Author(s): Sipola, Tuomo; Juvonen, Antti; Lehtonen, Joel

Title: Dimensionality reduction framework for detecting anomalies from network logs

Year: 2012

Version:

Please cite the original version:

Sipola, T., Juvonen, A., & Lehtonen, J. (2012). Dimensionality reduction framework for detecting anomalies from network logs. Engineering Intelligent Systems, 20(1/2), 87-97.

Dimensionality Reduction Framework for Detecting Anomalies from Network Logs

Tuomo Sipola Antti Juvonen Joel Lehtonen*
tuomo.sipola@jyu.fi antti.k.a.juvonen@jyu.fi joel.lehtonen@iki.fi

Department of Mathematical Information Technology
University of Jyväskylä
Jyväskylä, Finland

Abstract

Dynamic web services are vulnerable to a multitude of intrusions that could be previously unknown. Server logs contain vast amounts of information about network traffic, and finding attacks from these logs improves the security of the services. In this research features are extracted from HTTP query parameters using 2-grams. We propose a framework that uses dimensionality reduction and clustering to identify anomalous behavior. The framework detects intrusions from log data gathered from a real network service. This approach is adaptive, works on the application layer and reduces the number of log lines that needs to be inspected. Furthermore, the traffic can be visualized.

Keywords: intrusion detection, anomaly detection, n-grams, diffusion map, data mining, machine learning

1 Introduction

The goal of this paper is to present an adaptive way to detect security attacks from network log data. All networks and systems can be vulnerable to different types of intrusions. Such attacks can exploit e.g. legitimate features, misconfigurations, programming mistakes or buffer overflows [1]. This is why *intrusion detection systems* are needed. An intrusion detection system gathers data from the network, stores these data to log files and analyzes them to find malicious or anomalous traffic [2]. Systems can be vulnerable to previously unknown attacks, commonly known as zero-day attacks [3]. Because usually these attacks differ from the normal network traffic, they can be found using anomaly detection [4].

In modern networks clients request and send information using queries. In HTTP traffic these queries are strings containing arguments and values. It is easy to manipulate such queries to include malicious attacks. These injection attacks try to create requests that corrupt the server or collect confidential information [5]. Therefore, it is important to analyze the collected data in log files. Most intrusion detection systems

*Now with C2 SmartLight Ltd.

 The original print version appeared as: Sipola, T., Juvonen, A., Lehtonen, J. (2012). Dimensionality Reduction Framework for Detecting Anomalies from Network Logs. *Engineering Intelligent Systems* 20(1), 87–97.

analyze TCP packet data. There are not many application layer IDS systems available. Because the HTTP log data are very different from network packet data, they both need to be analyzed. Different attacks can be performed on different layers.

An *anomaly* is a pattern in data that is different from the well defined normal data [4]. In network data, this usually means an intrusion. There are two main approaches for detecting intrusions from network data: *misuse detection* and *anomaly detection* [2]. Misuse detection means using predefined attack signatures to detect the attacks, which is usually accurate but detecting new types of attacks is not possible. In anomaly detection the goal is to find actions that somehow deviate from normal traffic. This way it is possible to detect previously unknown attacks. However, not all anomalous traffic is intrusive. This means there might be more false alarms. Different kinds of machine learning based methods, such as self-organizing maps and support vector machines, have been used in anomaly detection [6, 7]. Information about other anomaly detection methods can be found in the literature [2]. *Unsupervised anomaly detection* techniques are most usable in this case, because no normal training data are available [4]. These techniques work without prior knowledge of attack patterns. This kind of adaptive framework is suitable for *a posteriori* network log analysis.

This study takes the approach of dimensionality reduction. Because the number of dimensions of the feature space grows large and sparse when analyzing textual information, such as log files, this is one of the most feasible techniques. Furthermore, the sparsity of data suggests about the underlying low dimensional structure. Almost the same amount of information can be represented with lower number of dimensions. Diffusion map is a manifold learning method that maps high-dimensional data to a low-dimensional diffusion space [8]. It provides tools for visualization and clustering [9]. The basic idea behind any manifold learning method is the eigendecomposition of a similarity matrix. By unfolding the manifold it reveals the underlying structure of the data that is originally embedded in the high-dimensional space [10]. Diffusion maps have been applied to various data mining problems. These include vehicle classification by sound [11], music tonality [12], sensor fusion [13], radio network problem detection [14, 15] and detection of injection attacks [16]. In addition to the advantage of reduced number of dimensions, the approach can be used for unsupervised learning [4].

2 Related research

Kruegel and Vigna [17] analyzed the parameter values of HTTP queries. The static queries with no parameters were removed. The underlying assumption is that attack patterns differ from normal traffic and that this difference can be expressed quantitatively. They used several different analyzing methods, such as attribute length and character distribution. The learning was based on previous data. The data were not labeled. The analysis of character distribution is similar to our research, because essentially the characters are n-grams with the length 1. We use 2-grams for higher detection rates, but we will also get more dimensions in the data matrices.

Hubballi et al. [18] used layered higher order n-grams for detecting intrusions. However, this analysis was not done on application layer data, but on the network packet payloads. Higher order n-grams are n-grams where $n > 2$. This means that the method is computationally more expensive, but rare events might be detected more accurately. The n-grams are organized into bins based on their frequency. The analysis starts with 1-grams, and it moves to higher n-grams incrementally to get higher accuracy. In the research the number of distinct and unique n-grams went up almost

linearly as n increased. Therefore, using higher order n-grams might not be as complex in practice as it could be theoretically. For example, the theoretical maximum number for 3-grams in ascii-characters is 256^3, which is considerably higher than the case with 2-grams.

Dimensionality reduction has been discussed in the context of anomaly detection from networks. Ringberg et al. studied the IP packet data and tried to detect anomalies using principal component analysis. They also identified the main challenges when using principal component dimensionality reduction approach. The finding about large anomalies contaminating the subspace is relevant also to our research. However, their network architecture is more complex than ours [19]. Callegari et al. analyzed similar packet data [20]. These studies used low-level IP packet datasets that need specific aggregation before they can be processed. Our research concerns the application level log data, which is text, while the IP packet datasets are numeric. In addition, we compare the results of principal component analysis and diffusion maps.

