Traffic Anomaly Detection Using K-Means Clustering

Gerhard Münz, Sa Li, Georg Carle
Computer Networks and Internet
Wilhelm Schickard Institute for Computer Science
University of Tuebingen, Germany

***Abstract*—Data mining techniques make it possible to search large amounts of data for characteristic rules and patterns. If applied to network monitoring data recorded on a host or in a network, they can be used to detect intrusions, attacks and/or anomalies. This paper gives an introduction to Network Data Mining, i.e. the application of data mining methods to packet and flow data captured in a network, including a comparative overview of existing approaches. Furthermore, we present a novel flow-based anomaly detection scheme based on the K-mean clustering algorithm. Training data containing unlabeled flow records are separated into clusters of normal and anomalous traffic. The corresponding cluster centroids are used as patterns for computationally efficient distance-based detection of anomalies in new monitoring data. We provide a detailed description of the data mining and the anomaly detection processes, and present first experimental results.**

I. Introduction

Increasing processing and storage capacities of computer systems make it possible to record and store growing amounts of data in an inexpensive way. Even though more data potentially contains more information, it is often difficult to interpret a large amount of collected data and to extract new and interesting knowledge. The term data mining is used for methods and algorithms that allow analyzing data in order to find rules and patterns describing the characteristic properties of the data. Data mining techniques are attractive as they can be applied to any kind of data in order to learn more about hidden structures and correlations. However, this universality also has shortcoming because the generated knowledge does not have to be meaningful or useful. It is necessary to evaluate and interpret the data mining results with respect to a specific goal or purpose of the data analysis.

Intrusion detection systems (IDS) process large amounts of monitoring data. As an example, a host-based IDS examines log files on a computer (or host) in order to detect suspicious activities. A network-based IDS, on the other hand, searches network monitoring data for harmful packets or packet flows. In the late 1990s, progress in data mining research and the necessity to find better methods for network and host-based intrusion detection resulted in research activities attempting to deploy data mining techniques for anomaly and attack detection. Following this trend, the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining dedicated the KDD-CUP'99 competition to intrusion detection [1].

This paper focuses on *Network Data Mining* (NDM), i.e. the application of data mining methods to monitoring data recorded in computer networks. Section II gives an introduction to NDM and provides an overview and comparison of existing approaches. Most of them rely on rule learning algorithms.

In Section III, we present a novel NDM approach for anomaly detection based on the K-means clustering algorithm. The raw data consists of flow records that have been exported by routers and/or network monitors using Cisco Netflow [2] or the IPFIX protocol [3]. Flow records are easily available in many networks since flow monitoring techniques are already widely deployed for accounting purposes. We transform flow records into datasets with a small number of features for predefined time intervals and service-specific port numbers. Our goal is to identify time intervals showing anomalous traffic behavior, as it may be caused by network malfunctions or malicious attack traffic. The processing steps of our approach can be summarized as follows:

1) Training data containing flow records of both normal and anomalous traffic are transformed into feature datasets.
2) The datasets are divided into different clusters for normal and anomalous traffic using the K-means clustering algorithm.
3) The resulting cluster centroids are deployed for fast detection of anomalies in new monitoring data based on simple distance calculations.

While clustering monitoring data and identifying anomalies based on outlier detection has already been tried before, we are not aware of previous attempts generating additional clusters for anomalous traffic as we do.

Although a profound evaluation is still subject to ongoing work, we are already able to present some initial results in Section IV, demonstrating the general feasibility of the proposed anomaly detection method. Section V finally concludes the paper with an outlook on future work.

II. Network Data Mining

NDM may serve two different purposes. First, knowledge about the analyzed monitoring data is generated. This allows determining dominant characteristics and identifying outliers within the data records that can be considered as anomalous or suspicious. Secondly, NDM can be deployed to define rules or patterns that are typical for specific kinds of traffic,

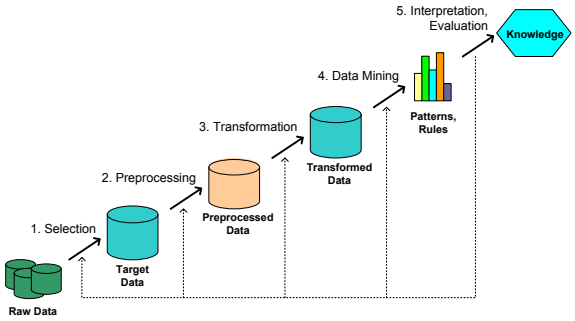

Fig. 1. Knowledge Discovery in Databases [4]

e.g. normal web traffic or traffic observed during a denial of service (DoS) attack. These rules and patterns can be used to analyze new sets of monitoring data and to check if these show similar properties and characteristics as the original data. Obvious applications profiting from such rules and patterns are network-based intrusion detection systems (NIDS) and traffic analyzers that characterize and classify traffic flows.

