Multi-timeline Based Real-time Anomaly Detection in Network Traffic

Kriangkrai Limthong

July 21, 2015

Outline

Introduction

Multi-timeline Detection System

Evaluation and Results

Conclusion

List of changes

- Propose a detection system instead of a model
- Conduct additional 4 experiments
  - Comparison between different representations
  - Scalability over different volumes of background traffic
  - Detectability on different time of anomaly occurrence
  - Comparison between with and without a weighting technique
- Reorganize and revise thesis chapters
- Provide precise problem statement and key idea of our proposal

- Provide more specific definition of training and testing condition

Introduction

What is network traffic anomaly ?

A network traffic anomaly is an incident, packet, flow, behavior etc., which do not conform to the normal

Causes of network anomaly

- Accidents
  e.g. Outages, Misconfigurations, Flash crowds
- Threats
  e.g. Attacks, Viruses, Worms, Scanning, Spamming

Intrusion detection $\neq$ Anomaly detection

Intrusion detection focuses on **only** threats

Anomaly detection focuses on **both** threats and accidents

Examples of intrusion detection in network traffic

Technique used	References
Signature based	[1–4]
Parametric statistical modeling	[5–7]
Nonparametric statistical modeling	[8]
Clustering based	[9–12]
Spectral based	[13–16]

Problems of prior studies

- Cannot detect anomalies in real time (alert after anomaly occurrence in a minute or less)
- Easy to evade or manipulate detection system when attackers perceive the technique
- Detect only anomalies caused by threats, cannot detect anomalies caused by accidents

Problem statement

"how to detect network traffic anomalies caused by threats or accidents in real time and attackers hardly evade or manipulate the detection system"

Contribution

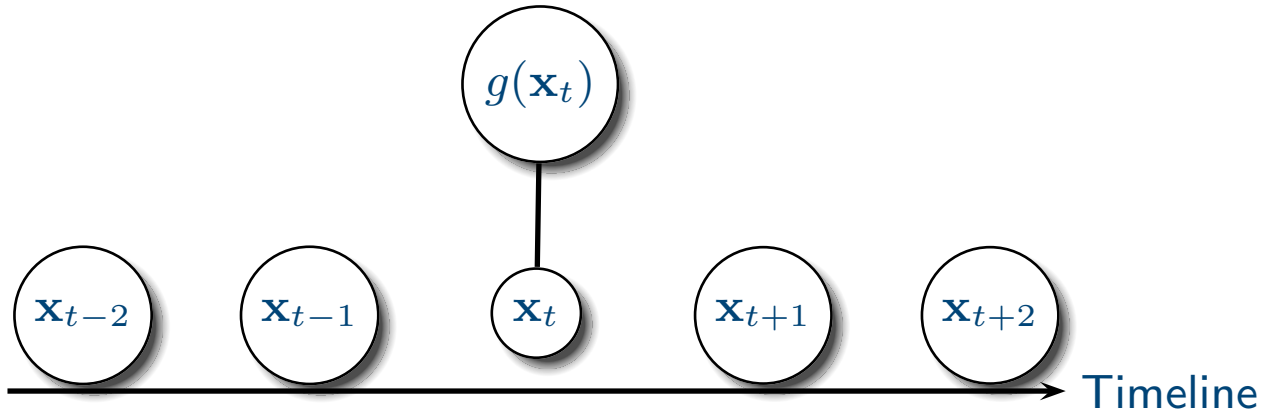
- We propose a system for real-time anomaly detection and related techniques
  - Introduce several input feature representations
  - Comparison of three learning algorithms
  - Comparison of nine traffic features and combined feature

Key ideas to solve the problems

- For scalability to achieve realtime, we used aggregated feature time series instead of per packet-based detection in the previous studies
- For robustness against forged data and detect accident, we use a new input data representation called multi-timeline
- For accuracy improvement, we introduced multi-timeline and weighting technique

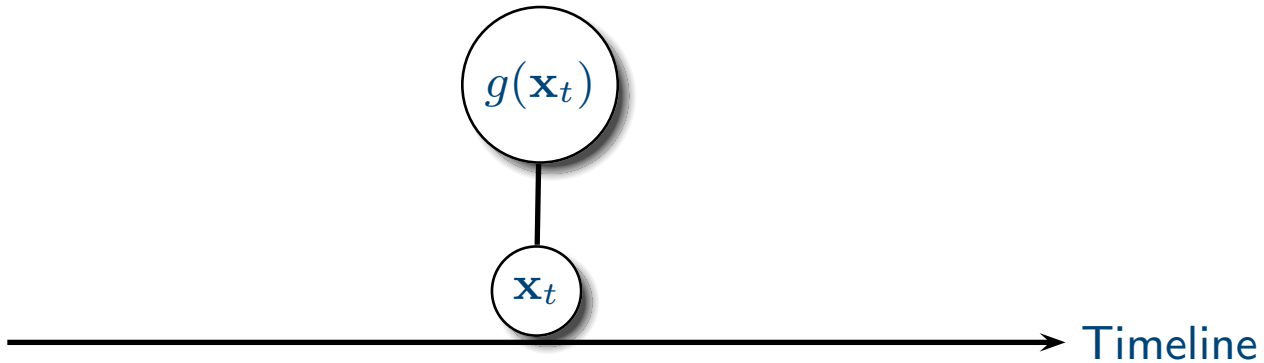
Representation of input data

- Manual-based representation
- Batch representation
- Real-time representation
- Multi-timeline representation (Proposal)
- Combination of realtime and multi-timeline representation (Proposal)



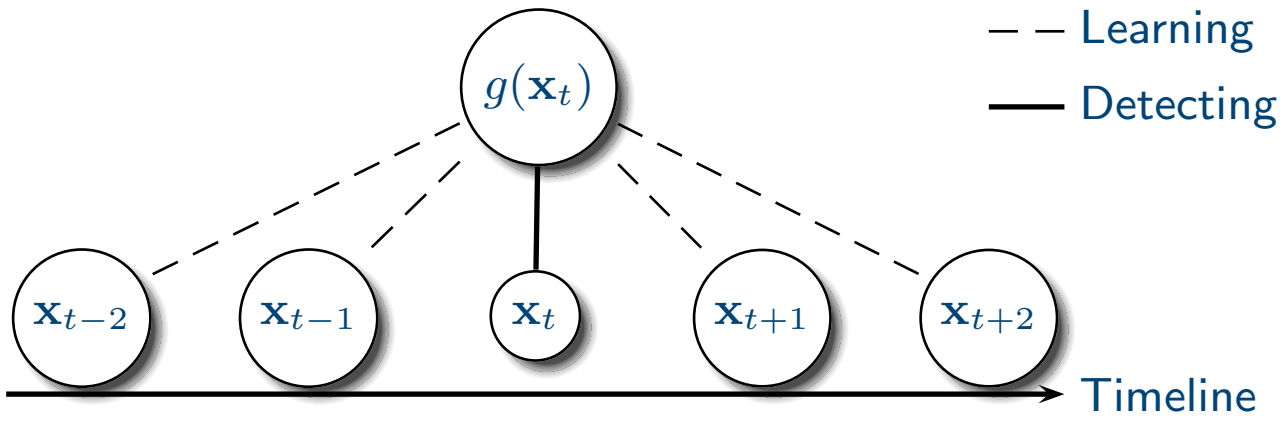
$\mathbf{x}_t$: a feature vector $\mathbf{x}$ occurs on the timeline at time t
$g(\mathbf{x}_t)$: a decision function of $\mathbf{x}_t$ (normal or anomaly)

Manual-based representation



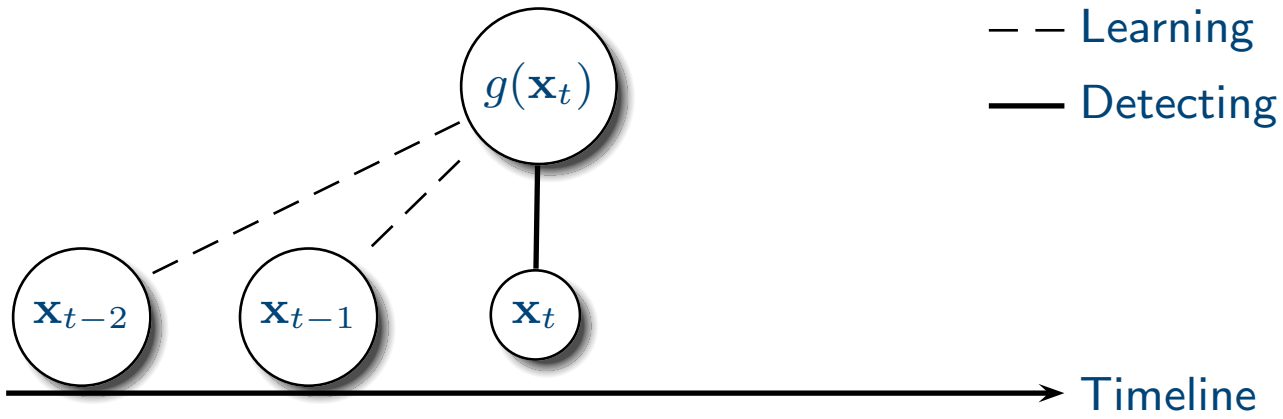
Decision function: $g(\mathbf{x}_t|\text{expertise information})$
Example: Snort [1], Bro [4], NetSTAT [17], RealSecure, and Cisco Secure IDS

