Unsupervised Network Anomaly Detection with Sub-Space Clustering

Pedro CASAS

AIT - Austrian Institute of Technology
Vienna, Austria

Artificial Intelligence for Data Communication Networks

Montevideo, Uruguay
December 2019

Unsupervised NIDS based on Clustering Analysis

We propose a NIDS based on clustering analysis and outliers detection.

The problem to tackle: current network security is based on an **"acquired knowledge"** perspective:

Unsupervised NIDS based on Clustering Analysis

We propose a NIDS based on clustering analysis and outliers detection.

The problem to tackle: current network security is based on an **"acquired knowledge"** perspective:

Signatures-based: detect what I ALREADY KNOW

- $(+)$ highly effective to detect what it is programmed to alert on.
- $(-)$ can not defend the network against unknown attacks.
- $(-)$ signatures are expensive to produce: human manual inspection.

Unsupervised NIDS based on Clustering Analysis

We propose a NIDS based on clustering analysis and outliers detection.

The problem to tackle: current network security is based on an **"acquired knowledge"** perspective:

Signatures-based: detect what I ALREADY KNOW

- (+) highly effective to detect what it is programmed to alert on.
- (−) can not defend the network against unknown attacks.
- (−) signatures are expensive to produce: human manual inspection.

Anomaly detection: detect what DIFFERS from WHAT I KNOW

- (+) it can detect new attacks out-of a baseline profile.
- (−) requires some kind of training for profiling.
- (−) robust and adaptive models are difficult to conceive, specially in an evolving context.

Unsupervised Detection of Network Attacks

- Unsupervised Detection based on CLUSTERING
- HYPOTHESIS: attacking flows are sparse and different from normal traffic....in some representation (traffic aggregation)!!

Unsupervised Detection of Network Attacks

- Unsupervised Detection based on CLUSTERING
- HYPOTHESIS: attacking flows are sparse and different from normal traffic....in some representation (traffic aggregation)!!

Benefits of Unsupervised-based Detection

- no previous knowledge: neither signatures nor labeled traffic.
- no need for traffic modeling or profiling.
- can detect unknown attacks.
- a major step towards self-aware monitoring.

Unsupervised Detection of Network Attacks

- Unsupervised Detection based on CLUSTERING
- HYPOTHESIS: attacking flows are sparse and different from normal traffic....in some representation (traffic aggregation)!!

Benefits of Unsupervised-based Detection

- no previous knowledge: neither signatures nor labeled traffic.
- no need for traffic modeling or profiling.
- can detect unknown attacks.
- a major step towards self-aware monitoring.

Clustering for Unsupervised Detection is CHALLENGING

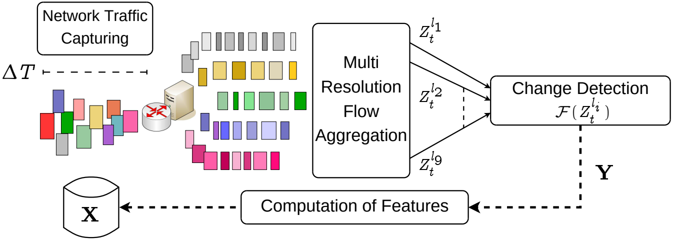
- lack of robustness: general clustering algorithms are sensitive to initialization, specification of number of clusters, etc.
- difficult to cluster high-dimensional data: structure-masking by irrelevant features, sparse spaces (“the curse of dimensionality”).

UNADA: Unsupervised Detection of Network Attacks

UNADA is a 3-steps detection algorithm:

UNADA: Unsupervised Detection of Network Attacks

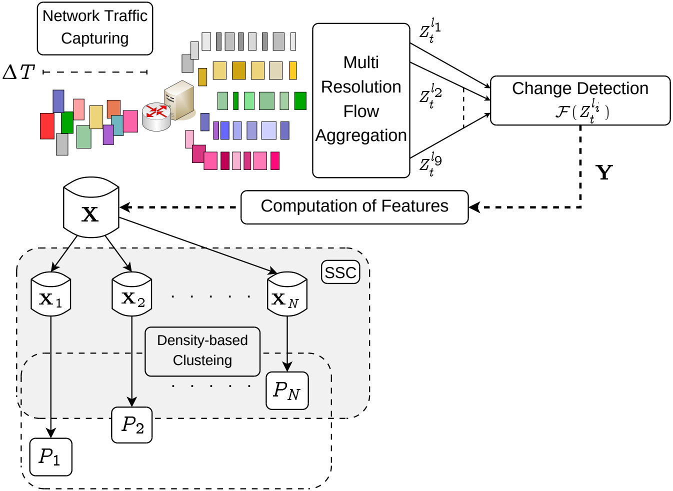
UNADA is a 3-steps detection algorithm:



(1) Multi-resolution change-detection & features computation.

UNADA: Unsupervised Detection of Network Attacks

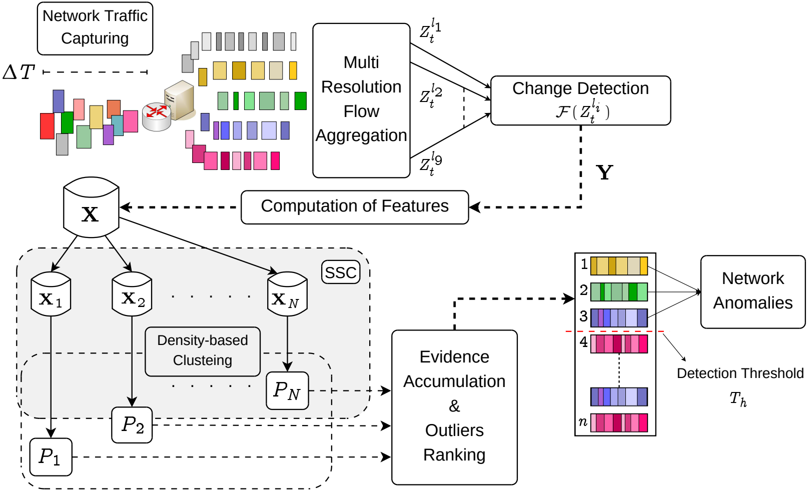
UNADA is a 3-steps detection algorithm:



(2) Sub-Space Clustering.

UNADA: Unsupervised Detection of Network Attacks

UNADA is a 3-steps detection algorithm:



(3) Evidence Accumulation and Flow Ranking.

Traffic Aggregation and Change-Detection

- Traffic is captured and aggregated in IP flows (5-tuples) every ΔT seconds, using a temporal sliding-window.

Change-detection in Multi-resolution Traffic Flows

Traffic Aggregation and Change-Detection

- Traffic is captured and aggregated in IP flows (5-tuples) every ΔT seconds, using a temporal sliding-window.
- Change-detection in simple traffic metrics to identify an anomalous time-slot (e.g., $\#pkts$, $\#bytes$, $\#IP\,flows$).

Change-detection in Multi-resolution Traffic Flows

Traffic Aggregation and Change-Detection

- Traffic is captured and aggregated in IP flows (5-tuples) every ΔT seconds, using a temporal sliding-window.
- Change-detection in simple traffic metrics to identify an anomalous time-slot (e.g., $\#pkts$, $\#bytes$, $\#IP\ flows$).

Multi-Resolution Analysis

- Analysis at different spacial resolutions, aggregating IP flows in *macro-flows*: hash-key $\{\mathtt{IPaddress/netmask}\}$.

Change-detection in Multi-resolution Traffic Flows

Traffic Aggregation and Change-Detection

- Traffic is captured and aggregated in IP flows (5-tuples) every ΔT seconds, using a temporal sliding-window.
- Change-detection in simple traffic metrics to identify an anomalous time-slot (e.g., $\#pkts$, $\#bytes$, $\#IP\ flows$).

Multi-Resolution Analysis

- Analysis at different spacial resolutions, aggregating IP flows in *macro-flows*: hash-key $\{\mathrm{IPaddress/netmask}\}$.
- Scan traffic from coarser to finer-grained macro-flows: $\mathrm{traffic\ per\ time\text{-}slot}$, $\mathrm{IP}/8$, $\mathrm{IP}/16$, $\mathrm{IP}/24$.