Diffusion maps have been applied in the network security context. David explored the use of diffusion map methods to find injection attacks in hyper-networks. His data included SQL injection examples that used a similar feature extraction as our research. The n-gram feature extraction was applied to tokenized SQL [16]. Our research, in contrast, focuses on the raw textual queries. Furthermore, David and Averbuch used a localized diffusion folder approach to classify network protocols, among other examples. Their data contains low-level features such as duration and the number of bytes [21]. However, our data comes from the application layer of the network, specifically web server logs. These are different from the low-level network features and contain lots of textual information in the form of queries. Moreover, we use the theoretical framework of spectral clustering as the basis of our research.

3 Methodology

Straightforward numerical methods are difficult to apply to textual data such as log files. Therefore, log data must be transformed into feature space. This mapping of textual information to numerical matrix enables mathematical analysis of the original log lines.

However, this leads to a large number of dimensions in the feature space. For efficient analysis, classification and visualization the number of dimensions must be reduced. This gives the opportunity to use a multitude of classification algorithms.

The proposed method consists of the following steps:

1. Removing lines that do not contain parameters.
2. Feature extraction from the log line using 2-grams.
3. Dimensionality reduction of the features.
4. Classifying the lines either as normal or attack.

After these steps the log file can be visualized as a figure where the attacks are more easily seen than from a text file. Furthermore, the suspected attack lines can be inspected in more depth. This facilitates finding abnormal activities because only these suspected lines are inspected, instead of thousands in the original log.

3.1 Feature extraction

The features include 2-grams from HTTP query parameters. The log files are simple text files where each line represents one query sent from the client to the server. Extracting the true intention of the query is challenging, and the text needs to be converted to a more machine-friendly format. The feature extraction essentially means converting this textual data into numerical matrices.

First let us define an *n*-gram as a consecutive sequence of *n* characters [22]. *N*-gram is a substring with length of *n*. For example, the string *ababc* contains unique 2-grams *ab*, *ba* and *bc*. The 2-gram *ab* appears twice, thus having frequency of 2. A list of tokens of text can be represented with a vector consisting of *n*-gram frequencies [22]. Feature vector describing this string would be $x_{ababc} = [2,\ 1,\ 1]$. The only features extracted are *n*-gram frequencies. Furthermore, syntactic features of the input strings might reveal the differences between normal and anomalous behavior. Computed *n*-grams can extract features that describe these differences. It is assumed that an anomalous query contains some text in the parameter part that differs from normal behaviour. This means that it must contain some *n*-grams that appear rarely in the data.

Here is an example of constructing the feature matrix using the *n*-gram analysis process with two words, *anomaly* and *analysis*. From these words we get the unique 2-grams `an`, `no`, `om`, `ma`, `al`, `ly`, `na`, `ys`, `si` and `is`. From this information we can construct a matrix with the *n*-gram frequencies.

an	no	om	ma	al	ly	na	ys	si	is
1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1

The feature matrix is constructed in this way, including all the different strings that appear in the parameter fields on each query. Each log line corresponds to a row in the matrix. From this we can see that there are 10 unique 2-grams in this example. The logs are ascii-coded, so they can contain 256 different characters. The theoretical maximum number for unique 2-grams using ascii-characters is 256^2, but in practice we did not get even near to that number. However, in a very varied and big dataset the number of dimensions could get very high.

The frequencies are collected to a feature matrix X. These *n*-gram frequencies are key-value fields, variable-length by definition. Key strings are ignored and 2-grams are produced from each parameter value. The count of occurrences of every occurring 2-gram is summed. In practice *n*-gram tables produced from real life data are very sparse, containing columns in which there are only zero occurrences. To minimize the number of columns, the processing is done in two passes, first determining the number of unique *n*-grams and then analyzing the frequencies. If a column contains no variation between entries, that column is not present in the final numeric matrix X. Therefore, only the columns that actually contain some useful information about the features are included in the analysis.

With this preprocessing technique it is possible to use *n*-grams whose value of *n* is higher than 2. However, the number of unique *n*-grams will increase and therefore the number of dimensions will increase as well.

3.2 Dimensionality reduction

The number of extracted features is so large that dimensionality reduction is performed using principal component analysis and diffusion map. Diffusion map is a manifold

learning method that embeds the original high-dimensional space into a low-dimensional diffusion space. Anomaly detection and clustering are easier in this embedded space [9].

The recorded data describe the behavior of the system. Let this data be $X = \{x_1, \ldots, x_N\}$, $x_i \in \mathbb{R}^n$. Here N is the number of samples and n the dimension of the original data. In practice the data are in a $N \times n$ matrix with features as columns and each sample as rows.

3.2.1 Principal component analysis

Principal component analysis (PCA) tries to extract orthogonal components maximizing their variance from the data. This simplifies the representation of the information within the data and also facilitates the analysis of the structure and features in the data. The principal components are linear combinations of the original features. The first principal component contains the largest amount of variance. PCA reveals the most information in terms of variance, but this does not necessarily mean that it separates different clusters in an optimal way [23, 24, 25].

PCA performs the eigendecomposition on the covariance matrix C of the centered data matrix X_c. The decomposition $C = U\Lambda U^*$ gives the eigenvectors in U that map the points in X to a low-dimensional space. This mapping can be calculated with $X_{PCA} = XU$. Another approach is to take the singular value decomposition (SVD) of the original matrix X. One way to interpret this is as rotation of axes to find the most important features. The new principal components are in the direction of most variance in the data and thus represent the most differentiating combination of features [23, 24, 25].

As with many dimensionality reduction methods using eigendecompositions, the number of selected components becomes a problem. One way to do this is to seek for the eigengap, i.e. a big change of eigenvalues. This way the eigenvalues reveal the principal components that cover most of the variance [23, 24, 25].

PCA is a linear method and has difficulties finding nonlinear dependencies between features. It has initial assumptions that restrict its use for latent variable separation and nonlinear dimensionality reduction [25].

3.2.2 Diffusion map

At first, an affinity matrix W is constructed. This calculation takes most of the computation time. The matrix describes the distances between the points. This study uses the common Gaussian kernel with Euclidean distance measure, as in equation 1 [9, 26].

$$W_{ij} = \exp\left(-\frac{||x_i - x_j||^2}{\varepsilon}\right) \tag{1}$$

The affinity neighborhood is defined by ε. Choosing the parameter ε is not trivial. It should be large enough to cover the local neighborhood but small so that it does not cover too much of it [11].