Batch representation



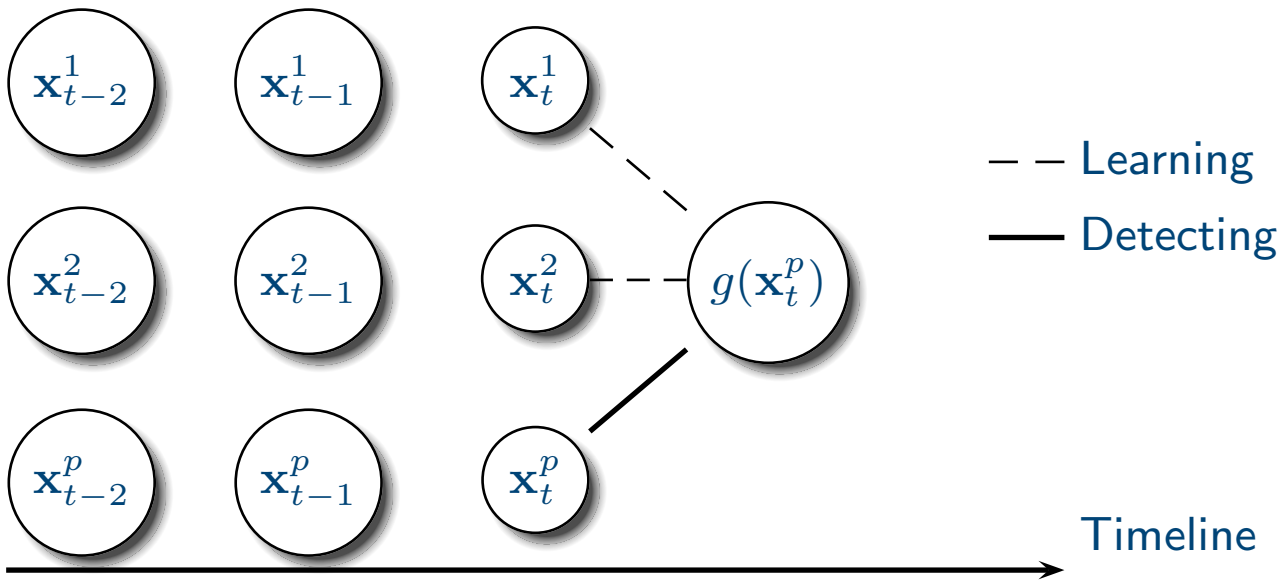
Decision function: $g(\mathbf{x}_t|\mathbf{x}_{t-2}, \mathbf{x}_{t-1}, \mathbf{x}_{t+1}, \mathbf{x}_{t+2})$
Example: S.Jiang et al. [18] and A.Kind et al. [19]

Real-time representation



Decision function: $g(\mathbf{x}_t|\mathbf{x}_{t-2}, \mathbf{x}_{t-1})$
Example: NIDES [20, 21] and RIDAN system [22]

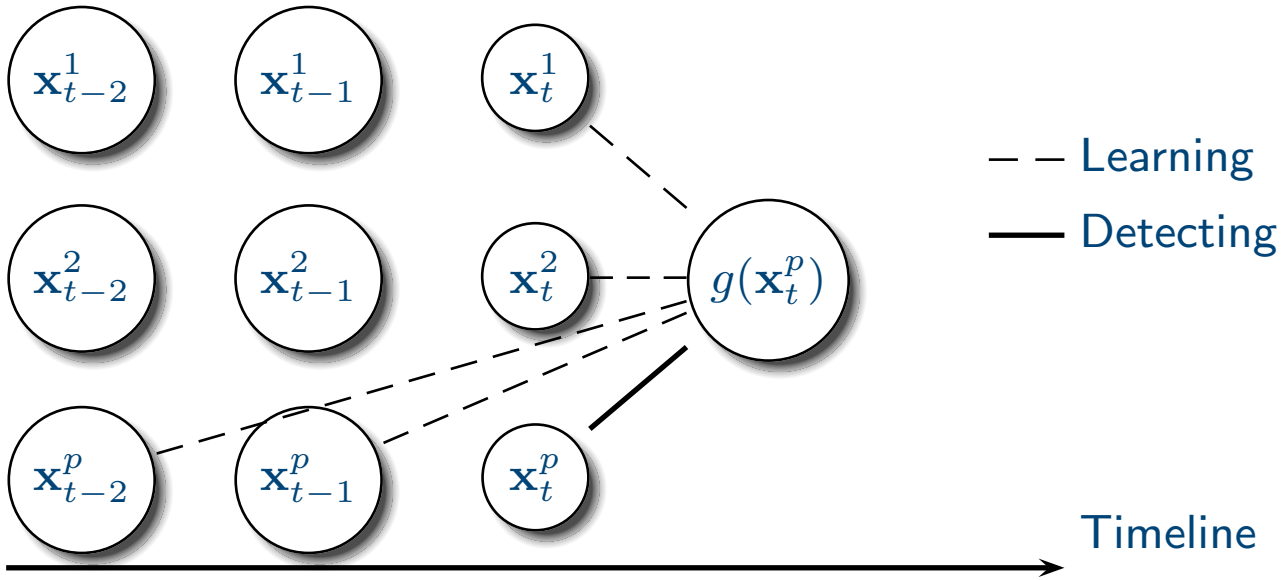
Multi-timeline representation (proposal)



Decision function: $g(\mathbf{x}^p_t|\mathbf{x}^1_t, \mathbf{x}^2_t, ..., \mathbf{x}^{p-1}_t)$

* In our experiments, we used the same time interval in past days (1day before, 2day before,...) *

Combination representation (proposal)

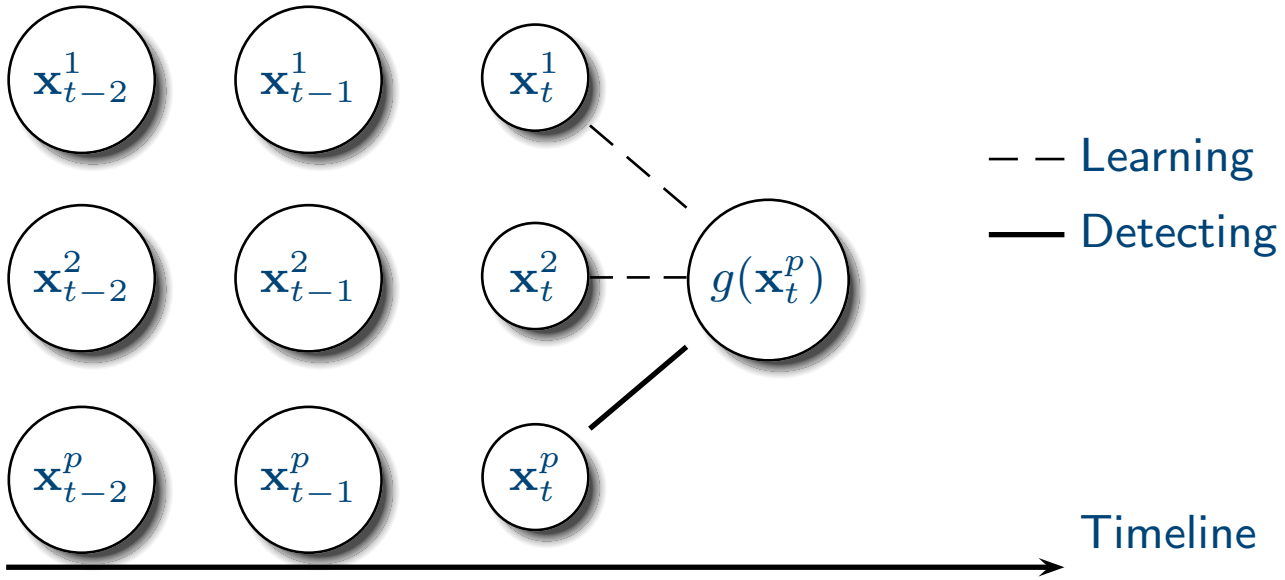


Decision function: $g(\mathbf{x}^p_t|\mathbf{x}^p_1, \mathbf{x}^p_2, ..., \mathbf{x}^p_{t-1}, \mathbf{x}^1_t, \mathbf{x}^2_t, ..., \mathbf{x}^{p-1}_t)$

* It combines real-time and multi-timeline representation *

Robustness in multi-timeline

- Periodicity of traffic is a key feature
- Attackers are difficult to manipulate historical data
- In case of accidents such as an outage, data from multiple timeline are more accurate than single timeline