Change-detection in Multi-resolution Traffic Flows

Traffic Aggregation and Change-Detection

- Traffic is captured and aggregated in IP flows (5-tuples) every ΔT seconds, using a temporal sliding-window.
- Change-detection in simple traffic metrics to identify an anomalous time-slot (e.g., $\#pkts$, $\#bytes$, $\#IP\ flows$).

Multi-Resolution Analysis

- Analysis at different spacial resolutions, aggregating IP flows in *macro-flows*: hash-key $\{\mathrm{IPaddress}/\mathrm{netmask}\}$.
- Scan traffic from coarser to finer-grained macro-flows: $\mathrm{traffic\ per\ time\text{-}slot}$, $\mathrm{IP}/8$, $\mathrm{IP}/16$, $\mathrm{IP}/24$.
- Scan in both directions (IPsrc and IPdst) permits to detect $1\text{-to-}1$, $1\text{-to-}N$, and $N\text{-to-}1$ attacks of different intensities.

Clustering for Anomaly Detection

- Let $\mathbf{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$ be the set of n macro-flows in the flagged time slot, aggregated at $\mathrm{IP}/32$.

Clustering for Anomaly Detection

- Let $\mathbf{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$ be the set of n macro-flows in the flagged time slot, aggregated at $\mathrm{IP}/32$.
- Each macro-flow $\mathbf{y}_i \in \mathbf{Y}$ is described by a set of m traffic features: $\mathbf{x}_i = (x_i(1), .., x_i(m)) \in \mathbb{R}^m$.

Clustering for Anomaly Detection

- Let $\mathbf{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$ be the set of n macro-flows in the flagged time slot, aggregated at $\mathrm{IP}/32$.
- Each macro-flow $\mathbf{y}_i \in \mathbf{Y}$ is described by a set of m traffic features: $\mathbf{x}_i = (x_i(1), .., x_i(m)) \in \mathbb{R}^m$.
- Number of sources & destinations ($\mathrm{nSrcs}, \mathrm{nDsts}$), packet rate ($\mathrm{nPkts}/\mathrm{sec}$), fraction of SYN packets ($\mathrm{nSYN}/\mathrm{nPkts}$), etc.

Clustering for Anomaly Detection

- Let $\mathbf{Y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_n\}$ be the set of n macro-flows in the flagged time slot, aggregated at $\mathrm{IP}/32$.

- Each macro-flow $\mathbf{y}_i \in \mathbf{Y}$ is described by a set of m traffic features: $\mathbf{x}_i = (x_i(1), .., x_i(m)) \in \mathbb{R}^m$.

- Number of sources & destinations ($\mathrm{nSrcs}, \mathrm{nDsts}$), packet rate ($\mathrm{nPkts}/\mathrm{sec}$), fraction of SYN packets ($\mathrm{nSYN}/\mathrm{nPkts}$), etc.

- $\mathbf{X} = \{\mathbf{x}_1, .., \mathbf{x}_n\}$ is the complete matrix of features, referred to as the *feature space*.

Clustering for Anomaly Detection

$\mathbf{X}$ is a black box

How to detect an anomalous macro-flow in $\mathbf{X}$ via clustering?

- "Simple idea": cluster $\mathbf{X}$, big-size clusters correspond to normal-flows, outliers are anomalies.

Clustering for Anomaly Detection

$\mathbf{X}$ is a black box

How to detect an anomalous macro-flow in $\mathbf{X}$ via clustering?

- "Simple idea": cluster $\mathbf{X}$, big-size clusters correspond to normal-flows, outliers are anomalies.

Clustering for Anomaly Detection

IS NOT THAT SIMPLE!!!

How to detect an anomalous macro-flow in $\mathbf{X}$ via clustering?

- “Simple idea”: cluster $\mathbf{X}$, big-size clusters correspond to normal-flows, outliers are anomalies.

Sub-Space Clustering

How to Improve Robustness and Clustering Performance?

- Idea: combine the information provided by multiple partitions of $\mathbf{X}$ to "filter noise", easing the discovery of outliers.

Sub-Space Clustering

How to Improve Robustness and Clustering Performance?