The rows of the affinity matrix are normalized using the diagonal matrix D, which contains the row sums of the matrix W on its diagonal.

$$D_{ii} = \sum_{j=1}^{N} W_{ij} \tag{2}$$

P expresses normalization that represents the probability of transforming from one state to another. Now the sum of each row is 1.

$$P = D^{-1}W \tag{3}$$

Next we need to obtain the eigenvalues of this transition probability matrix. The eigenvalues of P are the same with the conjugate matrix in equation 4. The eigenvectors of P can be derived from $\tilde{P}$ as shown later.

$$\tilde{P} = D^{\frac{1}{2}}PD^{-\frac{1}{2}} \tag{4}$$

If we substitute the P in equation 4 with the one in equation 3, we get the symmetric probability matrix $\tilde{P}$ in equation 5. It is called the normalized graph Laplacian [27] and it preserves the eigenvalues [26].

$$\tilde{P} = D^{-\frac{1}{2}}WD^{-\frac{1}{2}} \tag{5}$$

This symmetric matrix is then decomposed with singular value decomposition (SVD). Because $\tilde{P}$ is a normal matrix, spectral theorem states that such a matrix is decomposed with SVD: $\tilde{P} = U\Lambda U^*$. The singular values of this symmetric square matrix equal to its eigenvalues, which lie on the diagonal of $\Lambda = \text{diag}([\lambda_1, \lambda_2, \ldots, \lambda_N])$. Matrix $U = [u_1, u_2, \ldots, u_N]$ contains in its columns the N eigenvectors u_k of $\tilde{P}$. Furthermore, because $\tilde{P}$ is conjugate with P, these two matrices share their eigenvalues. However, to calculate the right eigenvectors v_k of P, we use equation 6 and get them in the columns of $V = [v_1, v_2, \ldots, v_N]$ [26].

$$V = D^{-\frac{1}{2}}U \tag{6}$$

The coordinates of a data point in the embedded space using eigenvalues in Λ and eigenvectors in V are in the matrix Ψ in equation 7. The rows correspond to the samples and the columns to the new embedded coordinates [9].

$$\Psi = V\Lambda \tag{7}$$

Strictly speaking, the eigenvalues should be raised to the power of t. This scale parameter t tells how many time steps are being considered when moving from data point to another. Here we have set it $t = 1$ [9].

With suitable ε the decay of the spectrum is fast. Only d components are needed for the diffusion map for sufficient accuracy. It should be noted that the first eigenvector v_1 is constant and is left out. Using only the next d components the diffusion map for original data point x_i is presented in equation 8. Here $v_k(x_i)$ corresponds to the ith element of kth eigenvector [9].

$$\Psi_d : x_i \rightarrow [\lambda_2 v_2(x_i), \lambda_3 v_3(x_i), \ldots, \lambda_{d+1} v_{d+1}(x_i)] \tag{8}$$

This diffusion map embeds the known point x_i to a d-dimensional space. Dimension of the data are reduced from n to d. If desired, the diffusion map may be scaled by dividing the coordinates with λ_1.

3.3 Anomaly detection

After obtaining the low-dimensional presentation of the data it is easier to cluster the samples. Because spectral methods reveal the manifold, this clustering is called spectral clustering. This method reveals the normal and anomalous samples [28]. Alternatively, k-means or any other clustering method in the low-dimensional space is also possible [29]. Another approach is the density-based method [14].

Only the first few low-dimensional coordinates are interesting. They contain most of the information about the manifold structure. For diffusion map we use only the dimension corresponding to second eigenvector to determine the anomality of the samples. At 0, this dimension is divided into two clusters. The cluster with more samples is considered normal behavior. Conversely, the points in the other cluster are considered anomalous [30, 31, 28]. The second eigenvector acts as the separating feature for the two clusters in the low-dimensional space. The second eigenvalue is the solution to the normalized cut problem, which finds small weights between clusters but strong internal ties. This spectral clustering has probabilistic interpretation: grouping happens through similarity of transition probabilities between clusters [30, 32]. For PCA we use the first principal component in a similar way.

In practice the border between the normal and anomalous behavior might be unclear. This is the case especially with unsupervised learning, or when exploring the data for the first time. The normal cluster is usually very dense, and most of the data points lie within that cluster. The other points can be interpreted as deviating from the normal state, and thus anomalous.

4 Case 1: Validation with labeled data

4.1 Data acquisition

The data are acquired from a large real life web service. Let us call this dataset "A". This case has been presented in an earlier publication [33]. The log files contain mostly normal traffic, but they also include anomalities and actual intrusions. The log files are from several Apache servers and are stored in *combined log format*. Listing below provides an example of a single log line. It includes information about the user's IP address, time and timezone, the HTTP request including used resource and parameters, Apache server response code, amount of data sent to the user, the web page that was requested and used browser software.

```
127.0.0.1 - - [01/January/2011:00:00:01 +0300]
"GET /resource?parameter1=value1&parameter2=value2 HTTP/1.1"
200 2680 "http://www.address.com/webpage.html"
"Mozilla/5.0 (SymbianOS/9.2;...)"
```

The access log of a web site contains entries from multiple, distinct URLs. Most of them point to static requests like images, CSS files, etc. We do not focus on finding anomalies from those requests because it is not possible to inject code via static requests unless there are major deficiencies in the HTTP server itself. Instead, we focus on finding anomalies from dynamic requests because those requests are handled by the web application, which is run behind the HTTP server.

To reach this goal, the access log entries are grouped by the resource URL. That is the part between host name and parameters in the HTTP URL scheme. Resources

containing only HTTP GET requests with no parameters are ignored. Each remaining resource is converted to a separate numerical matrix. In this matrix, a row represents a single access log entry, and a column represents an extracted feature.

Feature extraction is done in two passes. In the first pass the number of features is determined, and in the second pass the resulting matrix is produced. In our study we extracted the number of occurrences of 2-grams produced from HTTP GET parameters. In the example above, the parameter values form a string `value1value2`. This string is then analyzed for 2-gram frequencies. These frequencies are normalized with logarithm in order to scale them. This ensures that the distances between the samples are comparable.