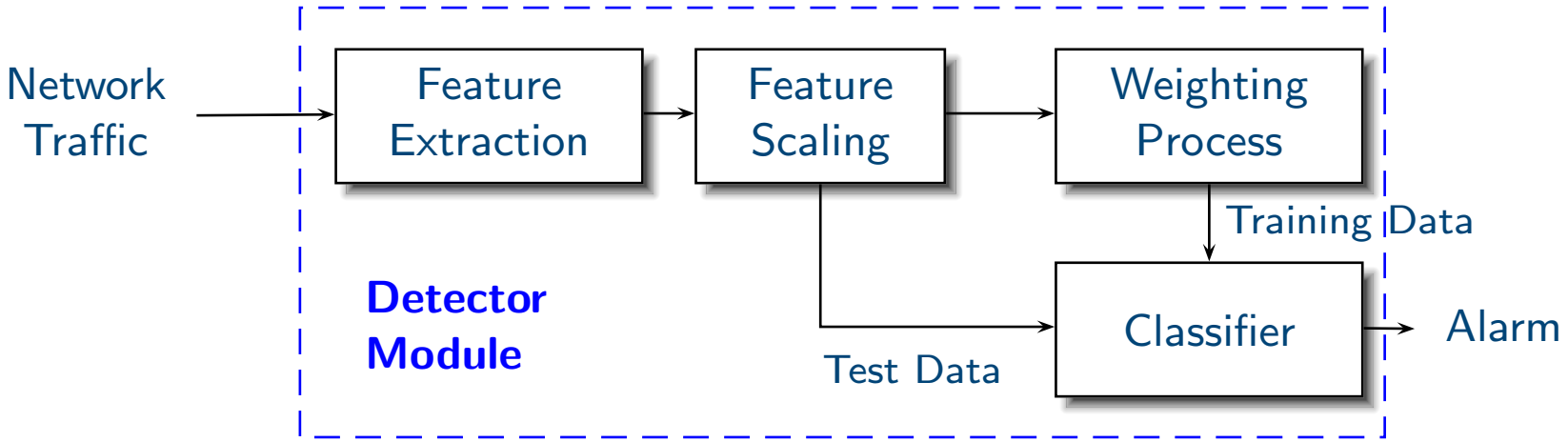
Comparison of representation of input data

Criteria	Manual	Batch	Real-time	Multi-timeline	Combination
Automation	No	Yes	Yes	Yes	Yes
Real-time detection	Yes	No	Yes	Yes	Yes
Flexibility	No	Yes	Yes	Yes	Yes
Robustness	No	No	No	Yes	Yes

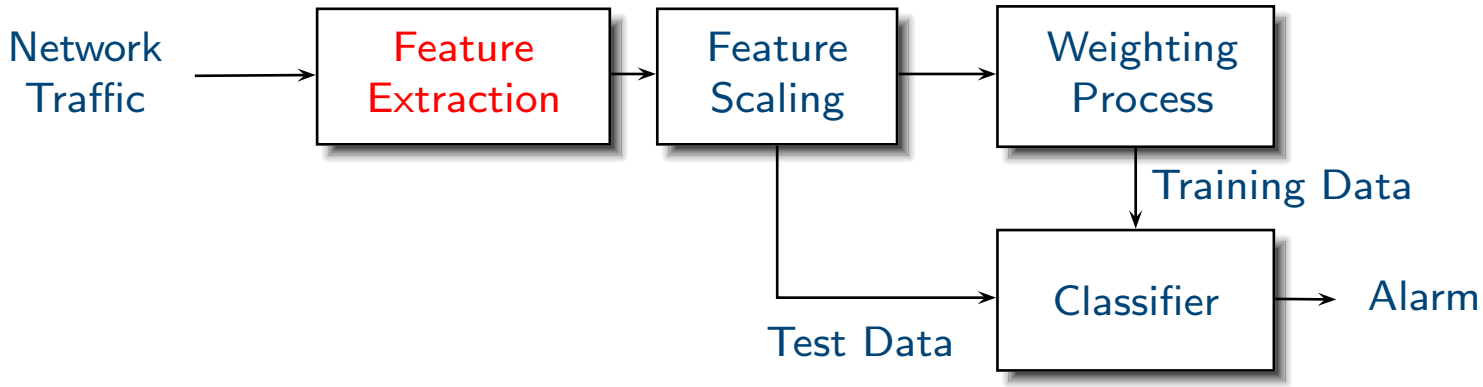
Automation: detect a new type of anomaly automatically
Real-time detection: alert in a minute or less
Flexibility: use different algorithms or features
Rubustness: hardly evade or manipulate the system

Multi-timeline Detection System

Detector module



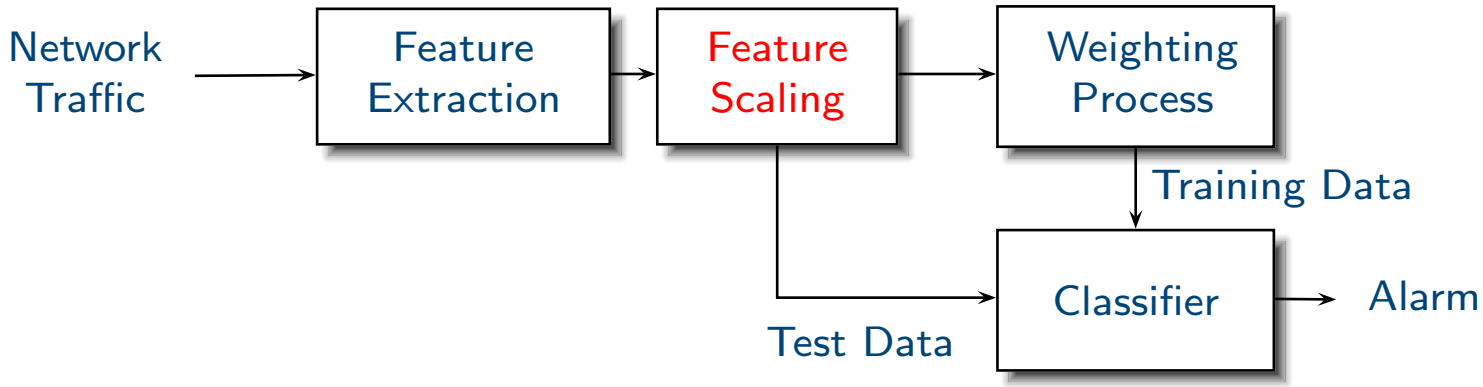
Feature extraction



$f\#$	Feature	Description
f_1	Packet	Number of packets
f_2	Byte	Sum of packet size
f_3	Flow	Number of flows
f_4	SrcAddr	Number of source addresses
f_5	DstAddr	Number of destination addresses
f_6	SrcPort	Number of source ports
f_7	DstPort	Number of destination ports
f_8	ΔAddr	|SrcAddr − DstAddr|
f_9	ΔPort	|SrcPort − DstPort|

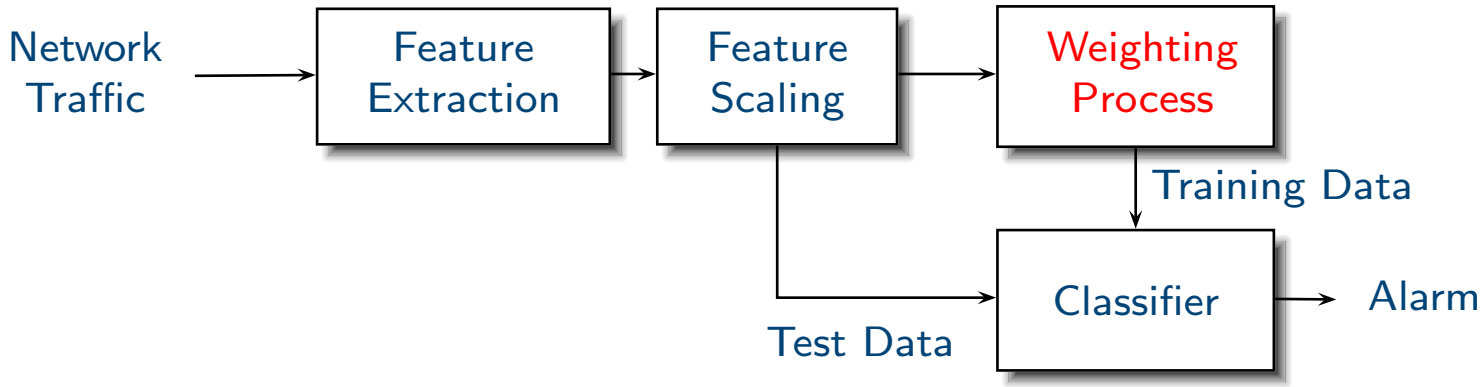
* We aggregate packet on an interval δ to extract these features *