- Idea: combine the information provided by multiple partitions of $\mathbf{X}$ to "filter noise", easing the discovery of outliers.
- How to produce multiple partitions? $\rightarrow$ Sub-Space Clustering.

Sub-Space Clustering

How to Improve Robustness and Clustering Performance?

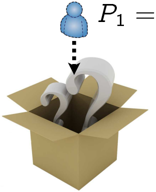
- Idea: combine the information provided by multiple partitions of $\mathbf{X}$ to "filter noise", easing the discovery of outliers.
- How to produce multiple partitions? $\longrightarrow$ Sub-Space Clustering.
- Each sub-space $\mathbf{X}_i \subset \mathbf{X}$ is obtained by projecting $\mathbf{X}$ in k out of the m original dimensions. Density-based clustering applied to $\mathbf{X}_i$.

Sub-Space Clustering

How to Improve Robustness and Clustering Performance?

- Idea: combine the information provided by multiple partitions of $\mathbf{X}$ to "filter noise", easing the discovery of outliers.
- How to produce multiple partitions? $\rightarrow$ Sub-Space Clustering.
- Each sub-space $\mathbf{X}_i \subset \mathbf{X}$ is obtained by projecting $\mathbf{X}$ in k out of the m original dimensions. Density-based clustering applied to $\mathbf{X}_i$.

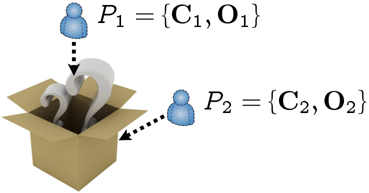
$P_1 = \{\mathbf{C}_1, \mathbf{O}_1\}$



Sub-Space Clustering

How to Improve Robustness and Clustering Performance?

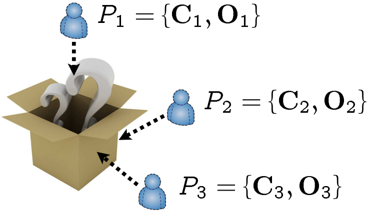
- Idea: combine the information provided by multiple partitions of $\mathbf{X}$ to “filter noise”, easing the discovery of outliers.
- How to produce multiple partitions? $\rightarrow$ Sub-Space Clustering.
- Each sub-space $\mathbf{X}_i \subset \mathbf{X}$ is obtained by projecting $\mathbf{X}$ in k out of the m original dimensions. Density-based clustering applied to $\mathbf{X}_i$.



Sub-Space Clustering

How to Improve Robustness and Clustering Performance?

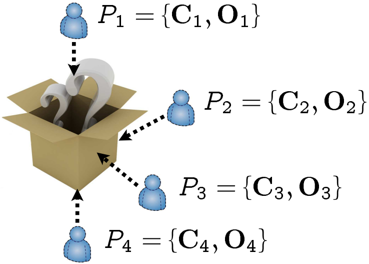
- Idea: combine the information provided by multiple partitions of $\mathbf{X}$ to "filter noise", easing the discovery of outliers.
- How to produce multiple partitions? $\rightarrow$ Sub-Space Clustering.
- Each sub-space $\mathbf{X}_i \subset \mathbf{X}$ is obtained by projecting $\mathbf{X}$ in k out of the m original dimensions. Density-based clustering applied to $\mathbf{X}_i$.



Sub-Space Clustering

How to Improve Robustness and Clustering Performance?

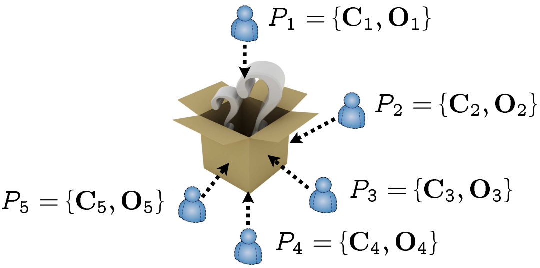
- Idea: combine the information provided by multiple partitions of $\mathbf{X}$ to "filter noise", easing the discovery of outliers.
- How to produce multiple partitions? $\rightarrow$ Sub-Space Clustering.
- Each sub-space $\mathbf{X}_i \subset \mathbf{X}$ is obtained by projecting $\mathbf{X}$ in k out of the m original dimensions. Density-based clustering applied to $\mathbf{X}_i$.