A. The KDD Model

NDM is applied to large amounts of monitoring data describing packet, flow, or connection attributes and statistics. These data are usually stored and processed in a database in order to retrieve the desired rules and patterns. Following the KDD (Knowledge Discovery in Databases) model [4] shown in Figure 1, five processing steps can be distinguished:

1) **Selection of raw data.** As an example, the data mining process can be restricted to monitoring data recorded in a specific period of time or observed at a certain monitor in the network.
2) **Data preprocessing.** Depending on the monitoring data and the data mining algorithm, it may be necessary or beneficial to perform cleaning and filtering of the data in order to avoid the generation of misleading or inappropriate rules or patterns. Disturbances can be noise, known changes such as dependencies on the time of day, missing data fields in some of the data records etc.
3) **Data transformation.** Applying data mining methods to raw monitoring data usually does not yield useful results since the numerous attributes within the original records are not equally relevant for the goal of the data mining task. Hence, a transformation has to be found that converts the raw data into datasets with a small number of relevant features. Also, it might be necessary to aggregate the data in order to decrease the number of datasets as well as the required memory and processing resources of the data mining algorithm.
4) **Data mining.** In this step, a data mining algorithm is applied in order to find rules or patterns.
5) **Interpretation and Evaluation.** It has to be evaluated if the data mining step generated useful results and which subset of the rules and patterns contains the most valuable information. In order to validate the rules and patterns, it might be necessary to repeat the process with a different selection of raw data.

As can be seen, the KDD model can be adopted to describe the necessary NDM processing steps at an abstract level although concrete realizations do not have to strictly follow this model. For example, some steps may be omitted and others performed in a slightly different order or jointly within in a single step.

B. Classification of Existing NDM Approaches

In this subsection, we present a selection of NDM approaches and compare them according to the following properties:

- **Type of raw data.** NDM can be applied to different types of monitoring data. Typical examples are packet data such as header information and/or payload, and statistical data about packet flows or connections including start and end time stamps, number of exchanged packets and bytes etc. Obviously, the data mining process as well as the achievable result depends very heavily on the type of raw data.
- **Extracted features.** The raw monitoring data is preprocessed and transformed in order to extract a set of relevant features. Extracting the set of most relevant features is essential for good data mining results.
- **Applied data mining algorithm.** A whole bunch of data mining algorithms have been presented in literature. They can be categorized according to the accomplished data mining task, e.g. clustering, classification, mining of association rules, characterization and discrimination etc.
- **Generated knowledge.** Finally, NDM approaches can be distinguished by the generated knowledge about the analyzed monitoring data and the capability to generate rules or patterns that can be applied to new sets of monitoring data.

In the following, we briefly describe the selected NDM approaches and compare them to our anomaly detection scheme. Table I summarizes the comparison based on the criteria listed above.

In the late 1990s, Lee and Stolfo studied different data mining methods for intrusion detection based on various kinds of monitoring and audit data. In the area of NDM, they used *tcpdump* [11] output as raw data and transformed it into connection records with the following features [5]: start time, duration, participating hosts, ports, the statistics of the connection, connection termination flag, and protocol (TCP or UDP). Additional attributes were added, e.g. the number of connections to the same host or service in the last w seconds. The rule-learning engine RIPPER [12] is applied to the training data with feature datasets labeled as normal or attack in order to generate rules for probing and denial-of-service attacks. These rules can later be used for misuse detection, i.e. in order to detect similar attacks in new monitoring data. However, they are not suitable for anomaly detection or the detection of new attacks showing a different behavior.

