q)^{l-1}\ln l.\end{aligned} \tag{27}$$

TABLE IX
DETECTION ACCURACY OF HYPERVISION AND THE BASELINES ON THE EXISTING DATASETS.

Method	Metric	Kitsune Datastes							CIC-IDS2017					CIC-DDoS2019		
		Mirai	Fuzz.	OS Scan	SSL DoS	SYN DoS	SSDP F.	Average	Tue.	Wed.	Thu.	Fri.	Average	Day1	Day2	Average
Jaqen	AUC	0.7452	0.9999	0.9998	0.9997	0.9965	0.9145	0.9426	/ [2]	/	/	/	/	0.9988	0.9986	0.9987
	F1	0.5170	0.9999	0.9998	0.9762	0.9951	0.9406	0.9048	/	/	/	/	/	0.9508	0.9620	0.9564
FlowLens	AUC	0.7818	0.9257	0.9809	0.9582	0.9999	0.9655	0.9353	0.9547	0.8876	0.8117	0.9484	0.9006	0.9909	0.8869	0.9389
	F1	0.3714	0.9543	0.8225	0.9295	0.8600	0.9706	0.8180	0.9193	0.8822	0.8148	0.8713	0.8719	0.8974	0.9337	0.9155
Whisper	AUC	0.9992	0.8294	0.9896	0.9998	0.9328	0.9887	0.9566	0.8101	0.7343	0.7677	0.7311	0.7608	-	-	-
	F1	0.8490	0.9531	0.9258	0.9778	0.8470	0.9792	0.9220	0.5077	0.6434	0.4915	0.5770	0.5549	-	-	-
Kitsune	AUC	0.9885	0.9986	0.9998	0.9275	0.9886	0.9946	0.9829	0.6891	0.4841	0.8091	0.9069	0.7223	-	-	-
	F1	0.9364	0.9710	0.9978	0.6006	0.5015	0.9695	0.8295	0.4745	0.3402	0.3745	0.5347	0.4310	-	-	-
DeepLog	AUC	0.8935	0.9457	0.9814	0.8106	0.9560	0.9999	0.9312	- [1]	-	-	-	-	-	-	-
	F1	0.8183	0.9281	0.9405	0.8106	0.9509	0.9943	0.9071	-	-	-	-	-	-	-	-
H.V.	AUC	0.9901	0.9999	0.9996	0.9914	0.9931	0.9587	0.9888	0.9970	0.9896	0.9420	0.9984	0.9818	0.9999	0.9999	0.9999
	F1	0.9974	0.9999	0.9999	0.9978	0.9979	0.9721	0.9942	0.9737	0.9679	0.9010	0.9676	0.9526	0.9997	0.9931	0.9964

[1] We highlight the best accuracy in ● and the worst accuracy in ●. And we mark - when an unsupervised method lacks benign traffic for training.
[2] Backslash means that Jaqen is designed to detect only volumetric attacks.

We observed that the second-order taylor series can accurately approach the second term of the entropy:

$$\mathcal{H}_{\text{Samp.}} = \frac{1}{2}\ln 2\pi esp(1-p) + \frac{\ln 2}{2}q(1-q). \tag{28}$$

Finally, we obtain the expected data scale and the information density similar to the analysis for the event based mode and complete the analysis for the sampling based mode.

3) Analysis of Graph based Mode in HyperVision: HyperVision applies different recording strategies for short and long flows, i.e., when $L > K$ it extracts the histogram for long flow feature distribution fitting, and when $L \leq K$ it records detailed per-packet features and aggregates short flows. Let $\mathcal{X}_{\text{H.V.}}$ denote the random set of the recorded information. For short flows, all the random variables are collected in $\mathcal{X}_{\text{H.V.}}$. For long flows, $\mathcal{X}_{\text{H.V.}}$ collects s counters of the histogram for each state on the state diagram of the DTMC. First, we decompose the entropy of the graph based recording mode as the terms for short and long flows:

$$\begin{aligned}\mathcal{H}_{\text{H.V.}} &= \mathcal{H}[\mathcal{X}_{\text{H.V.}}|L] = \sum_{l=1}^{\infty}\mathbb{P}[L=l]\cdot\mathcal{H}[\mathcal{X}_{\text{H.V.}}|L=l] \\ &= \mathcal{H}[\mathcal{X}^{\text{S}}_{\text{H.V.}}|L] + \mathcal{H}[\mathcal{X}^{\text{L}}_{\text{H.V.}}|L]\end{aligned} \tag{29}$$

$$\begin{cases}\mathcal{H}[\mathcal{X}^{\text{S}}_{\text{H.V.}}|L] &= \sum\limits_{l=1}^{K}\mathbb{P}[L=l]\cdot\mathcal{H}[\mathcal{X}_{\text{H.V.}}|L=l] \\ \mathcal{H}[\mathcal{X}^{\text{L}}_{\text{H.V.}}|L] &= \sum\limits_{l=K+1}^{\infty}\mathbb{P}[L=l]\cdot\mathcal{H}[\mathcal{X}_{\text{H.V.}}|L=l].\end{cases}$$