Feature scaling



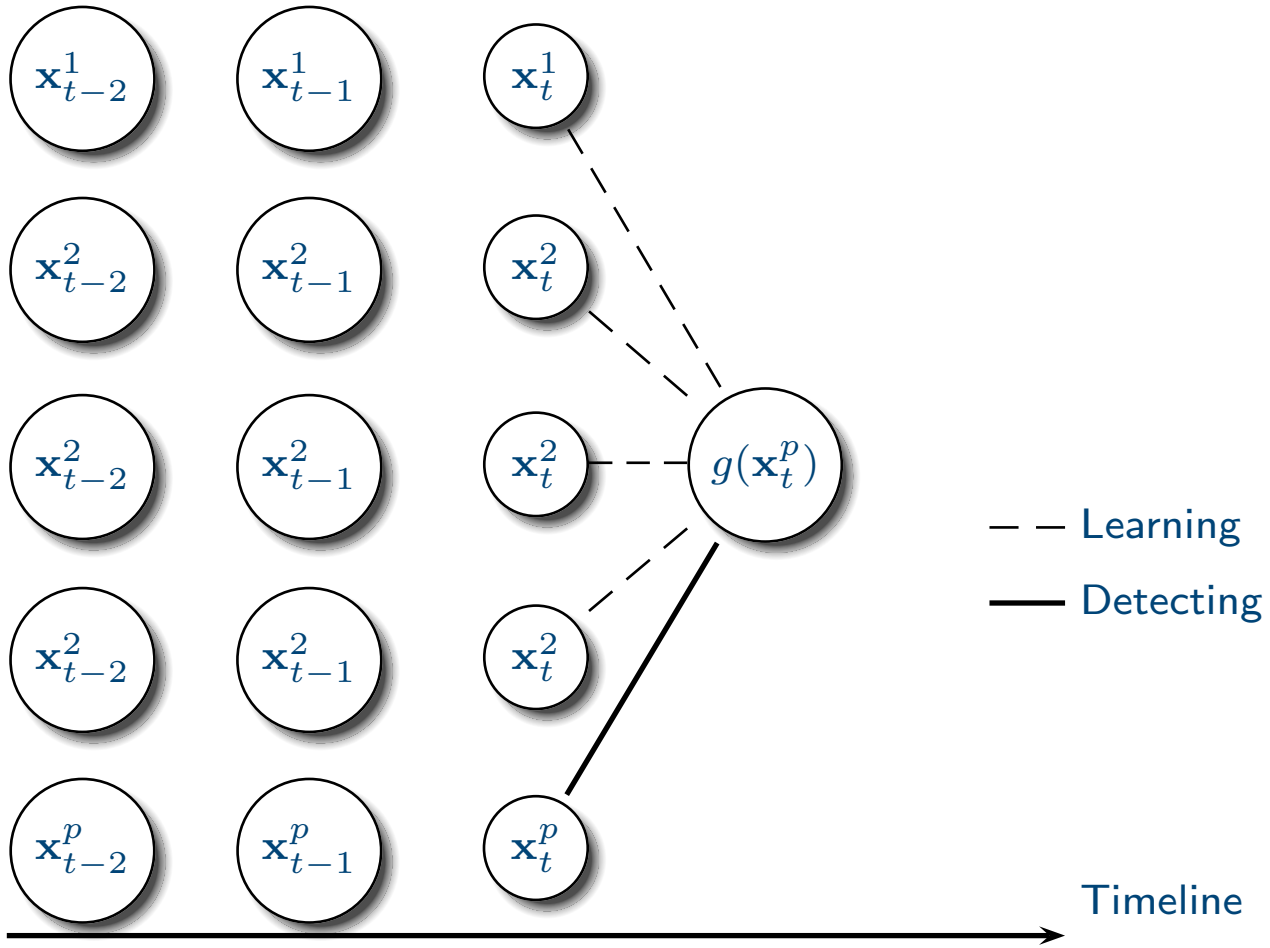
- Find the maximum value for each feature
- Normalize by dividing each feature value with the maximum value
- The range of feature value is between 0 and 1

Weighting process



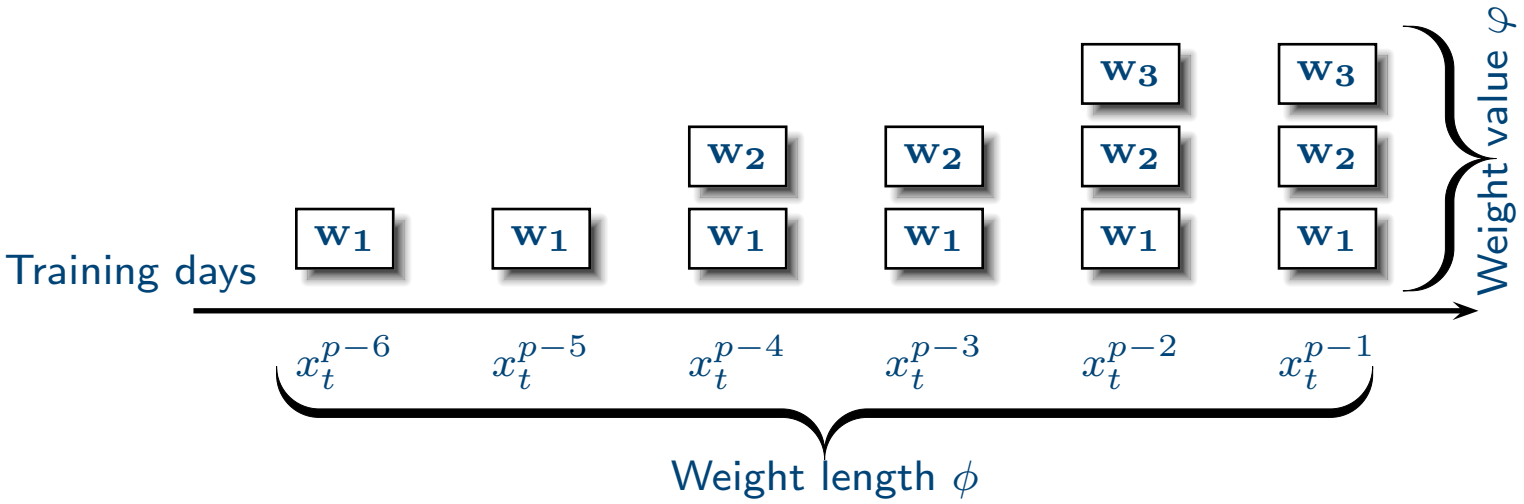
- Bias decision function on recent timelines
- Perform during learning process
- Use gradual weighting function [23]

Weighting process



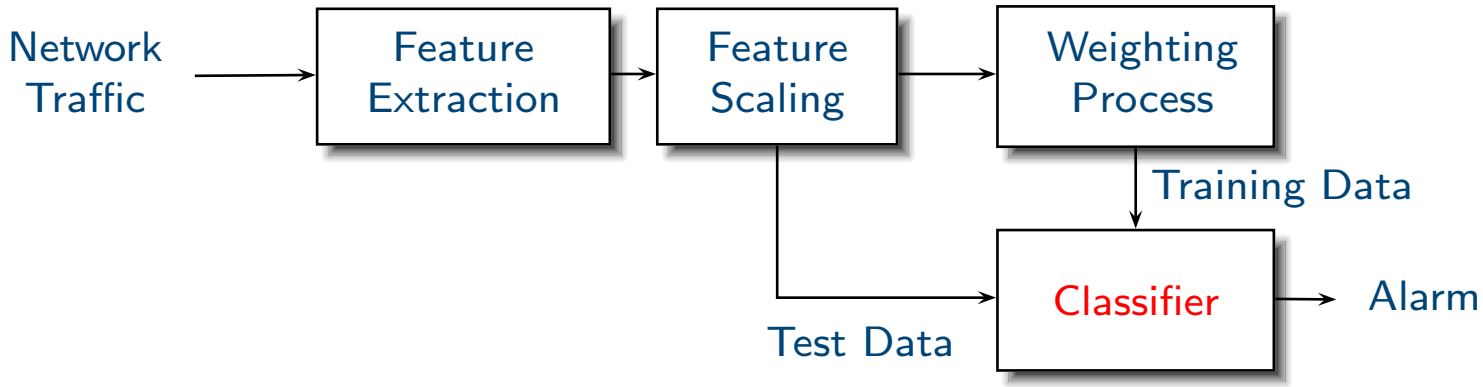
Decision function: $g(\mathbf{x}^p_t|\mathbf{x}^1_t, \mathbf{x}^2_t, \mathbf{x}^2_t, \mathbf{x}^2_t)$

Weight length and weight value



- We apply linear gradual weighting technique [23] with two parameters
- Weight length ϕ is the number of weighted training days
- Weight value φ is the maximum number of replication
- Decision function: $g(\mathbf{x}_t^p|\mathbf{x}_t^1,...,\mathbf{x}_t^{p-2},\mathbf{x}_t^{p-1},\mathbf{x}_t^{p-1},\mathbf{x}_t^{p-1})$

Classifier



- **Multivariate normal distribution (MND)** [24]
  Pros: parametric method, simplicity
  Cons: normal distribution, low accuracy
- **k-nearest neighbor (KNN)** [25]
  Pros: robust to noisy data, fast training
  Cons: biased by value of k, memory limitation
- **One-class support vector machine (OSVM)** [26]
  Pros: kernel-based method, flexibility
  Cons: slow training, sensitive to noisy data