TABLE I
CLASSIFICATION OF NDM APPROACHES

Approach	Raw data	Features	DM algorithm	Knowledge
Lee & Stolfo (1998) [5]	Packets (tcpdump)	Connection records	Rule learning	Association rules and frequent episodes
Dokas et al. (2002) [6]	Packets (tcpdump) converted into TCP connection records	Time-based features	Outlier detection	Anomalies in analyzed data
Ertoz et al. (2003) [7]	Flow records (Cisco Netflow)	Time-based and connection-based features	Outlier detection	Anomalies in analyzed data
Esposito et al. (2005) [8]	Packets (tcpdump)	Connection records	Rule learning	Classification rules
Barbará et al. (2001) [9]	Packets (tcpdump)	Connection records	Rule learning	Association and classification rules
Luo et al. (2000) [10]	Packets (tcpdump)	Counters for TCP SYN/FIN/RST packets and number of different destination ports	Rule learning	Fuzzy association rules and fuzzy frequent episodes
Our approach	Flow records (Cisco Netflow, IPFIX)	Counters of bytes, packets, active flows for different time intervals and service-specific ports	K-means clustering	Centroids for normal and anomalous clusters

Researchers of the Computer Science Department at University of Minnesota performed similar studies applying rule-learning algorithms to connection records supplied as part of the KDD-CUP'99 competition [13] which consist of similar features as used by Lee and Stolfo. Later, they focused on outlier detection in order to detect anomalies based on time-based and connection-based feature sets [6], [14]. The LOF (local outlier factor) detection method [15] was selected to be integrated into the Minnesota Intrusion Detection System (MINDS) [7]. In MINDS, connection records ranked as highly anomalous by the LOF algorithm are further examined by an association pattern analysis module in order to generate rules and patterns for misuse detection. MINDS uses Cisco Netflow data as raw data of the data mining process.

Esposito et al. [8] deploy ToolDiag, a pattern recognition toolbox, to identify a small subset with the maximum discriminating power out of the connection features used by Lee and Stolfo. After, network behavior patterns are generated by applying a rule-learning algorithm to the selected features. Unfortunately, the authors do not disclose which connections features they finally used. Barbará et al. achieved very good results in the DARPA'99 intrusion detection test applying classification and association rules to connection records [9] . In [10], [16], the usage of fuzzy association rules and fuzzy frequent episodes is proposed.

In the next section, we present a novel NDM approach which is comparable to MINDS since it also uses flow records as input of the data mining process. However, we do not aim at classifying individual flows as normal or anomalous, but we use a smaller feature set in order to detect time intervals at which traffic anomalies occur. While MINDS groups normal flow records in a single cluster, we assume that normal and anomalous traffic form different clusters in the feature space. Therefore, we deploy K-means clustering to determine clusters for normal and anomalous traffic. Finally, we use a fast distance-based method to classify new monitoring data and detect outliers.

III. K-MEAN CLUSTERING OF MONITORING DATA

Our NDM approach deploys the K-mean clustering algorithm [17] in order to separate time intervals with normal and anomalous traffic in the training dataset. The resulting cluster centroids are then used for fast anomaly detection in new monitoring data.

In the next subsection, we describe the raw data and the extracted features that serve as input for the data mining algorithm. Thereafter, we explain the K-mean clustering algorithm and the resulting patterns. Finally, we show how the patterns can be used for classification and outlier detection.

A. Raw Data and Extracted Features

As input to the data mining process, we make use of flow records which are available in many networks due to the wide deployment of Cisco Netflow [2], [18] and compatible exporters. In this context, a flow is defined as a unidirectional stream of IP packets identified by a common IP five-tuple (protocol type, source IP address, destination IP address, source port, destination port). Apart from the IP five-tuple information, a flow record contains statistical information such as the number of packets and bytes observed in a certain period of time.

First, flow records are classified according to the transport protocol and predefined port numbers which are typical for commonly used services. As an example, TCP records with

source or destination port 80 are grouped as Web/HTTP traffic. The reason for this classification is that normal traffic looks very different depending on the service or application. Distinguishing flows by their protocol and service-specific port numbers thus allows applying the K-means clustering algorithm separately for different services identified by their (*protocol, port*) pairs. Flow records that do not fit into any of the predefined service classes are assigned to the corresponding default class for TCP, UDP, or ICMP traffic.

Separately for each class, we aggregate and transform flow records into datasets for equally spaced time intervals, considering the start time of each flow. The lengths of the time interval is chosen to be equal to or greater than the maximum expected flow duration in order to avoid that records of long-lasting flows cover multiple intervals. Note that the maximum flow duration is a configuration parameter of the router or network monitor exporting the flow records. Hence, setting this parameter to a small value (e.g. 1 minute or 10 seconds) allows generating datasets at a comparable small time scale. Each dataset contains the following features:

- *Total number of packets* sent from and to the given port in the considered time interval.
- *Total number of bytes* sent from and to the given port in the considered time interval.
- *Number of different source-destination pairs*[1] matching the given service-specific port and protocol and being observed in the considered time interval.