4.2 Data analysis

To measure the effectiveness of the method the data are labeled so that classification accuracy can be measured. However, this labeling is not used for training the diffusion map. The class labels are not input for the method.

Diffusion map reveals the structure of the data, and all the anomalies are detected. The n-gram features of the data are mapped to lower dimensions. Figure 1 shows the resulting low-dimensional diffusion space with $\varepsilon = 100$. The normal behavior (N=2999) lies in the dense area to the upper left corner. Anomalous points (N=1293) are to the right of 0.

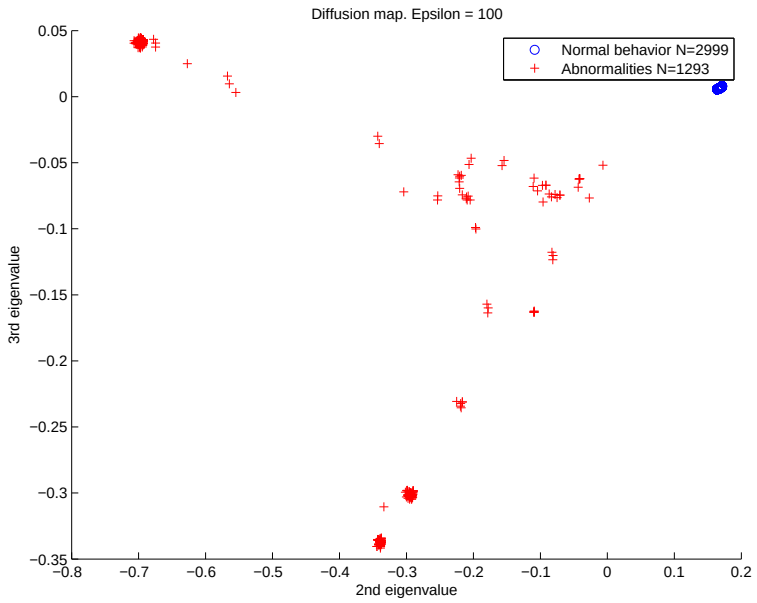

Figure 1: Two-dimensional diffusion map of the dataset A.

Figure 2 shows that the eigenvalues converge rapidly with $\varepsilon = 100$. This means that the first few eigenvalues and eigenvectors cover most of the differences observed in the data. The first value is 1 and corresponds to the constant eigenvector that is left out in the analysis. Eigenvalues $\lambda_2 = 0.331$ and $\lambda_3 = 0.065$ cover large portions of the data when compared to the rest that have values below 0.005.

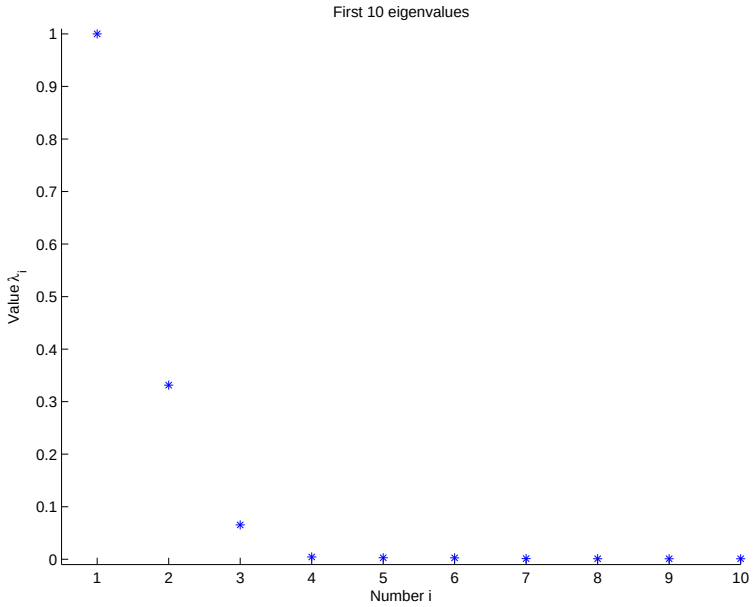

Figure 2: Eigenvalues of transition matrix with $\varepsilon = 100$ (dataset A).

Classification is tested with different values of ε, which defines the neighborhood for diffusion map. Accuracy of classification is defined as $accuracy = (tp+tn)/(tp+fp+fn+tn)$. Figure 3 shows how the accuracy of classification changes when ε is changed. Higher values of ε result in better accuracy. Precision of classification is defined $precision = tp/(tp+fp)$. The precision stays at 1 once any anomalies are detected, which means that all the anomalies detected are real anomalies regardless of the accuracy [23, p. 361].

For comparison, principal component analysis (PCA) is performed on the same normalized feature matrix [23, p. 79]. Results are very similar to the diffusion map approach, because of the simple structure of the feature matrix. This suggests that data points are linearly dependent. Furthermore, PCA reaches the same accuracy and precision as diffusion map. The low-dimensional presentation is also very similar. Figure 4 shows the first two coordinates of PCA.

5 Case 2: Unknown data

5.1 Data acquisition

After testing the methods with known data, we now analyze data that is totally unknown. We call this dataset "B". This is the realistic situation with the web service that we are trying to analyze. There is no previous information about any attacks or other anomalies. The goal is to find a small amount of interesting lines that can then be analyzed more accurately. The number of log lines is so big that it is impossible to check all the lines manually. This is why anomaly detection is needed.