Evaluation and Results

Data sets: Normal traffic

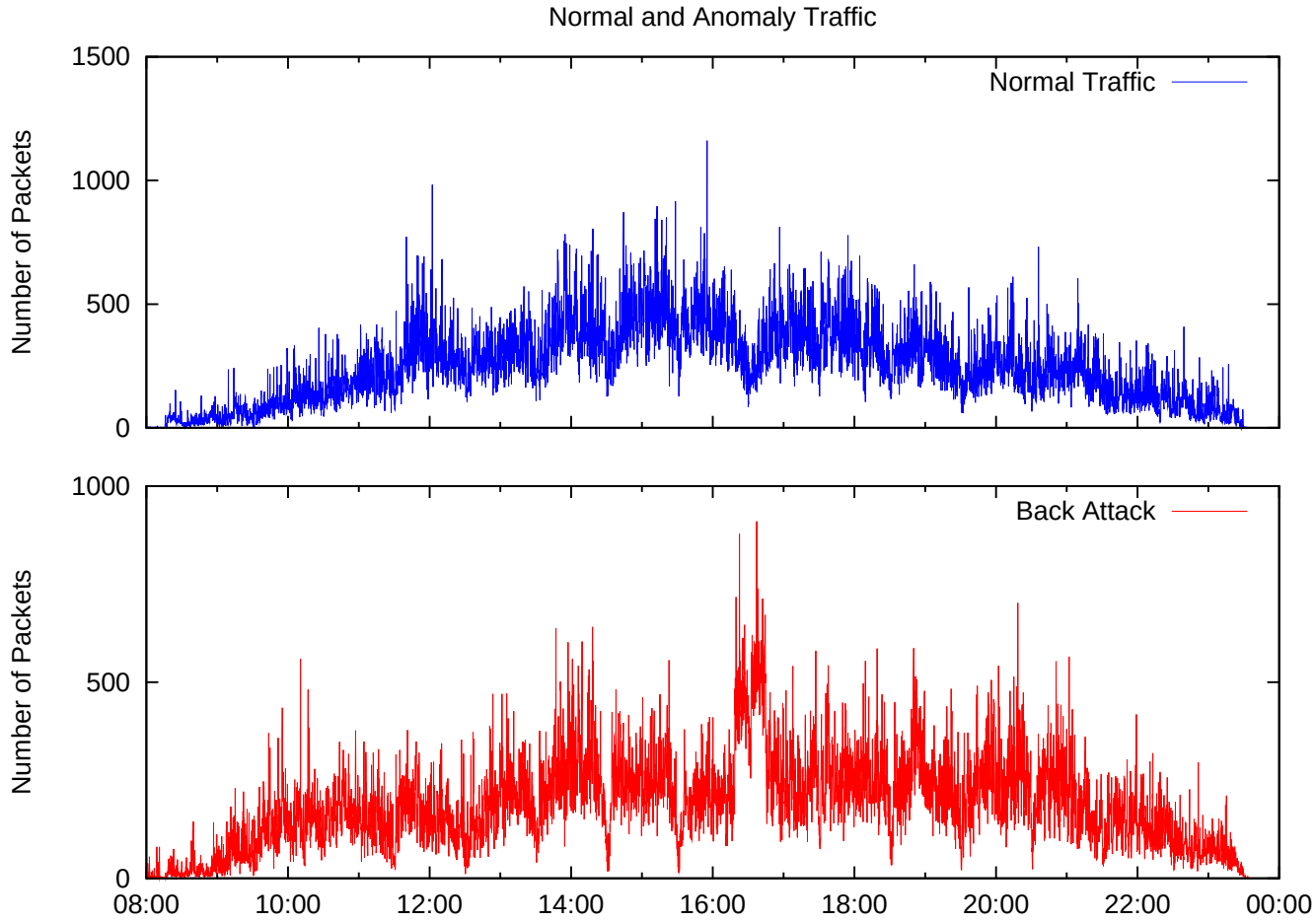
- Normal Traffic from a campus network [27] (26.19 packet/sec.)

- 175 clients and approximate 1,300 users every day
- Service hours are 8:00 - 24:00 on workdays and 8:00 - 18:00 on weekends
- We acquired traffic traces for 6 months and manually select for 55 days
- We divided traffic traces 39 days ($\approx 70\%$) for training data and 16 days ($\approx 30\%$) for test data

Data sets: Anomaly traffic

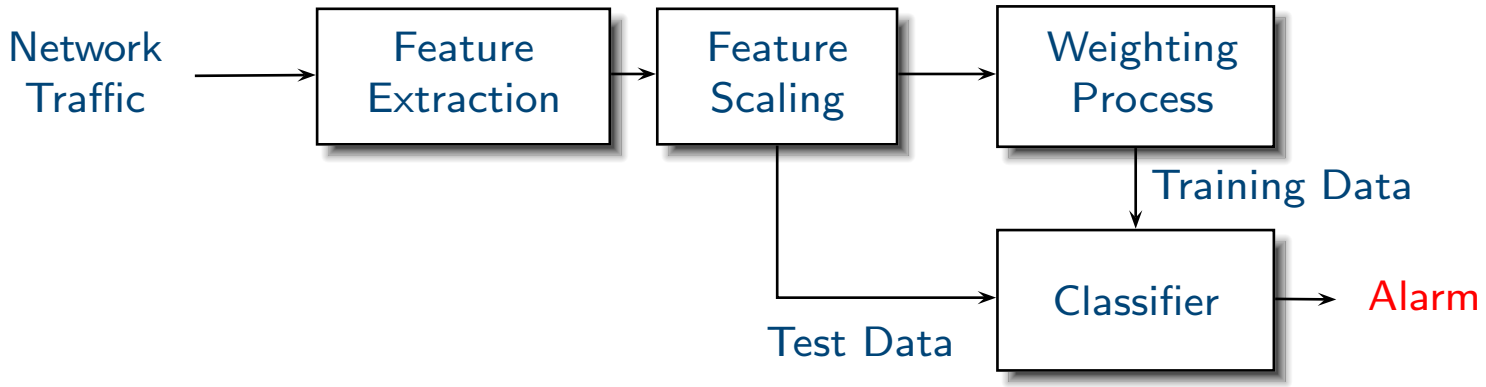
- Anomaly Traffic from MIT testbed data set [28–30]
  - Back - DoS attack through port 80 (54.04 packet/sec.)
  - IpSweep - Ping sweep on multiple ports/addresses (22 packet/sec.)
  - Neptune - SYN flood attack (67.92 packet/sec.)
  - PortSweep - Sweep on a single host with multiple ports (1.2 packet/sec.)
  - Smurf - ICMP echo reply flood attack (966.63 packet/sec.)

Example of normal and anomaly traffic



* Back attack occurred from 16:18 to 16:44 *

Performance evaluation



$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

$$F\text{-}score = 2 \times \frac{precision \times recall}{precision + recall}$$

F-score $= 0$ indicates 100% incorrect; 1 indicates 100% correct

Experiments

- Exp.1: Vary interval values
- Exp.2: Apply different features
- Exp.3: Comparison with other representations (new)
- Exp.4: Test for accidental anomaly
- Exp.5: Vary the number of training data
- Exp.6: Measure time consumption
- Exp.7: Vary background traffic (new)
- Exp.8: Alter anomaly occurrence (new)
- Exp.9: Effect of weighting (new)

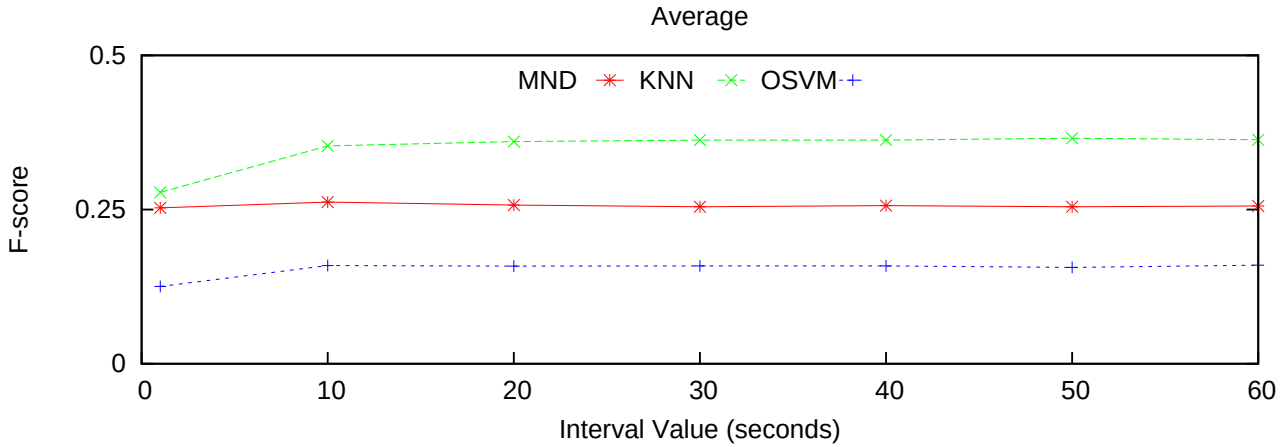
Exp.1: Vary interval values

Objective:

- Identify the practical value of interval
- Observe detection performance on different interval values

Procedure: Vary the interval value with 1, 10, 20, 30, 40, 50, and 60 seconds

Exp.1: Results and implications



- The best interval value for our data is 10 seconds
- Detection performance for 10-60 seconcds are very similar
- Interval value affects detection performance

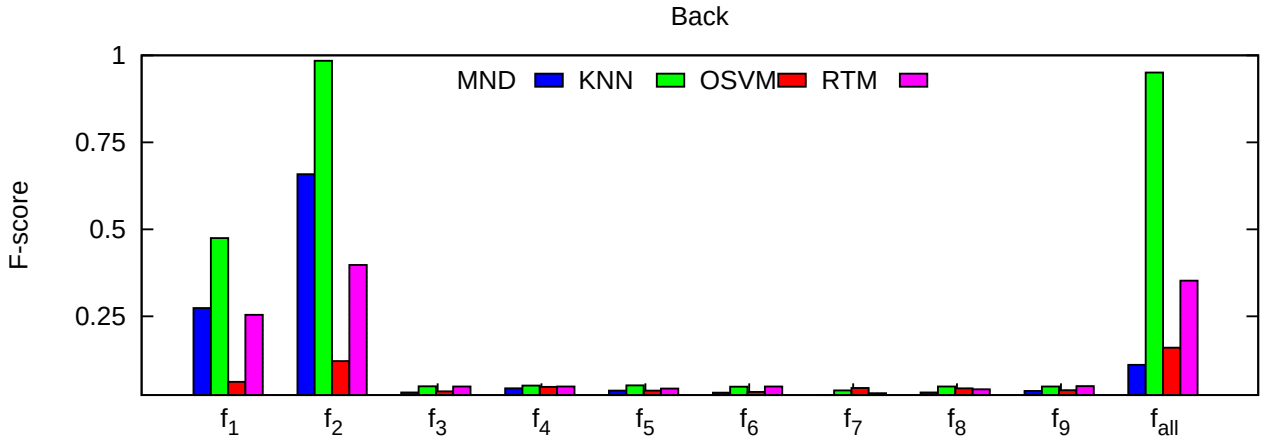
Exp.2: Apply different features

Objective:

- Investigate effective features for each type of anomaly
- Compare detection performance between single features and combined all features

Procedure: Apply different features to each type of anomaly

Exp.2: Example results and implications



F-score	MND	KNN	OSVM	Real-time
Average	0.57	0.69	0.25	0.39
Maximum	0.84	0.99	0.40	0.57

- Most detection performance of single features are better than those of combined all features
- Detection performances of multi-timeline are better than those of real-time representation

Exp.3: Comparison with other representations

Objective:

- Investigate other representations of input data
- Compare detection performance between multi-timeline and others representations

Procedure: Apply different other representations

t	Real-time	Combination
60	5.24-7.13%	1.08-2.11%
360	6.93-10.47%	4.97-6.34%
8640	15.71-23.53%	9.82-18.31%

* Table shows performance degradation over t which the number of training data on single timeline *

Implications: The multi-timeline representation outperforms both single and combination representation

Exp.4: Test for accidental anomaly

- Exp.4.1: No packet in test data
- Exp.4.2: No packet in training data

Exp.4.1: No packet in test data

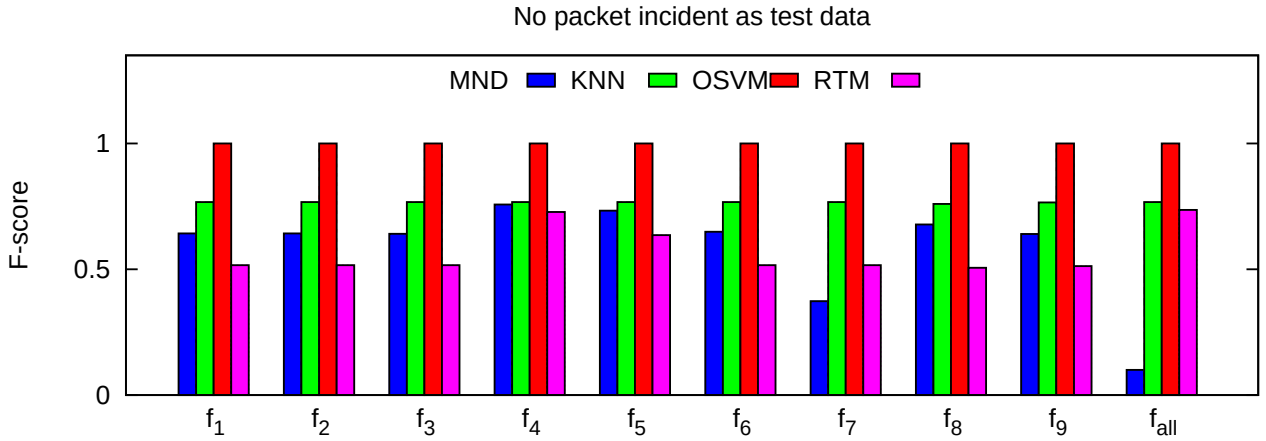
Objective:

- Examine multi-timeline detection system for accident detection
- Compare detection performance between real-time and multi-timeline detection system

Procedure:

- Simulate no packet (anomaly caused by outage) in test data
- Compare real-time and multi-timeline detection system
- Interval value $\delta = 10$ seconds and the number of training days $\beta = 39$ days.

Exp.4.1: Results and implications



- Multi-timeline well performs the task for anomaly caused by accidents
- Detection performances by using multi-timeline are better than those by using real-time system

Exp.4.2: No packet in training data

Objective:

- Examine robustness to incorrect training data or manipulating data by attackers
- Compare detection performance between pure normal traffic (Exp.2) and contaminated traffic in training data

Procedure: Interval values $\delta = 10$ seconds and the number of training days $\beta = 39$ days $+$ 1, 3, and 5 days of no packet incident

Exp.4.2: Example results from KNN and implications

Attack	Multi-timeline			Real-time		
	1 day	3 days	5 days	1 day	3 days	5 days
Back	+0.15%	+0.41%	+0.41%	+0.16%	+0.45%	+0.45%
IpSweep	-0.01%	+0.01%	+0.01%	+0.01%	+0.04%	+0.04%
Neptune	+0.40%	+0.86%	+0.86%	+0.22%	+0.55%	+0.55%
PortSweep	+0.06%	+0.20%	+0.20%	+0.02%	+0.04%	+0.04%
Smurf	+0.01%	+0.04%	+0.04%	+0.38%	+0.86%	+0.86%

- The multi-timeline is quite robust to incorrect training data or manipulating data
- KNN is the most robust algorithm in this experiment

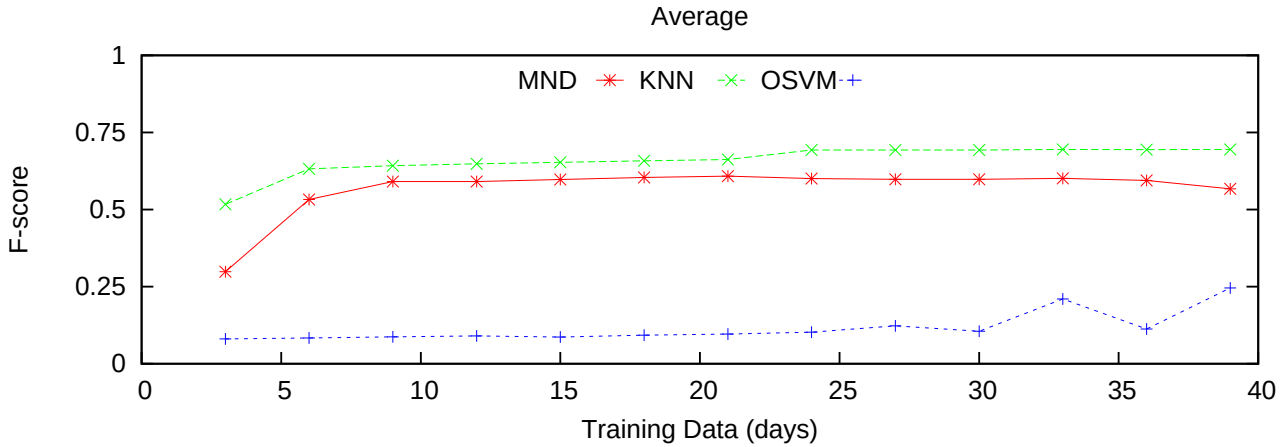
Exp.5: Vary the number of training data

Objective:

- Draw a learning curve for each algorithm
- Examine trend of each learning curve
- Identify the minimum number of training data for each algorithm

Procedure: Select the best feature and parameters for each anomaly (Exp.2), then vary the number of training data from 3 to 39 days

Exp.5: Results and implications



- Detection performance increases and converges into a constant value when the number of training data increase
- MND and KNN need only 6 or 9 days for training
- OSVM needs more than 30 days for training

Exp.6: Measure time consumption

Objective:

- Measure run time for preprocessing steps
- Measure run time for learning and detecting process
- Examine whether the proposed system performs in real time

Procedure: Generate the number of training data and features from 5 to 100, and vary interval value from 1 to 60 seconds

Exp.6: Results and implication

Run time over 1000x background traffic

Process	Extraction	Scaling	Algorithm	Total
Learning	3.13 s	0.37 ms	2.19 ms (MND)	$\approx$ 3.13 s
			0.00 ms (KNN)	$\approx$ 3.13 s
			1.86 ms (OSVM)	$\approx$ 3.13 s

Process	Extraction	Scaling	$T(\mathbf{x}) = \delta + d(\mathbf{x})$	Total
Detecting	2.96 s	0.35 ms	10 s + 0.49 ms (MND)	$\approx$ 12.96 s
			10 s + 0.25 ms (KNN)	$\approx$ 12.96 s
			10 s + 0.19 ms (OSVM)	$\approx$ 12.96 s

$T(\mathbf{x})$:total processing time; δ:interval value; $d(\mathbf{x})$:detecting time

Implications:

- Our proposed system can apply to real-time anomaly detection
- Run time depends on algorithms but total time are very similar