The motivation behind this feature selection is that the number of packets and bytes allows detecting anomalies in traffic volume, while the third feature helps detecting network and port scans as well as distributed attacks, which both result in an increased number of source-destination pairs.

The K-means clustering algorithm is applied to these datasets as explained in the next subsection.

B. K-means Clustering

K-means clustering [17] is a clustering analysis algorithm that groups objects based on their feature values into K disjoint clusters. Objects that are classified into the same cluster have similar feature values. K is a positive integer number specifying the number of clusters, and has to be given in advance. Here are the four steps of the K-means clustering algorithm:

1) Define the number of clusters K.
2) Initialize the K cluster centroids. This can be done by arbitrarily dividing all objects into K clusters, computing their centroids, and verifying that all centroids are different from each other. Alternatively, the centroids can be initialized to K arbitrarily chosen, different objects.
3) Iterate over all objects and compute the distances to the centroids of all clusters. Assign each object to the cluster with the nearest centroid.
4) Recalculate the centroids of both modified clusters.
5) Repeat step 3 until the centroids do not change any more.

[1]considering IP addresses and port numbers

A distance function is required in order to compute the distance (i.e. similarity) between two objects. The most commonly used distance function is the Euclidean one which is defined as:

$$d(x,y) = \sqrt{\sum_{i=1}^{m} (x_i - y_i)^2}$$

where $x = (x_1, ..., x_m)$ and $y = (y_1, ..., y_m)$ are two input vectors with m quantitative features. In the Euclidean distance function, all features contribute equally to the function value. However, since different features are usually measured with different metrics or at different scales, they must be normalized before applying the distance function.

An alternative to Euclidean distance is the Mahalanobis distance function that uses the inverse covariance matrix S^{-1} to reflect statistical correlations between different features:

$$d(x,y) = \sqrt{(x-y)^T S^{-1} (x-y)}$$

However, calculating and inverting the covariance matrix is computationally demanding for feature vectors with a large number of dimensions.

For initial evaluation of the proposed anomaly detection approach, we used a weighted Euclidean distance defined as:

$$d(x,y) = \sqrt{\sum_{i=1}^{m} \left(\frac{x_i - y_i}{s_i} \right)^2}$$

where s_i is an empirical normalization and weighing factor of the i-th feature. Note that the larger s_i, the smaller is the influence of the i-th feature on the distance. We found that good coefficients for the number of *packets*, *bytes*, and source-destination pairs (*src-dst*) are $s_{packets} = s_{bytes} = 5$ and $s_{src-dst} = 1$.

We apply the K-means clustering algorithm to training datasets which may contain normal and anomalous traffic without being labeled as such in advance. The rationale behind this approach is the assumption that normal and anomalous traffic form different clusters in the features space. Of course, the data may contain outliers which do not belong to a bigger cluster, yet this does not disturb the K-means clustering process as long as the number of outliers is small. As already mentioned, the clustering is done individually for the predefined services, identified by their typical *(protocol, port)* pair, as well as for the default classes that cover the remaining flows distinguished by the *protocol* value only.

The clustering algorithm divides the training data into K clusters, but does not determine if a cluster reflect time intervals of normal or anomalous traffic. This decision has to be made manually or by heuristics. For example, a higher average in the number of packets can be taken as an indicator for an anomalous cluster. It may occur that clusters are very close to each other. This can have several reasons: Either the number of clusters K has been badly chosen or the training data is very homogeneous, e.g. because it does not contain any anomalous traffic or because the anomalous traffic looks very