Sub-Space Clustering

How to Improve Robustness and Clustering Performance?

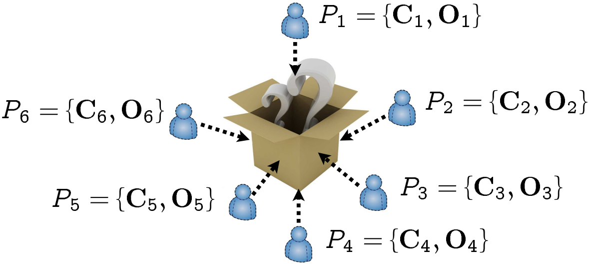
- Idea: combine the information provided by multiple partitions of $\mathbf{X}$ to "filter noise", easing the discovery of outliers.
- How to produce multiple partitions? $\rightarrow$ Sub-Space Clustering.
- Each sub-space $\mathbf{X}_i \subset \mathbf{X}$ is obtained by projecting $\mathbf{X}$ in k out of the m original dimensions. Density-based clustering applied to $\mathbf{X}_i$.



Sub-Space Clustering

How to Improve Robustness and Clustering Performance?

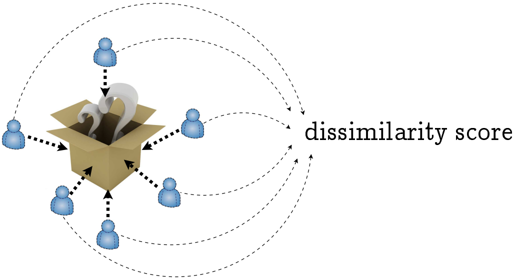
- Idea: combine the information provided by multiple partitions of $\mathbf{X}$ to "filter noise", easing the discovery of outliers.
- How to produce multiple partitions? $\rightarrow$ Sub-Space Clustering.
- Each sub-space $\mathbf{X}_i \subset \mathbf{X}$ is obtained by projecting $\mathbf{X}$ in k out of the m original dimensions. Density-based clustering applied to $\mathbf{X}_i$.



Sub-Space Clustering

How to Improve Robustness and Clustering Performance?

- Idea: combine the information provided by multiple partitions of $\mathbf{X}$ to "filter noise", easing the discovery of outliers.
- How to produce multiple partitions? $\rightarrow$ Sub-Space Clustering.
- Each sub-space $\mathbf{X}_i \subset \mathbf{X}$ is obtained by projecting $\mathbf{X}$ in k out of the m original dimensions. Density-based clustering applied to $\mathbf{X}_i$.



Evidence Accumulation for Outliers Ranking

Evidence Accumulation to combine the results of SSC:

- Build a new dissimilarity measure $D = \{d_1,\, d_2, \ldots, d_n\}$: d_i measures how different is flow i from the majority of the traffic.

Evidence Accumulation for Outliers Ranking

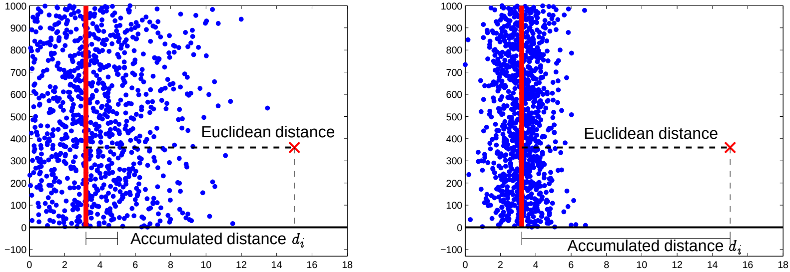
Evidence Accumulation to combine the results of SSC:

- Build a new dissimilarity measure $D = \{d_1, d_2, \ldots, d_n\}$: d_i measures how different is flow i from the majority of the traffic.
- Accumulate in d_i the *weighted* distance from outliers to biggest cluster in each sub-space.