We start with relatively new dataset that has about 10 million lines. However, the lines with no parameters in the HTTP queries can be filtered out, because they are

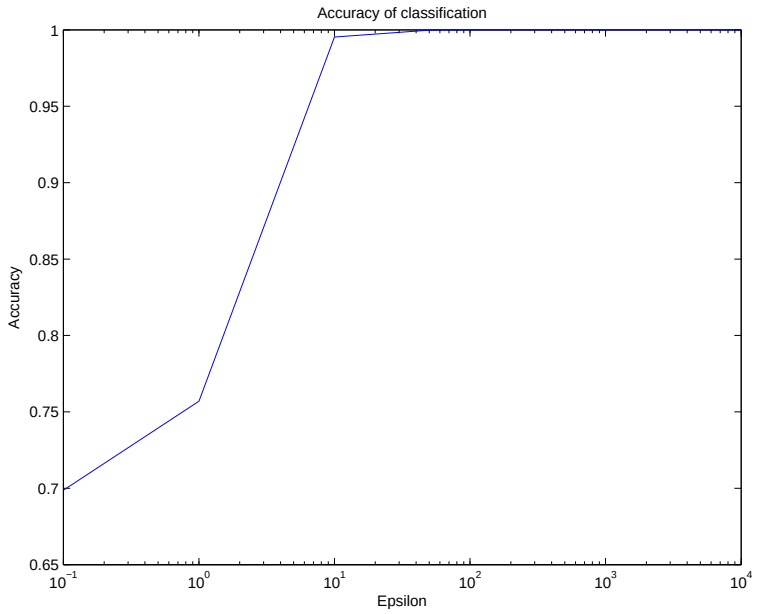

Figure 3: Accuracy of classification changes when the parameter ε is changed (dataset A).

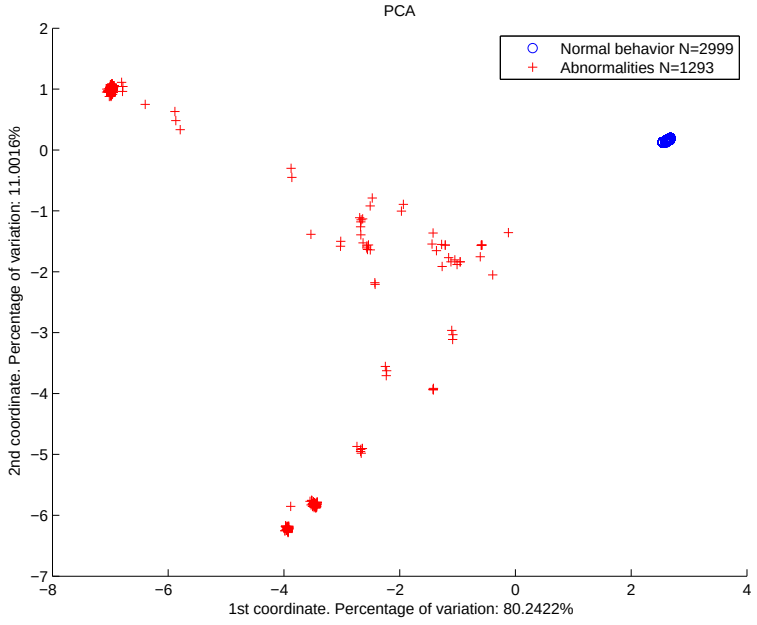

Figure 4: PCA of dataset A, first two coordinates.

not a big security risk. This leaves us with 2.5 million lines. These lines are then divided into different files based on the HTTP request URI. The entire log file cannot be analyzed at once, because different resources have very different parameters. The normal parameters for one single URI are usually quite similar, however. It makes more sense trying to analyze them individually. The number of different resources is quite high, more than 80 000, but many of the resources include just a few lines. In this case, it is sufficient to analyze some of the most frequently used resources. In addition to finding anomalies, this will give us more information about the web service traffic in general. The traffic can also be visualized.

After preprocessing the number of unique 2-grams in the most used resource URI was more than 1100. This means that dimensionality reduction is definitely needed, but the number of dimensions is not close to the theoretical maximum of 256^2 2-grams.

The new log file acquired for this case is in a different format than the log file analyzed in the first case. Some additional information, such as time in UNIX time, is also included. However, this information is not used in this analysis. The features extracted are the same as in the previous case. The feature matrix is calculated only from the parameter values. In this example the string to be analyzed would be `value`.

```
1305167880 111.222.111.222 965 633 29112
GET /path/to/resource.png?parameter=value HTTP/1.1
/path/to/resource.png parameter=value
/full/path/to/resource.png 200 + -
image/png,image/*;q=0.8,*/*;q=0.5 GB2312,utf-8;q=0.7,*;q=0.7
gzip,deflate fi-fi,fi;q=0.5 http://example.com Mozilla/5.0
(Windows; U; Windows NT 5.1; fi-FI; rv:1.9.2.15)
Gecko/20110101 Firefox/3.6.15
```

Feature matrix is constructed and logarithmic scaling applied in the same way as presented earlier with the dataset A. Even though the number of lines in the log file is quite high, the preprocessing phase takes only less than 10 minutes. Everything is written into temporary files to save memory. If more memory is available, the preprocessing could be changed to use it and it would get faster.

5.2 Data analysis

We choose two commonly used resources for analysis from the whole log data B. These resources are called "B1" and "B2". We aim to find possible intrusion attempts from them. Dimensionality reduction is performed with both PCA and diffusion map. The results are then compared. Choosing ε for diffusion map is done differently than for dataset A. The sum $L = \sum_{i,j} W_{ij}$ plotted using logarithmic scale reveals the desirable linear region for ε [34, 35]. The value is chosen from that range, however, because even small changes of ε in that area affect the resulting embedding drastically, some human discretion must be used. Classification is done using spectral clustering.

Dataset B1 turns out to be a simple case where most of the data points are similar. The few deviations are easy to find from the feature matrix. First B1 is analyzed using PCA. Figure 5 shows that most of the normal behavior (N=14206) is concentrated to a very dense cluster. Our classifier assumes the points (N=87) to the right of the normal cluster to be anomalous. This clustering is feasible because the log lines contain actual previously unknown intrusions, although not all anomalies are intrusive. The anomalous points also seem to form clusters. These could indicate different types of

attacks that happen frequently. This information could be used to further protect the service in the future.