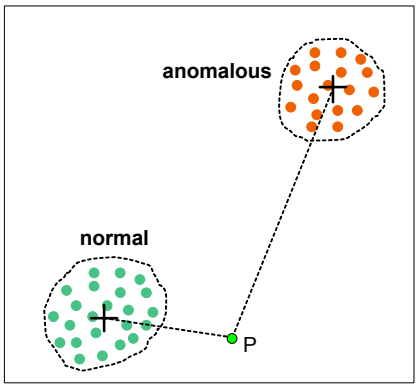

Fig. 2. Classification for $K = 2$

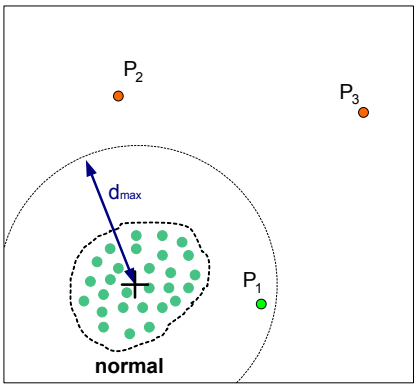

Fig. 3. Outlier detection

similar to normal traffic. Nevertheless, the cluster centroids can still be used for outlier detection as explained in the next subsection.

An essential problem of the K-means clustering method is to define an appropriate number of clusters K. As initial value, we chose $K = 2$, assuming that normal and anomalous traffic in the training data form two different clusters. Obviously, a different number of clusters may result in better clusters, e.g. if the considered service already shows distinct periods of very low and very high traffic volume under normal conditions. However, the determination of an optimum number of clusters based on a cluster evaluation criterion is subject to our ongoing research and not covered in this paper.

C. Classification and Outlier Detection

The K-means clustering process results in cluster centroids for normal and anomalous traffic which can be used to detect anomalies in new flow records monitored in the same network. New flow records have to be preprocessed and transformed like the training data in order to obtain the same features. For the purpose of anomaly detection, we deploy two distance-based methods – classification and outlier detection – that both use the K-means clustering results and that can be applied individually or in a combined way.

Classification. The distances to the cluster centroids of the corresponding traffic class are calculated using the weighted Euclidean distance function. An object is classified as normal if it is closer to the normal cluster centroid than to the anomalous one, and vice versa. This is illustrated in Figure 2 with a two-dimensional feature space: Object P is closer to the normal cluster, therefore P is normal. This distance-based classification allows detecting known kinds of anomalies, i.e. anomalous traffic with similar characteristics as in the training datasets.

Outlier detection. An outlier is an object that differs from most other objects significantly. Therefore it can be considered as an anomaly. For outlier detection, only the distance to the appropriate centroid of the normal cluster is calculated. If the distance between an object and the centroid is larger than a predefined threshold d_{max}, the object is treated as an outlier and anomaly. This is depicted in Figure 3 where P_2 and P_3 lie outside the d_{max} circle. In contrast to the classification method, outlier detection does not make use of the anomalous cluster centroid, i.e. it may be less accurate in detecting known kinds of anomalies. On the other hand, it allows detecting new anomalies that do not appear in the training datasets.

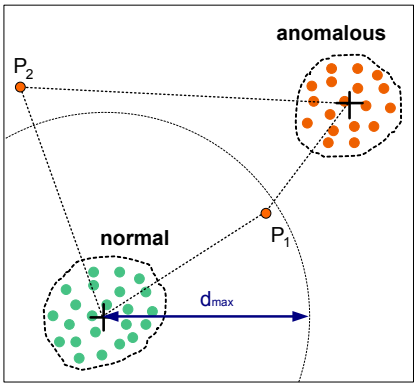

Fig. 4. Combined classification and outlier detection

Combined classification and outlier detection. Classification and outlier detection can be used in a combined way in order to overcome the limitations of each individual method. If the two methods are applied simultaneously, an object is treated as an anomaly if it is closer to the anomalous cluster centroid than to the normal one, or if its distance to the normal cluster centroid is larger than the predefined threshold. In Figure 4, for example, both objects P_1 and P_2 are regarded as anomalies. P_1 is closer to the anomalous cluster and P_2's distance to the normal group is larger than the threshold d_{max}.

In the following section, we present and discuss first experiments with the proposed clustering and anomaly detection approach.

IV. Experimental Results

We tested and evaluated our NDM approach with monitoring data from two different sources. First, we generated traffic in a local testbed. Secondly, we played back tcpdump traces recorded in a real network. Both times, we deployed