Evidence Accumulation for Outliers Ranking

Evidence Accumulation to combine the results of SSC:

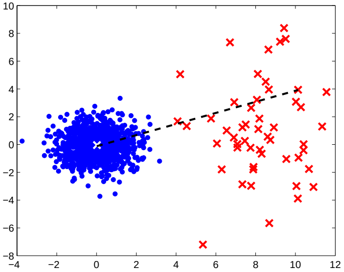
- Build a new dissimilarity measure $D = \{d_1, d_2, \ldots, d_n\}$: d_i measures how different is flow i from the majority of the traffic.
- Accumulate in d_i the *weighted* distance from outliers to biggest cluster in each sub-space.
- Most dissimilar flows w.r.t. D are flagged as anomalies.

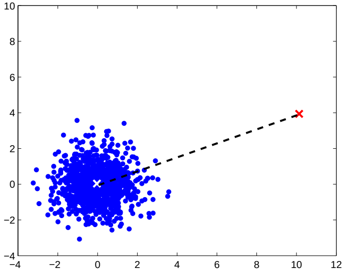


Evidence Accumulation for Outliers Ranking

Evidence Accumulation to combine the results of SSC:

- Build a new dissimilarity measure $D = \{d_1, d_2, \ldots, d_n\}$: d_i measures how different is flow i from the majority of the traffic.
- Accumulate in d_i the *weighted* distance from outliers to biggest cluster in each sub-space.
- Most dissimilar flows w.r.t. D are flagged as anomalies.



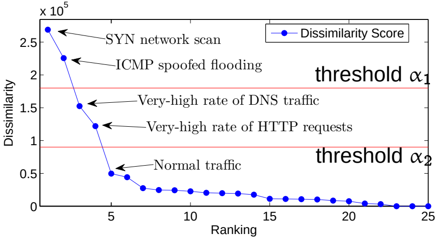


Attacks Detection in MAWI Traffic

- MAWI: packet traces from link Japan-U.S.A. of the WIDE network.
- Ex: worm scanning, ICMP flooding attack, $\mathrm{IPsrc}/32$ macro-flows.

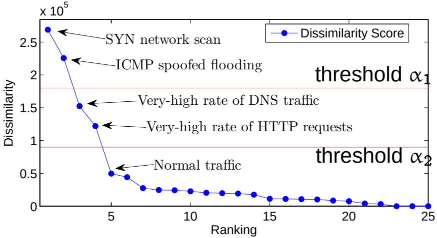
Attacks Detection in MAWI Traffic

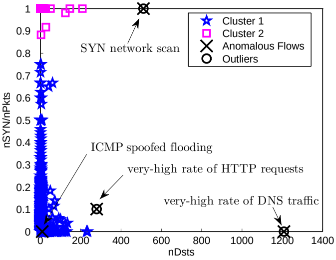
- MAWI: packet traces from link Japan-U.S.A. of the WIDE network.
- Ex: worm scanning, ICMP flooding attack, $\mathrm{IPsrc}/32$ macro-flows.

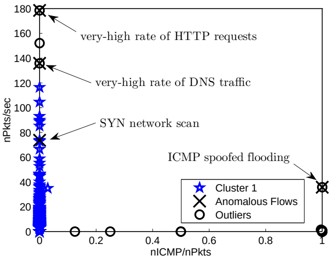


Attacks Detection in MAWI Traffic

- MAWI: packet traces from link Japan-U.S.A. of the WIDE network.
- Ex: worm scanning, ICMP flooding attack, $\texttt{IPsrc}/32$ macro-flows.







Ground-Truth (GT) Attacks in METROSEC & MAWI

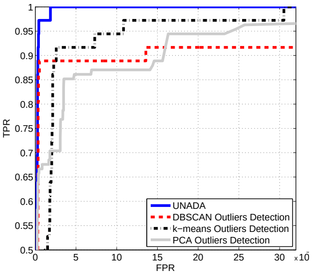
- METROSEC, DDoS attacks of different intensities (70% to 4%), $\mathrm{IPdst}/32$ macro-flows.