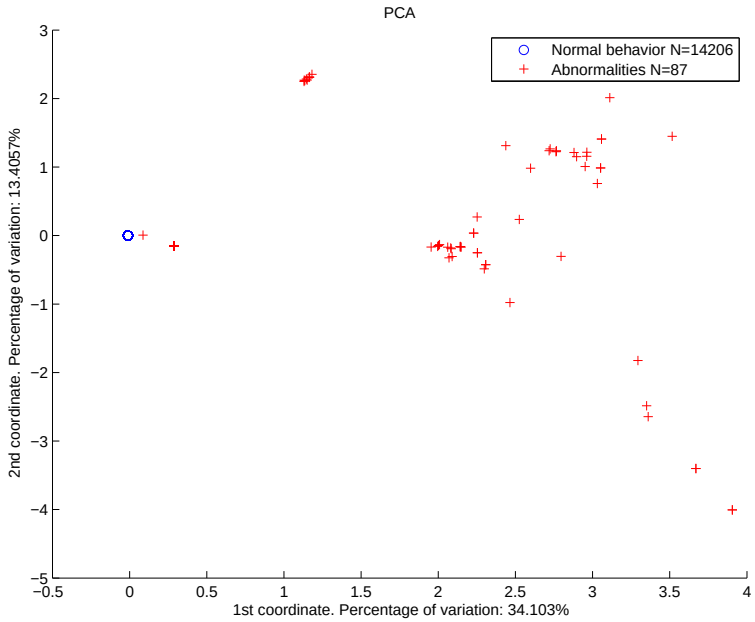

Figure 5: PCA of dataset B1.

Dataset B1 is also analyzed with diffusion map. The classification results are similar to PCA, even though the Figure 6 looks different. The normal behavior (N=14216) is concentrated leaving the anomalies (N=77) to the right of zero based on the first coordinate. The most differing anomalies are very far from the normal cluster. Some anomalies are very close to the normal data points. This means that the border between anomalous and normal traffic is not very clear. For this reason, 10 intrusion attempts that PCA detected were not discovered by diffusion map. This explains the difference in the number of found anomalies. Finding an optimal ε value would improve the result. However, this is a difficult task because of the unsupervised approach. Even though the low dimensional picture of PCA does not look as clear as the diffusion map, the result for PCA is better due to 10 false negatives that diffusion map fails to detect.

Dataset B2 contains more difficult and complex queries. This set is an example where the low dimensional representation by PCA and diffusion map are clearly different. Figure 7 shows the PCA of this dataset. The structure of the dataset is seen from the figure but the exact location of anomalies is difficult to find. This is because even the normal query lines include long and dynamically changing strings. The sparse left side is actually normal traffic, but there seems to be a lot of variation in the normal traffic alone. The anomalies found by diffusion map are situated in the upper right corner of the PCA representation. Data points do not form a distinct cluster, making anomaly detection and clustering very difficult with this representation. The used simple spectral clustering clearly does not work in this case. Further clustering with more advanced algorithms might reveal what types of queries the log file contains. Most variance is captured by the first principal component. However, two first principal components do not contain most of the total cumulative variance. Even this kind of visualization

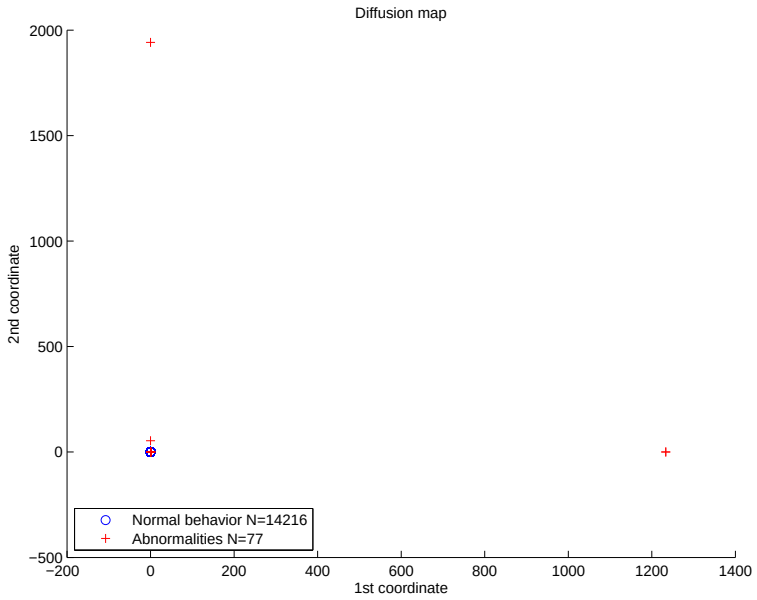

Figure 6: Diffusion map of dataset B1, $\varepsilon = 7$.

facilitates the analysis of huge text files (N=21406).

Diffusion map finds anomalies from dataset B2. The first two coordinates capture almost all of the difference between normal and anomalous queries (Figure 8). In addition, the clusters are very clearly separated and the normal traffic is easy to distinguish. This dataset shows a clear difference between PCA and diffusion map results. The anomalous cluster (N=173) contains the points on the far left and the anomalous points near the normal cluster. Again, the normal cluster (N=21233) is very dense. The found anomalies contain 88 real intrusions. The intrusions are related to injecting malicious SQL queries or scripts into the HTTP query. Some non-intrusive queries are also included, but they can be manually screened afterwards. The number of log lines is small enough so that system administrator can inspect the anomalous lines and easily find the intrusion attempts. Anomaly detection seems to find attacks from a large and varying dataset. The anomalous traffic forms two distinct clusters, one of which contains the intrusions. Diffusion map with a correctly selected ε helps in finding anomalies and automatically detecting normal cluster. Larger values of ε make the diffusion map behave more like PCA. These approachees are more suitable for visual inspection and multicluster analysis.

6 Conclusion

The goal of this study is to find security attacks from network data. The proposed anomaly detection scheme could be used for query log analysis in real life situations. We concentrate on web server log data, which contains text queries that are the focus of our analysis. In these kinds of practical situations the boundary between normal traffic and intrusions is not always very clear. However, the relative strangeness of the sample could indicate how severe an alert is.