Short Flow Information. HyperVision records detailed per-packet feature sequences for short flows which is the same as the brute recording in the idealized mode. Thus, the increasing rate of information equals the entropy rate of the DTMC:

$$\mathcal{H}[\mathcal{X}_{\text{H.V.}}|L=l] = l\cdot\mathcal{H}[\mathcal{G}], \tag{30}$$

$$\begin{aligned}\mathcal{H}[\mathcal{X}^{\text{S}}_{\text{H.V.}}|L] &= \sum_{l=1}^{K}\mathbb{P}[L=l]\cdot l\cdot\mathcal{H}[\mathcal{G}] \\ &= q\cdot\mathcal{H}[\mathcal{G}]\cdot\sum_{l=1}^{K}(1-q)^{l-1}\cdot l \\ &= \frac{1-(Kq+1)(1-q)^K}{q}\cdot\mathcal{H}[\mathcal{G}].\end{aligned} \tag{31}$$

Long Flow Information. When $L > K$, the random set collects the counters for distribution fitting. When the DTMC has s states, the histogram has s counters $\upsilon_1, \upsilon_2, \dots, \upsilon_s$, i.e., $\mathcal{X}_{\text{H.V.}} = \{\upsilon_1, \upsilon_2, \dots, \upsilon_s\}$. We assume that the counters are independent:

$$\upsilon_i = \sum_{j=1}^{L}\delta_j, \qquad \delta_j = \begin{cases}1, & \text{if } s_j \text{ is the } i^{\text{th}} \text{ state} \\ 0, & \text{else.}\end{cases} \tag{32}$$

We observe that $\langle\upsilon_1, \upsilon_2, \dots, \upsilon_s\rangle$ is a binomial process:

$$\begin{aligned}\upsilon_i &\sim B(L, \mathbb{P}[s_i = i]) \\ &\sim B(L, C_s^i p^i(1-p)^{s-i}).\end{aligned} \tag{33}$$

To obtain the closed-form solution, we use $\frac{(sp)^i e^{-sp}}{i!}$ as an estimation of $C_s^i p^i (1-p)^{s-i}$. Moreover, the length of the per-packet feature sequence of a long flow is relatively large which implies υ_i approaches a Poisson distribution:

$$\begin{aligned}\upsilon_i &\sim \pi(L\cdot\mathbb{P}[s_i = i]) \\ &\sim \pi(\lambda_i), \quad \lambda_i = \frac{(sp)^i e^{-sp}}{i!}.\end{aligned} \tag{34}$$

Basing on the distribution of the collected counters, we obtain the entropy of the random set:

$$\begin{cases}\mathcal{H}[\upsilon_i|L=l] &= \frac{1}{2}\ln 2\pi el\frac{(sp)^i e^{-sp}}{i!} \\ \mathcal{H}[\mathcal{X}^{\text{L}}_{\text{H.V.}}|L=l] &= \sum\limits_{i=1}^{s}\mathcal{H}[\upsilon_i|L=l],\end{cases} \tag{35}$$

$$\begin{aligned}\mathcal{H}[\mathcal{X}^{\text{L}}_{\text{H.V.}}|L] &= \sum_{l=K+1}^{\infty}\mathbb{P}[L=l]\cdot\mathcal{H}[\mathcal{X}^{\text{L}}_{\text{H.V.}}|L=l] \\ &= \sum_{l=K+1}^{\infty}q(1-q)^{l-1}\cdot\sum_{i=1}^{s}\frac{1}{2}\ln 2\pi el\frac{(sp)^i e^{-sp}}{i!} \\ &= \frac{(1-q)^K}{2}[s\ln 2\pi e + \frac{s(s+1)}{2}\ln sp \\ &\quad - sp^2 - \sum_{i=1}^{s}\ln i!] + \frac{qs}{2}[\sum_{l=K+1}^{\infty}(1-q)^{l-1}\ln l].\end{aligned}$$

The assumption of $q > 0.5$ implies K^{th} order taylor series can accurately approach the last term in (35). Moreover, we utilize the quadric term of s in the taylor series of $\sum_{i=1}^{s}\ln i!$ to approach the entropy of long flows (γ is Euler–Mascheroni constant):

$$\begin{aligned}\mathcal{H}[\mathcal{X}^{\text{L}}_{\text{H.V.}}|L] = \frac{1}{4}&s(1-q)^K[(1+s)\ln ps + \\ &2\ln 2\pi e + 2q\ln K - 2s(1+p+\gamma)].\end{aligned} \tag{36}$$

Finally, we take (31) and (36) in (29) and complete the analysis for the entropy of the graph based recording mode. Similarly, we obtain the expected data scale by analyzing the conditions of short and long flows separately:

$$\begin{aligned}\mathcal{L}_{\text{H.V.}} &= \mathrm{E}[\mathcal{L}^{\text{S}}_{\text{H.V.}}|L] + \mathrm{E}[\mathcal{L}^{\text{L}}_{\text{H.V.}}|L] \\ &= \sum_{l=1}^{K} \mathbb{P}[L=l]\cdot\frac{L}{C} + \sum_{l=K+1}^{\infty} s\cdot\mathbb{P}[L=l] \\ &= s(1-q)^K + \frac{1-(Kq+1)(1-q)^K}{Cq},\end{aligned} \tag{37}$$

where C is the average number of flows denoted by an edge associated with short flows. Also, we obtain the expected information density by its definition: $\mathcal{D}_{\text{H.V.}} = \mathcal{H}_{\text{H.V.}}/\mathcal{L}_{\text{H.V.}}$ and complete the analysis for the graph based recording mode used by HyperVision.