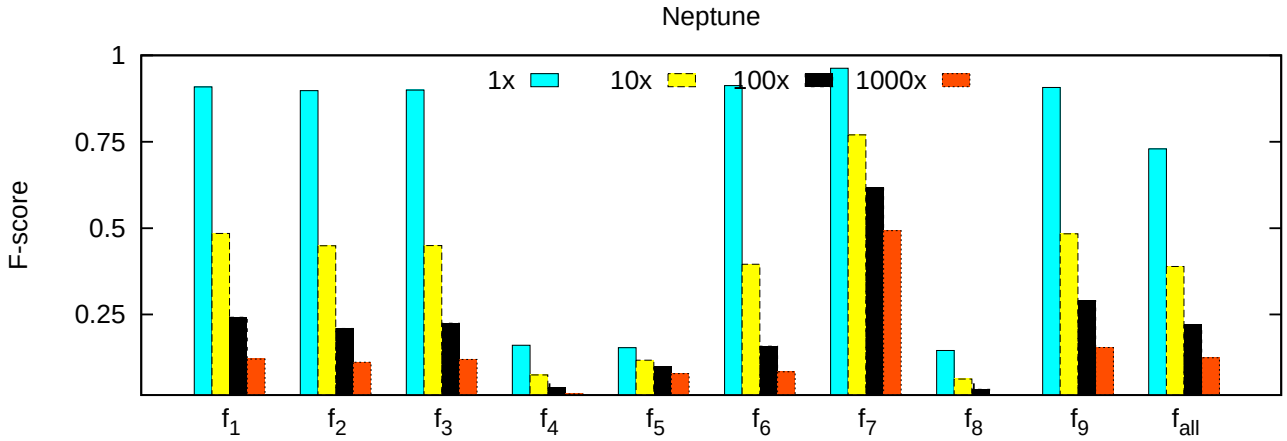
Exp.7: Vary background traffic

Objective:

- Measure detection performance over different volume of background traffic
- Compare performance of each feature over different volume of background traffic
- Examine possibility of multi-timeline detecion system on backbone network

Procedure: generate new background traffic with 10x, 100x, and 1000x times of original background traffic

Exp.7: Results and implications



- Detection performance exponentially decreased in most features
- Detection performance linear decreased in f_5 and f_7
- Multi-timeline is suitable for access networks rather than backbone networks

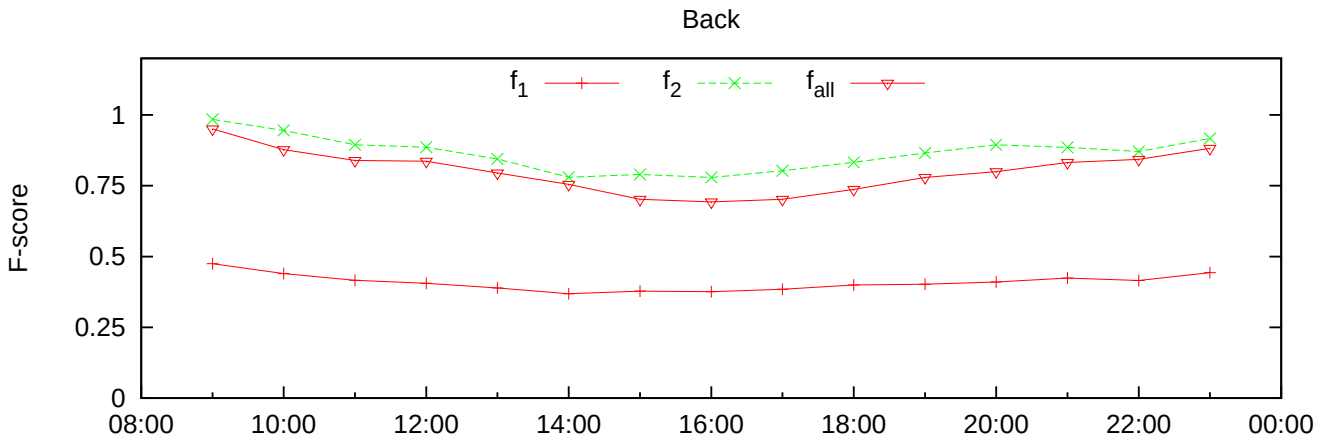
Exp.8: Alter anomaly occurrence

Objective:

- Investigate effects on time of anomaly occurrence
- Examine detection performance of anomalies over different times

Procedure: Simulate each anomaly occure at the beginnig of each hours from 9am to 11pm and select top 3 features to detect each anomaly

Exp.8: Results and implications



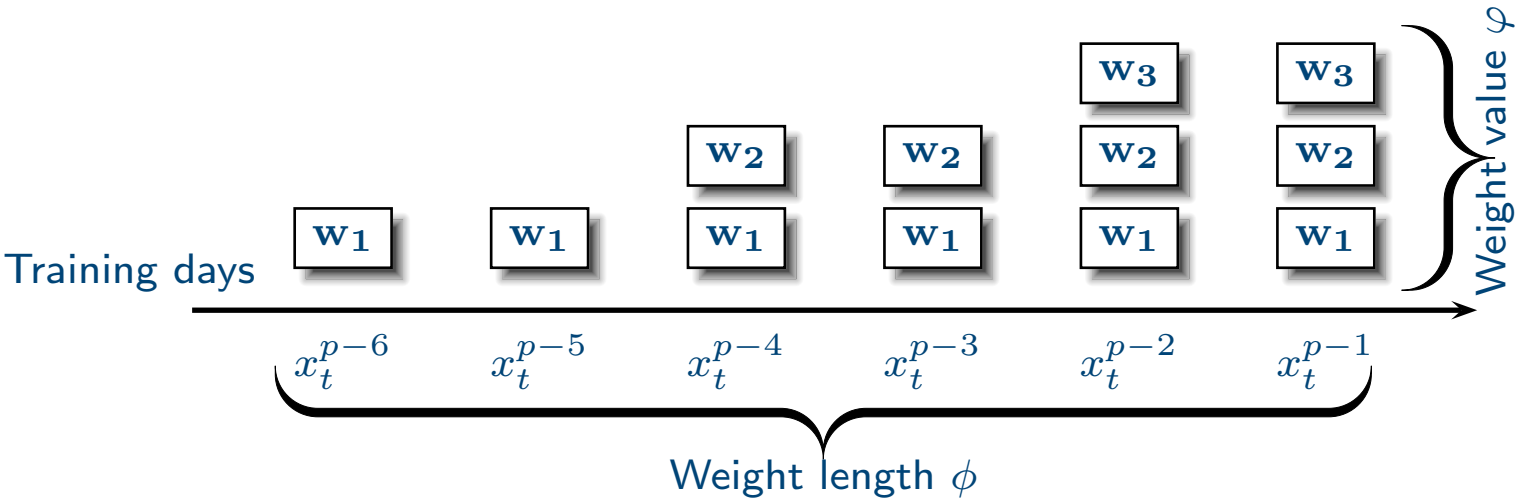
- High detection performance at low volume of background traffic
- Low detection performance at high volume of background traffic
- Time of anomaly occurrence has an effect on detection performance

Exp.9: Effect of weighting

Objective:

- Compare multi-timeline detection system with and without weighting process

Procedure: Alter the weight length $\phi = 1, 5, 10, 15, 20, 25$ days and the weight value $\varphi = 1, 3, 5, 7, 9$



Exp.9: Result and implications

- Multi-timeline system with weighting process outperforms those without weighting process
- Detection improvement is low in MND and KNN (2.7-16.6%), but the improvement of OSVM is high (3.5-112%)
- Performance of weighting technique strongly depends on learning algorithms

Conclusion

Discussion (Advantage)

- Real-time detection
  - Take 12.96 sec from packet extraction to detection (Exp.6)
- Detectability
  - Multi-timeline achieves F-score 0.69 on average and 0.99 on max (Exp.2 and 8)
  - Multi-timeline improves 19.8-31.2% than single and combined one (Exp.3)
  - Large background traffic degrades detectability exponentially (Exp.7)
  - Improve 2.7-112% by using weighting technique (Exp.9)
  - 9 days of training data and 10 sec time interval is the best (Exp.1 and 5)
- Flexibility
  - KNN algorithm is the best among 3 algorithms (Exp.2)
- Robustness

Discussion (Limitation)

- Need attack-free or almost attack-free traffic for training
- Need training data for a certain period
  - at least 9 days for MND and KNN, more than 30 days for OSVM
- Not provide any detail about anomalies
  - providing an inspector to examine the interval contained anomalies [31]
- Suitable for access networks rather than backbone networks
  - Large traffic volume hides anomalies in time bin
  - Impossible to obtain attack-free traffic in backbone

Conclusion

- Propose a system for real-time anomaly detection and related techniques
  - Evaluate various key aspects of the system
  - Introduce several input feature representations (multi-timeline and combination)
  - Comparison of three learning algorithms
  - Comparison of nine traffic features and combined feature

Future work

- Apply the proposed system to other network environment
- Provide anomaly details after alarms
- Deploy the system to an university environment

Publications

1. **Kriangkrai Limthong**; Kensuke Fukuda; Yusheng Ji; Shigeki Yamada, "Weighting Technique on Multi-timeline for Machine Learning-based Anomaly Detection System," *The 21st Asia-Pacific Conference on Communications (APCC)*, pp.-, October 2015 (Submitted).
2. **Kriangkrai Limthong**; Kensuke Fukuda; Yusheng Ji; Shigeki Yamada, "Unsupervised learning model for real-time anomaly detection in computer networks," *IEICE Transactions on Information and Systems*, vol.E97-D, no.8, pp.2084-2094, August 2014.
3. **Kriangkrai Limthong**; Kensuke Fukuda; Yusheng Ji; Shigeki Yamada, "Impact of Time Interval on Naive Bayes Classifier for Detecting Network Traffic Anomalies," *Computer Science and Engineering Conference (ICSEC), 2011 International Conference on*, pp.1-6, 7-9 September 2011.
4. **Kriangkrai Limthong**; Pirawat Watanapongse; Kensuke Fukuda, "A wavelet-based anomaly detection for outbound network traffic," *Information and Telecommunication Technologies (APSITT), 2010 8th Asia-Pacific Symposium on*, pp.1-6, 15-18 June 2010.