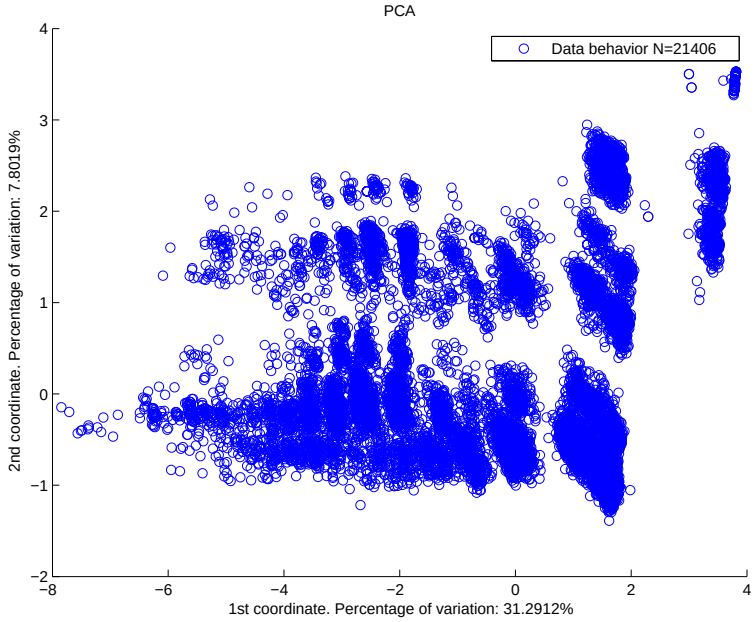

Figure 7: PCA of dataset B2.

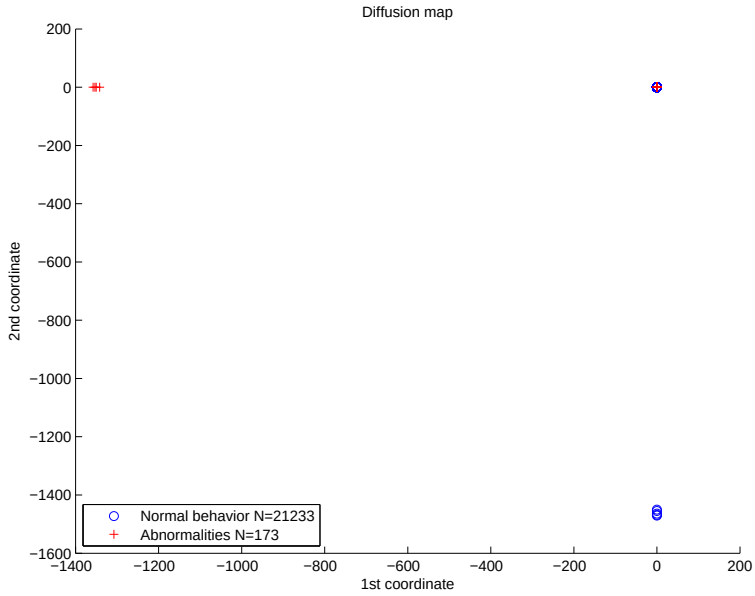

Figure 8: Diffusion map of dataset B2, $\varepsilon = 7$.

The dimensionality reduction framework adapts to the log data. It assumes that only few variables are needed to express the interesting information, and finds a coordinate system that describes the global structure of the data. These coordinates could be used for further analysis of characteristics of anomalous activities.

The main benefits of this framework include:

- The amount of log lines that needs to be inspected is reduced. This is useful for system administrators trying to identify intrusions. The number of interesting log lines is low compared to the total number of lines in the log file.
- The unsupervised nature and adaptiveness of the framework. The proposed methods adapt to the structure of the data without training or previous knowledge. This makes it suitable for exploration and analysis of data without prior examples or attack signatures. This means that the framework also detects zero-day attacks.
- It works on the application layer in the network. The attacks themselves must in some way target the actual applications running on the computer. These logs might be more available than pure low-level network packet data.
- Visualization of text log data. It is much easier to analyze the structure of traffic using visualizations than it is to read raw textual log.

The data in question are rather sparse and the discriminating features are quite evident from the feature matrix. This is the merit of n-gram feature extraction which creates a feature space that separates the normal behavior in a good manner. The features describe the data clearly, and they are easy to process afterwards. Still, an attacker might take advantage of the features used by the intrusion detection system. If the n-gram frequencies of the attack query are similar enough to normal behavior, the currently proposed system could not detect the attacks. Also, if most of the traffic in a single log file consists of attack queries, they will be considered to be normal. This might be a problem in rarely used services.

One advantage of the diffusion map methodology is that it has only one metaparameter, ε. There exists estimation methods for finding the optimal value. If for some reason the threshold sensitivity needs to be changed, ε gives the flexibility to adapt to the global structure. However, the quality of the results is sensitive to changes of this parameter. Values that are too small or large give non-desirable results.

The presented anomaly detection framework performs well on real data. Several actual intrusions are detected. As an unsupervised algorithm this approach is well suited for finding previously unknown intrusions. This method could be applied to offline systems, as well as extended to a real-time intrusion detection system.

There are several points in this framework that could benefit from further research. The feature extraction from the web log is currently done with n-grams. However, this is only one method for it and other text-focused features might better describe the dataset. Furthermore, the dimensionality reduction scheme could be developed to adapt to this kind of data more efficiently, and the quality of the reduction could also be evaluated. The classification method may be improved or changed altogether to another algorithm. Finally, automated root cause detection would make the system more usable in practice.

Acknowledgements

The authors thank Professors Amir Averbuch, Timo Hämäläinen and Tapani Ristaniemi for their continued support. Thanks are extended to Juho Knuuttila and Kristian Siljander for useful discussions and ideas.

References

[1] Mukkamala, S. and Sung, A. *A comparative study of techniques for intrusion detection* (2003).

[2] Patcha, A. and Park, J. *An overview of anomaly detection techniques: Existing solutions and latest technological trends*. *Computer Networks*, 51(12):3448–3470 (2007).

[3] Pietro, R. and Mancini, L. *Intrusion detection systems*. Advances in information security. Springer (2008).

[4] Chandola, V., Banerjee, A., and Kumar, V. *Anomaly detection: A survey*. *ACM Comput. Surv.*, 41(3):1–58 (2009).

[5] Nguyen-Tuong, A., Guarnieri, S., Greene, D., Shirley, J., and Evans, D. *Automatically hardening web applications using precise tainting*. In R. Sasaki, S. Qing, E. Okamoto, and H. Yoshiura, editors, *Security and Privacy in the Age of Ubiquitous Computing*, volume 181 of *IFIP Advances in Information and Communication Technology*, pp. 295–307. Springer Boston (2005).

[6] Ramadas, M., Ostermann, S., and Tjaden, B. *Detecting anomalous network traffic with self-organizing maps*. In G. Vigna, E. Jonsson, and C. Kruegel, editors, *Recent Advances in Intrusion Detection*, pp. 36–54. Springer (2003).