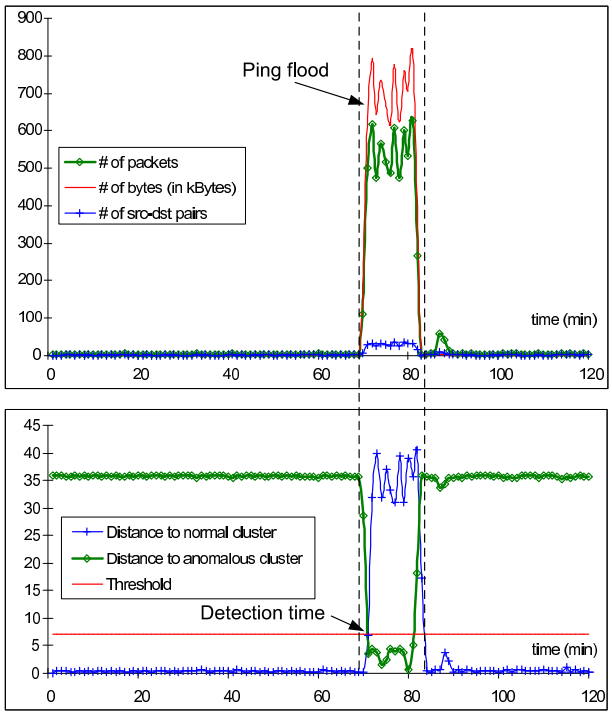

Fig. 5. Ping flood detection with generated traffic

the monitoring probe Vermont [19] to perform flow accounting and to store the resulting flow records in a database. In order to implement the data mining and anomaly detection algorithms, we extended the traffic analysis tool of the HISTORY (HIgh-Speed neTwork mOnitoRing and analYsis) project [20], [21].

These first experiments proof the general feasibility of the concept, yet they are not meant to provide quantitative results, such as false-positive and false-negative ratios. In subsection IV-C, we finally discuss the algorithmic complexity.

A. Evaluation with Generated Traffic

In a testbed environment, we evaluated the capability to detect port scans and DoS attacks. Therefore, we generated normal background traffic and attack traffic using multiple PCs and captured the traffic at the monitoring port of the interconnecting switch. As background traffic, we generated several TCP, UDP and ICMP flows of variable bit rate using the traffic generator *npag* [22]. This tool was also used to produce a ping flood and a UDP flood against port 53 (DNS). Additional port scans were performed with *nmap* [23].

Even though the generated traffic is not representative for real Internet traffic, it enables us to verify the basic capability to detect anomalous attack traffic. As an example, we illustrate the detection of an ICMP ping flood. The K-means clustering algorithm was applied to training data that included flow records of normal and attack traffic. Hence, the data mining process resulted in cluster centroids for normal and anomalous traffic.

Figure 5 shows the detection of a similar ICMP ping flood observed in a monitoring data set. The upper diagram depicts the three features *number of packets*, *number of bytes*, and *number of source-destination pairs* over time for the ICMP traffic class (interval length $T = 1$ min). As can be seen, the ping flood starts at $t = 70$ min. In the lower diagram, it is shown how the anomaly is detected: At $t = 71$ min the distance to the anomalous cluster falls below the distance to the normal cluster. The traffic is classified as normal again after the distance to the normal cluster becomes smaller than the distance to the anomalous cluster and at the same time smaller than the threshold $d_{max} = 7$ used for outlier detection ($t = 84$ min). Similarly, UDP floods as well as port scans could be detected.

B. Evaluation with Real Traces

In order to obtain first results on how the clustering method performs in real networks, we applied it to packet traces recorded at a gateway router that connects a student's residential network to the Internet. These anonymized traces are publicly available as tcpdump files from the traffic measurement data repository of the University of Twente [24], [25]. Each trace file contains a snippet of 15 minutes monitored at a 300Mbit/s Ethernet link.

We played back the trace file loc1-20020526-1115.bz2 at 10 percent of the recording speed using *tcpreplay* [26]. This allowed us to use standard PCs for monitoring and flow accounting without risking packet losses by the software monitoring processes. As a result of the reduced playback speed, the duration of the trace was stretched to 150 minutes. Again, the interval length was set to $T = 1$ min, corresponding to $T = 6$ sec at the original time scale.

A port analysis of the monitoring data revealed that more than 85 percent of both incoming and outgoing traffic was directed to unusual destination port numbers. This corresponds to the expectation that traffic of a student's residential network shows a large amount of traffic from peer-to-peer applications and online games. Nevertheless, we applied the K-means clustering algorithms to specific services such as HTTP, FTP and SSH. The resulting centroids of the normal and anomalous cluster were very close to each other, which indicates that there were no traffic anomalies in the analyzed data. Only the clusters for FTP control traffic (TCP port 21) can be clearly distinguished by a large difference in the number of packets and bytes (the difference in *cn* was small). However, we assume that both clusters represented normal behavior: The cluster with large number of packets and bytes can be explained by FTP clients scanning the directories on an FTP server.