References

[1] M. Roesch, "Snort - lightweight intrusion detection for networks," in *Proceedings of the 13th USENIX conference on System administration*, LISA '99, (Berkeley, CA, USA), pp. 229–238, USENIX Association, 1999.
[2] E. Albin and N. Rowe, "A realistic experimental comparison of the suricata and snort intrusion-detection systems," in *Advanced Information Networking and Applications Workshops (WAINA), 2012 26th International Conference on*, pp. 122–127, 2012.
[3] J. S. White, T. Fitzsimmons, and J. N. Matthews, "Quantitative analysis of intrusion detection systems: Snort and suricata," in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, vol. 8757 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, May 2013.
[4] V. Paxson, "Bro: A system for detecting network intruders in real-time," in *Computer Networks*, pp. 2435–2463, 1999.
[5] R. Gwadera, M. J. Atallah, and W. Szpankowski, "Reliable detection of episodes in event sequences," *Knowledge and Information Systems*, vol. 7, pp. 415–437, May 2005.
[6] M. Atallah, W. Szpankowski, and R. Gwadera, "Detection of significant sets of episodes in event sequences," in *Data Mining, 2004. ICDM '04. Fourth IEEE International Conference on*, pp. 3–10, Nov 2004.
[7] G. Tandon and P. K. Chan, "Weighting versus pruning in rule validation for detecting network and host anomalies," in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, (New York, NY, USA), pp. 697–706, ACM, 2007.
[8] C. Chow, "Parzen-window network intrusion detectors," in *Proceedings of the 16 th International Conference on Pattern Recognition (ICPR'02) Volume 4 - Volume 4*, ICPR '02, (Washington, DC, USA), pp. 40385–, IEEE Computer Society, 2002.
[9] K. Sequeira and M. Zaki, "Admit: anomaly-based data mining for intrusions," in *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '02, (New York, NY, USA), pp. 386–395, ACM, 2002.
[10] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo, "A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data," in *Applications of Data Mining in Computer Security*, Kluwer, 2002.
[11] N. Wu and J. Zhang, "Factor analysis based anomaly detection," in *Information Assurance Workshop, 2003. IEEE Systems, Man and Cybernetics Society*, pp. 108–115, June 2003.
[12] M. E. Otey, A. Ghoting, and S. Parthasarathy, "Fast distributed outlier detection in mixed-attribute data sets," *Data Min. Knowl. Discov.*, vol. 12, pp. 203–228, May 2006.

References

[13] A. Lakhina, M. Crovella, and C. Diot, "Mining anomalies using traffic feature distributions," *SIGCOMM Comput. Commun. Rev.*, vol. 35, pp. 217–228, Aug. 2005.
[14] M. Thottan and C. Ji, "Anomaly detection in ip networks," *IEEE Transactions on Signal Processing*, vol. 51, pp. 2191–2204, aug. 2003.
[15] G. Nychis, V. Sekar, D. G. Andersen, H. Kim, and H. Zhang, "An empirical evaluation of entropy-based traffic anomaly detection," in *Proceedings of the 8th ACM SIGCOMM Conference on Internet Measurement*, IMC '08, (New York, NY, USA), pp. 151–156, ACM, 2008.
[16] Y. Kanda, R. Fontugne, K. Fukuda, and T. Sugawara, "Admire: Anomaly detection method using entropy-based PCA with three-step sketches," *Computer Communications*, vol. 36, no. 5, pp. 575 – 588, 2013.
[17] G. Vigna and R. A. Kemmerer, "Netstat: a network-based intrusion detection system," *Journal of Computer Security*, vol. 7, pp. 37–71, Jan. 1999.
[18] S. Jiang, X. Song, H. Wang, J.-J. Han, and Q.-H. Li, "A clustering-based method for unsupervised intrusion detections," *Pattern Recognition Letters*, vol. 27, pp. 802–810, May 2006.
[19] A. Kind, M. Stoecklin, and X. Dimitropoulos, "Histogram-based traffic anomaly detection," *IEEE Transactions on Network and Service Management*, vol. 6, pp. 110–121, june 2009.
[20] D. Anderson, T. Frivold, A. Tamaru, and A. Valdes, "Next generation intrusion detection expert system (NIDES), software users manual beta-update release." Technical Report SRI-CSL-95-0, May 1994.
[21] D. Anderson, A. Tamaru, H. Javitz, and T. F. Lunt, "Detecting unusual program behavior using the statistical component of the next-generation intrusion detection expert system NIDES," Tech. Rep. SRI-CSL-95-06, Stanford Research Institute (Menlo Park, CA US), May 1995.
[22] I. Stamouli, P. G. Argyroudis, and H. Tewari, "Real-time intrusion detection for ad hoc networks," in *In WOWMOM '05: Proceedings of the Sixth IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks (WoWMoM'05*, pp. 374–380, IEEE Computer Society, 2005.
[23] S. Amasaki and C. Lokan, "The effects of gradual weighting on duration-based moving windows for software effort estimation," in *Product-Focused Software Process Improvement* (A. Jedlitschka, P. Kuvaja, M. Kuhrmann, T. Männistö, J. Münch, and M. Raatikainen, eds.), vol. 8892 of *Lecture Notes in Computer Science*, pp. 63–77, Springer International Publishing, 2014.
[24] S. Theodoridis and K. Koutroumbas, *Pattern Recognition, Fourth Edition*. Academic Press, 4th ed., 2008.
[25] T. M. Mitchell, *Machine Learning*. New York, NY, USA: McGraw-Hill, Inc., 1 ed., 1997.

References

[26] B. Schölkopf, J. C. Platt, J. C. Shawe-Taylor, A. J. Smola, and R. C. Williamson, "Estimating the support of a high-dimensional distribution," *Neural Computation*, vol. 13, pp. 1443–1471, July 2001.
[27] K. Limthong, P. Watanapongse, and K. Fukuda, "A wavelet-based anomaly detection for outbound network traffic," in *8th Asia-Pacific Symposium on Information and Telecommunication Technologies, 2010. APSITT 2010. International Conference on*, Jun 2010.
[28] R. Lippmann, D. Fried, I. Graf, J. Haines, K. Kendall, D. McClung, D. Weber, S. Webster, D. Wyschogrod, R. Cunningham, and M. Zissman, "Evaluating intrusion detection systems: the 1998 darpa off-line intrusion detection evaluation," in *DARPA Information Survivability Conference and Exposition, 2000. DISCEX '00. Proceedings*, vol. 2, pp. 12 –26 vol.2, 2000.
[29] K. Kendall, "A database of computer attacks for the evaluation of intrusion detection systems," Master's thesis, Massachusetts Institute of Technology, Cambridge, MA, 1999.
[30] C. Thomas, V. Sharma, and N. Balakrishnan, "Usefulness of darpa dataset for intrusion detection system evaluation," in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, vol. 6973 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, Mar. 2008.
[31] J. Mazel, R. Fontugne, and K. Fukuda, "A taxonomy of anomalies in backbone network traffic," in *International Wireless Communications and Mobile Computing Conference, IWCMC 2014, Nicosia, Cyprus, August 4-8, 2014*, pp. 30–36, 2014.

References

[1] M. Roesch, "Snort - lightweight intrusion detection for networks," in *Proceedings of the 13th USENIX conference on System administration*, LISA '99, (Berkeley, CA, USA), pp. 229–238, USENIX Association, 1999.

[2] E. Albin and N. Rowe, "A realistic experimental comparison of the suricata and snort intrusion-detection systems," in *Advanced Information Networking and Applications Workshops (WAINA), 2012 26th International Conference on*, pp. 122–127, 2012.

[3] J. S. White, T. Fitzsimmons, and J. N. Matthews, "Quantitative analysis of intrusion detection systems: Snort and suricata," in *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, vol. 8757 of *Society of Photo-Optical Instrumentation Engineers (SPIE) Conference Series*, May 2013.

[4] V. Paxson, "Bro: A system for detecting network intruders in real-time," in *Computer Networks*, pp. 2435–2463, 1999.

[5] R. Gwadera, M. J. Atallah, and W. Szpankowski, "Reliable detection of episodes in event sequences," *Knowledge and Information Systems*, vol. 7, pp. 415–437, May 2005.

[6] M. Atallah, W. Szpankowski, and R. Gwadera, "Detection of significant sets of episodes in event sequences," in *Data Mining, 2004. ICDM '04. Fourth IEEE International Conference on*, pp. 3–10, Nov 2004.

[7] G. Tandon and P. K. Chan, "Weighting versus pruning in rule validation for detecting network and host anomalies," in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '07, (New York, NY, USA), pp. 697–706, ACM, 2007.

[8] C. Chow, "Parzen-window network intrusion detectors," in *Proceedings of the 16 th International Conference on Pattern Recognition (ICPR'02) Volume 4 - Volume 4*, ICPR '02, (Washington, DC, USA), pp. 40385–, IEEE Computer Society, 2002.