[7] Tran, Q., Duan, H., and Li, X. *One-class support vector machine for anomaly network traffic detection*. *China Education and Research Network (CERNET), Tsinghua University, Main Building*, 310 (2004).

[8] Coifman, R.R., Lafon, S., Lee, A.B., Maggioni, M., Nadler, B., Warner, F., and Zucker, S.W. *Geometric diffusions as a tool for harmonic analysis and structure definition of data: Diffusion maps*. In *Proceedings of the National Academy of Sciences of the United States of America*, volume 102, p. 7426 (2005).

[9] Coifman, R.R. and Lafon, S. *Diffusion maps*. *Applied and Computational Harmonic Analysis*, 21(1):5–30 (2006).

[10] Bengio, Y., Delalleau, O., Roux, N.L., Paiement, J.F., Vincent, P., and Ouimet, M. *Feature Extraction*, chapter Spectral Dimensionality Reduction, pp. 519–550. Studies in Fuzziness and Soft Computing. Springer Berlin, Heidelberg (2006).

[11] Schclar, A., Averbuch, A., Rabin, N., Zheludev, V., and Hochman, K. *A diffusion framework for detection of moving vehicles*. *Digital Signal Processing*, 20(1):111–122 (2010).

[12] İzmirli, Ö. *Tonal-atonal classification of music audio using diffusion maps*. In *10th International Society for Music Information Retrieval Conference (ISMIR 2009)* (2009).

[13] Keller, Y., Coifman, R., Lafon, S., and Zucker, S. *Audio-visual group recognition using diffusion maps*. *Signal Processing, IEEE Transactions on*, 58(1):403–413 (2010).

[14] Turkka, J., Ristaniemi, T., David, G., and Averbuch, A. *Anomaly detection framework for tracing problems in radio networks*. In *Proc. to ICN 2011* (2011).

[15] Chernogorov, F., Turkka, J., Ristaniemi, T., and Averbuch, A. *Detection of sleeping cells in LTE networks using diffusion maps*. In *Vehicular Technology Conference (VTC Spring), 2011 IEEE 73rd*, pp. 1–5. IEEE (2011).

[16] David, G. *Anomaly Detection and Classification via Diffusion Processes in Hyper-Networks*. Ph.D. thesis, Tel-Aviv University (2009).

[17] Kruegel, C. and Vigna, G. *Anomaly detection of web-based attacks*. In *Proceedings of the 10th ACM conference on Computer and communications security*, pp. 251–261. ACM (2003).

[18] Hubballi, N., Biswas, S., and Nandi, S. *Layered higher order n-grams for hardening payload based anomaly intrusion detection*. In *Availability, Reliability, and Security, 2010. ARES'10 International Conference on*, pp. 321–326. IEEE (2010).

[19] Ringberg, H., Soule, A., Rexford, J., and Diot, C. *Sensitivity of PCA for traffic anomaly detection*. *ACM SIGMETRICS Performance Evaluation Review*, 35(1):109–120 (2007).

[20] Callegari, C., Gazzarrini, L., Giordano, S., Pagano, M., and Pepe, T. *A novel PCA-based network anomaly detection*. In *Communications (ICC), 2011 IEEE International Conference on*, pp. 1–5. IEEE (2011).

[21] David, G. and Averbuch, A. *Hierarchical data organization, clustering and denoising via localized diffusion folders*. *Applied and Computational Harmonic Analysis* (2011).

[22] Damashek, M. *Gauging similarity with n-grams: Language-independent categorization of text*. *Science*, 267(5199):843 (1995).

[23] Han, J. and Kamber, M. *Data mining: concepts and techniques*. Morgan Kaufmann (2006).

[24] Abdi, H. and Williams, L. *Principal component analysis*. *Wiley Interdisciplinary Reviews: Computational Statistics*, 2(4):433–459 (2010).

[25] Lee, J. and Verleysen, M. *Nonlinear dimensionality reduction*. Springer Verlag (2007).

[26] Nadler, B., Lafon, S., Coifman, R., and Kevrekidis, I.G. *Diffusion maps – a probabilistic interpretation for spectral embedding and clustering algorithms*. In T.J. Barth, M. Griebel, D.E. Keyes, R.M. Nieminen, D. Roose, T. Schlick, A.N. Gorban, B. Kégl, D.C. Wunsch, and A.Y. Zinovyev, editors, *Principal Manifolds for Data Visualization and Dimension Reduction*, volume 58 of *Lecture Notes in Computational Science and Engineering*, pp. 238–260. Springer Berlin Heidelberg (2008).

[27] Chung, F.R.K. *Spectral Graph Theory*, p. 2. AMS Press, Providence, R.I (1997).

[28] von Luxburg, U. *A tutorial on spectral clustering*. *Statistics and Computing*, 17:395–416 (2007).

[29] Ng, A.Y., Jordan, M.I., and Weiss, Y. *On spectral clustering: Analysis and an algorithm*. In *Advances in Neural Information Processing Systems 14*, pp. 849–856. MIT Press (2001).

[30] Shi, J. and Malik, J. *Normalized cuts and image segmentation*. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):888 –905 (2000).

[31] Kannan, R., Vempala, S., and Vetta, A. *On clusterings: Good, bad and spectral*. *J. ACM*, 51:497–515 (2004).

[32] Meila, M. and Shi, J. *Learning segmentation by random walks*. In *NIPS*, pp. 873–879 (2000).

[33] Sipola, T., Juvonen, A., and Lehtonen, J. *Anomaly detection from network logs using diffusion maps*. In L. Iliadis and C. Jayne, editors, *Engineering Applications of Neural Networks*, volume 363 of *IFIP Advances in Information and Communication Technology*, pp. 172–181. Springer Boston (2011).

[34] Hein, M. and Audibert, J. *Intrinsic dimensionality estimation of submanifolds in* $\mathbb{R}^d$. In *Proceedings of the 22nd international conference on Machine learning*, pp. 289–296. ACM (2005).

[35] Coifman, R., Shkolnisky, Y., Sigworth, F., and Singer, A. *Graph Laplacian tomography from unknown random projections*. *Image Processing, IEEE Transactions on*, 17(10):1891–1899 (2008).