A more interesting result was obtained by analyzing the UDP traffic of the default class. The K-means clustering resulted in the following two cluster centroids:

Cluster	pkts	bytes	src-dst
normal	28274	3288007	1896
anomalous	39725	3510792	14831

As can be seen, the average value of source-destination pairs in

the anomalous cluster is eight times higher than in the normal cluster. The corresponding anomalous traffic was detected in a time interval of about three minutes which corresponds to 15 to 20 seconds on the original time scale. More than 22.000 small UDP packets of 37 bytes were sent from a single IP address in the residential network to more than 20.000 different, mostly external IP addresses. The predominating destination port was 27015 (more than 14.000 packets), the main source port was 1830, followed by 1831, 1832, and 1833. The majority of the transmitted packets resulted in a small reply packet (34 bytes) sent back to the IP address in the residential network. According to some information found on the Web, UDP port 27015 is used by servers of the online game *half-life*. Hence, this traffic anomaly was probably caused by the client program of this game updating its list of available game servers in the Internet.

C. Complexity Estimation

The complexity of the K-means clustering algorithm is $O(K\,n\,t)$ where K is the number of clusters, n the number of objects to be classified, and t the number of iterations which depends on the initial classification of the objects and the feature value distribution (typically, $t << n$). We apply the K-means clustering algorithm with $K = 2$ to the feature datasets of different predefined services separately. If m is the number of time intervals T in the training data, for each traffic class up to m feature datasets may occur. In case that we consider L specific services, we get $O\,(K\,(L+3)m\,t)$ (we have to add 3 for the default traffic classes for TCP, UDP, and ICMP). As a comparison: The LOF clustering algorithm deployed in [7] is $O(n^2)$ complex.

More important than the complexity of the data mining process is the detection complexity that is required to classify new monitoring data as normal or anomalous. This is because a low and constant detection complexity is essential for a scalable real-time detection mechanism. Our distance-based detection requires the transformation of new monitoring data into the feature space. Thereby, the flow records occurring in one time interval T are converted into one feature dataset for each service. The corresponding distances to the normal and anomalous cluster centroids have to be calculated and compared to each other as well as to the outlier detection threshold. Applying the detection to monitoring data of one time interval T is thus $O\,(K(L+3))$ complex.

These estimations show that the complexity of the data mining process is acceptable and that the detection complexity only depends on the number of clusters K and the number of separately considered services L. In order to keep L at a low level without losing the advantage of having distinct profiles for different services, services with similar traffic behavior could be analyzed as a group.

V. Conclusion

In this paper, we presented a novel *Network Data Mining* approach that applies the K-means clustering algorithm to feature datasets extracted from flow records. Training data are divided into clusters of time intervals of normal and anomalous traffic. While the data mining process is relatively complex, the resulting cluster centroids can be used to detect anomalies in new on-line monitoring data with a small number of distance calculations. This allows deploying the detection method for scalable real-time detection, e.g. as part of a intrusion detection system. Applying the clustering algorithm separately for different services (identified by their transport protocol and port number) improves the detection quality. We presented and discussed the results of first experiments using generated and real traffic.

We are currently working on several improvements of the presented approach, thus comparing clustering results achieved with different K in order to determine the optimum number of clusters, considering additional features such as the average flow duration, and considering different distance metrics, such as the Mahalanobis distance. Finally, we are going to evaluate the detection performance using reference data such as DARPA'98 traces [27] as well as real Internet traffic.

References

[1] KDD Cup 1999 Data, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html, Oct. 1999.

[2] B. Claise, G. Sadasivan, V. Valluri, and M. Djernaes, "Cisco Systems NetFlow Services Export Version 9," RFC 3954 (Informational), Oct. 2004.

[3] B. Claise, S. Bryant, G. Sadasivan, S. Leinen, and T. Dietz, "IPFIX Protocol Specifications," Internet-Draft, work in progress, draft-ietf-ipfix-protocol-24, Nov. 2006.

[4] U. M. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From data mining to knowledge discovery in databases." *AI Magazine*, vol. 17, no. 3, pp. 37–54, 1996.

[5] W. Lee and S. Stolfo, "Data mining approaches for intrusion detection," in *Proceedings of the 7th USENIX Security Symposium*, San Antonio, TX, 1998.