Ground-Truth (GT) Attacks in METROSEC & MAWI

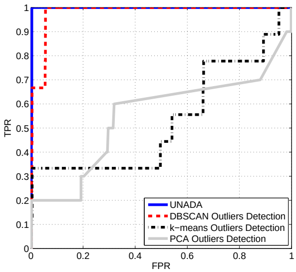
- METROSEC, DDoS attacks of different intensities (70% to 4%), $\mathrm{IPdst}/32$ macro-flows.
- MAWI, worm scanning (Sasser and Dabber), DoS/DDoS attacks, GT attacks detected by signatures + Anomaly Detection.

Ground-Truth (GT) Attacks in METROSEC & MAWI

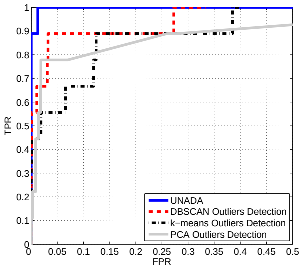
- METROSEC, DDoS attacks of different intensities (70% to 4%), $\mathrm{IPdst}/32$ macro-flows.
- MAWI, worm scanning (Sasser and Dabber), DoS/DDoS attacks, GT attacks detected by signatures + Anomaly Detection.
- Compared against traditional unsupervised approaches: DBSCAN based, k-means based, and PCA based outliers detection.



(a) MAWI, IPsrc key.



(b) MAWI, IPdst key.



(c) METROSEC, IPdst key.

Detectiong Attacks in KDD99

- DARPA - KDD99 dataset, DoS (udp storm, pod, appache flooding, etc.), scans (port, net), Remote-2-Local attacks (guess password, imap, http tunnel, etc.), User-2-Root (buffer overflows).

Detectiong Attacks in KDD99

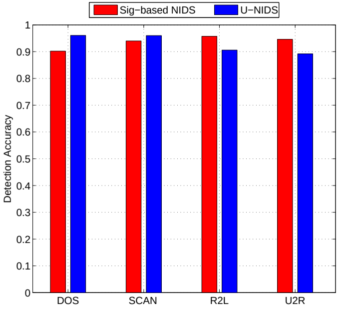
- DARPA - KDD99 dataset, DoS (udp storm, pod, appache flooding, etc.), scans (port, net), Remote-2-Local attacks (guess password, imap, http tunnel, etc.), User-2-Root (buffer overflows).
- Compared against signature-based detection $\longrightarrow$ NIDS based on decision trees (C4.5).

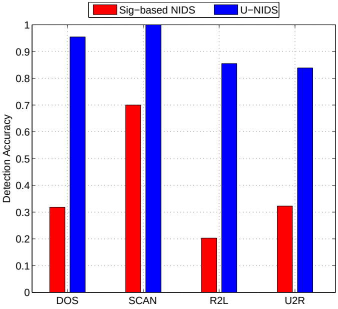
Detectiong Attacks in KDD99

- DARPA - KDD99 dataset, DoS (udp storm, pod, appache flooding, etc.), scans (port, net), Remote-2-Local attacks (guess password, imap, http tunnel, etc.), User-2-Root (buffer overflows).
- Compared against signature-based detection $\rightarrow$ NIDS based on decision trees (C4.5).
- Trees are constructed for a set of known attacks, and tested with *unknown* attacks.

Detectiong Attacks in KDD99

- DARPA - KDD99 dataset, DoS (udp storm, pod, appache flooding, etc.), scans (port, net), Remote-2-Local attacks (guess password, imap, http tunnel, etc.), User-2-Root (buffer overflows).
- Compared against signature-based detection $\longrightarrow$ NIDS based on decision trees (C4.5).
- Trees are constructed for a set of known attacks, and tested with *unknown* attacks.





References

- R. O. Duda, P. E. Hart, and D. G. Stork, “Pattern Classification (2nd edition)”, Wiley, ISBN: 978-0-471-05669-0, 2000.
- C. M. Bishop, “Pattern Recognition and Machine Learning”, Springer-Verlag, ISBN: 0-387-31073-8, 2006.
- A. K. Jain, R. P.W. Duin, and J. Mao “Statistical Pattern Recognition: A Review”, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 1, 2000.
- A. K. Jain, “Data Clustering: 50 Years Beyond K-means”, Pattern Recognition Letters, vol. 31, issue 8, 2010.

Thank You for Your Attention!! 

Remarks & Questions?