[6] P. Dokas, L. Ertoz, V. Kumar, A. Lazarevic, J. Srivastava, and P. N. Tan, "Data mining for network intrusion detection," in *Proceedings of the NSF Workshop on Next Generation Data Mining*, Nov. 2002.

[7] L. Ertoz, E. Eilertson, A. Lazarevic, P.-N. Tan, J. Srivastava, V. Kumar, and P. Dokas, *Next Generation Data Mining*. MIT Press, 2004, ch. 3: The MINDS - Minnesota Intrusion Detection System.

[8] M. Esposito, C. Mazzariello, F. Oliviero, S. P. Romano, and C. Sansone, "Evaluating pattern recognition techniques in intrusion detection systems." in *Proceedings of the 5th International Workshop on Pattern Recognition in Information Systems (PRIS) 2005*, May 2005, pp. 144–153.

[9] D. Barbará, J. C. S. Jajodia, L. Popyack, and N. Wu, "ADAM: Detecting intrusions by data mining," in *Proceedings of the IEEE Workshop on Information Assurance and Security*, Jun. 2001, pp. 11–16.

[10] J. Luo and S. Bridges, "Mining fuzzy association rules and fuzzy frequency episodes for intrusion detection," *International Journal of Intelligent Systems*, vol. 15, no. 8, pp. 687–704, 2000.

[11] Libpcap/Tcpdump Homepage, http://www.tcpdump.org, 2007.

[12] W. W. Cohen, "Fast effective rule induction," in *Proceedings of the 12th International Conference on Machine Learning*, A. Prieditis and S. Russell, Eds., Tahoe City, CA, Jul. 1995, pp. 115–123.

[13] R. Agarwal and M. V. Joshi, "PNrule: A new framework for learning classifier models in data mining," Department of Computer Science, University of Minnesota, Tech. Rep. 00-015, 2000.

[14] A. Lazarevic, L. Ertöz, V. Kumar, A. Ozgur, and J. Srivastava, "A comparative study of anomaly detection schemes in network intrusion detection," in *Proceedings of the Third SIAM International Conference on Data Mining*, May 2003.

[15] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying density-based local outliers," in *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, May 2000, pp. 93–104.

[16] S. M. Bridges and R. M. Vaughn, "Fuzzy data mining and genetic algorithms applied to intrusion detection," in *Proceedings of the Twenty-third National Information Systems Security Conference*, Oct. 2000.
[17] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability*. University of California Press, 1967, pp. 281–297.
[18] "Introduction to Cisco IOS NetFlow - A Technical Overview," Cisco Systems, Inc., Tech. Rep., Feb. 2006. [Online]. Available: http://www.cisco.com
[19] R. T. Lampert, C. Sommer, G. Münz, and F. Dressler, "Vermont - A Versatile Monitoring Toolkit for IPFIX and PSAMP," in *Proc. of IEEE/IST Workshop on Monitoring, Attack Detection and Mitigation (MonAM 2006)*, Tuebingen, Germany, Sep. 2006.
[20] F. Dressler and G. Carle, "History - high speed network monitoring and analysis," in *Proceedings of the 24th IEEE Conference on Computer Communications (IEEE INFOCOM 2005), Poster Session*, Mar. 2005.
[21] History Project Homepage, http://www.history-project.net, 2007.
[22] F. Dressler, "Policy-based traffic generation for IP-based networks," in *25th IEEE Conference on Computer Communications (IEEE INFOCOM 2006), Poster Session*, Apr. 2006.
[23] Nmap Network Scanner Homepage, http://insecure.org/nmap/, 2007.
[24] R. van de Meent, "M2C Measurement Data Repository," University of Twente, Enschede, The Netherlands, M2C Deliverable D1.5, Dec. 2003.
[25] University of Twente - Traffic Measurement Data Repository, http://traffic-repository.ewi.utwente.nl, 2007.
[26] Tcpreplay Homepage, http://tcpreplay.synfin.net/trac/, 2007.
[27] R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. R. Kendall, D. McClung, D. Weber, S. E. Webster, D. Wyschogrod, R. K. Cunningham, and M. A. Zissman, "Evaluating Intrusion Detection Systems: the 1998 DARPA Off-Line Intrusion Detection Evaluation," in *Proceedings of the 2000 DARPA Information Survivability Conference and Exposition*, 2000.