ISSN 1392 – 1215 ELECTRONICS AND ELECTRICAL ENGINEERING
ELEKTRONIKA IR ELEKTROTECHNIKA
2008. No. 5(85)

T180 TELECOMMUNICATIONS ENGINEERING
TELEKOMUNIKACIJŲ INŽINERIJA

Unauthorized Network Services Detection by Flow Analysis

M. Ekmanis, V. Novikovs, A. Ruško
Faculty of Electronics and Telecommunication, Riga Technical University,
Riga, Azenes str. 12-137, LV-1048, phone: +371 29119856, e-mail: Martins.Ekmanis@mil.lv

Introduction

Most of non public networks have strict rules forbidding unauthorized setup of any services, servers and even applications on workstations. This is done in most of banks, government and military networks to provide necessary security level. Each unauthorized service or application possibly contains some malicious code to steal confidential information, open backdoors and can allow unauthorized remote control of the systems. In the same time some services must still run on these computers to allow remote administration, system updates and information exchange. Setting up of firewalls and intrusion detection system on each subnet is a time consuming and expensive process.

We propose to create a service detection system operating on flow data exported from access routers. The scope of this research is to confirm the hypothesis that network services can be recognized from flow data, even when random port numbers are used and payload is unavailable. A new distance measure between flow sources is proposed.

Network Flow Data

The IP network service is accessed by internet socket which is a unique communication endpoint. We denote each endpoint as a vector $I=\{i_1,i_2,i_3\}$, where i_1 – IP address, i_2 – protocol, and i_3 – port number (if applicable). The application therefore can be described as $A=\{F,I\}$, where F is the network flow vector. The network flow vector consists of group of flow records $F=\{f_1, f_2,..., f_n\}$, where each flow record consists of several counters $f_n=\{a_1,a_2,...,a_n\}$. As base counters we use bidirectional information about sent and received octets, the number of sent and the number of received packets per flow, and flow duration.

Initially we don't know what kind of applications are running in the network, which applications have similar functionality and which not. This can be solved by automatic classification of flows into different subsets (clusters). Clustering algorithms group objects according to some defined distance measure, hence we need some kind of distance function $d(A_1,A_2)$. Different distance definitions serve different purposes. There is the Hamming distance in coding theory [1], Levenshtein distance [2] in image comparison and spell checking, Chebyshev distance, Mahalanobis distance – correlation distance to compare signals, Normalized Compression Distance (NDC) based on Kolmogorov complexity [3] to compare data content like documents, images and emails etc.

Distinction Distance Measure

The Kolmogorov complexity $C(x)$ of a finite binary string x is the length of the shortest program that is able to return x (1). There is no definition about how the program should look like, or which programming language should be used.

$$NCD(x,y)=\frac{C(xy)-\min\{C(x),C(y)\}}{\max\{C(x),C(y)\}}. \qquad (1)$$

Similarly like in NCD we propose some other kind of algorithms (classification algorithms) as a distinction measurement between two data sets. If both data sets overlap, they cannot be separated well, so they are close to each other, otherwise they are distinct.

The performance of classification system (Fig.1) is commonly evaluated using the confusion matrix (Fig.2) [3].

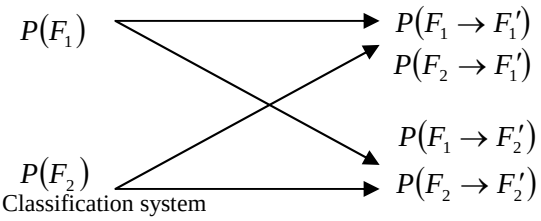

Fig. 1. Classification system

		Predicted	
		Class F_1	Class F_2
Actual	Class F_1	a	b
	Class F_2	c	d

Fig. 2. Confusion matrix

Information entropy quantifies the uncertainty of data, but classification system reduces it. In information theory

the entropy is defined as (2), where X is the set of all messages x.

$$H(X) = -\sum_{x \in X} p(x) \log p(x). \quad (2)$$

The joint entropy (3) implies that if X and Y are independent, then their joint entropy is the sum of their

$$H(X,Y) = -\sum_{x,y} p(x,y) \log p(x,y) \quad (3)$$

individual entropies. In case of classification system this means if the system cannot distinct classes, the system input and output will be independent. Thereof we can calculate mutual information (4), which measures the amount of information that can be obtained about

$$I(X;Y) = \sum_{x,y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \quad (4)$$

classes in input data by observing classification system output.

If both classes are not distinct, mutual information gained by system will be zero (5). Mutual information is symmetric (6) and positive (7), and that satisfies (8), (9) and (10).

$$I(X,Y) = 0, \quad (5)$$

$$I(X,Y) = I(Y,X) = H(X) + H(Y) - H(X,Y), \quad (6)$$

$$I(X,Y) \geq 0 \quad (7)$$

A space defined like this does not satisfy the triangle inequality (11), so it falls under the semimetric space class [4].

$d(x,y) = 0$ if and only if $x = y$; (8)

$d(x,y) = d(y,x)$ the distance between x and y is the same in either direction; (9)

$d(x,y) \geq 0$ the distance is nonnegative; (10)

$d(x,z) \leq d(x,y) + d(y,z)$ the distance between two points is the shortest distance along any path. (11)

The distinction distance is definable only in a limited range $d(x,y) \in [0;1]$, since maximum information gained when choosing between two states is 1 bit (12). If the sets do not overlap at all, they can be perfectly separated and have $d(x,y) = 1$

$$I(m) = \log\left(\frac{1}{P(m)}\right) = \log\left(\frac{1}{0.5}\right) = 1. \quad (12)$$

Classifier

The naive Bayes classifier [5] with several adjustments is used in our tests. This classifier is widely used in spam [6], intrusion and anomalies detection [7] systems. Using Bayes theorem we can obtain (13):

$$P(I \mid a_1, a_2, ..., a_n) = \frac{P(I)P(a_1, a_2, ..., a_n \mid I)}{\sum_{i=1,2} P(i)P(a_1, a_2, ..., a_n \mid I)} \quad (13)$$

In practice each attribute combination $a_1, a_2, ..., a_n$ is infrequent, so the formula will always return zero [8]. Assuming that each feature a_i is conditionally independent of any other feature a_j, we can rewrite it as (14). Assuming that each training set will consist of 50%

$$P(I \mid a_1, a_2, ..., a_n) = \frac{P(I)\prod_{j=1}^{n} P(a_j \mid I)}{\sum_{i=1,2} P(i)\prod_{j=1}^{n} P(a_j \mid i)}. \quad (14)$$

data from one class and 50% from another, we can reduce formula (14) to (15).

$$P(I \mid a_1, a_2, ..., a_n) = C\prod_{j=1}^{n} P(a_j \mid I). \quad (15)$$

The attributes are continuous so discretization must be performed. Flow attribute value distribution approximates well as exponential distribution (Fig.3), so exponential discretization can be used to get more regular data distribution. Optimal classification performance is obtained if the number of intervals is about 10% of the training set size. Divide 66% of each network flow vector F as training set and 34% as testing set is used.

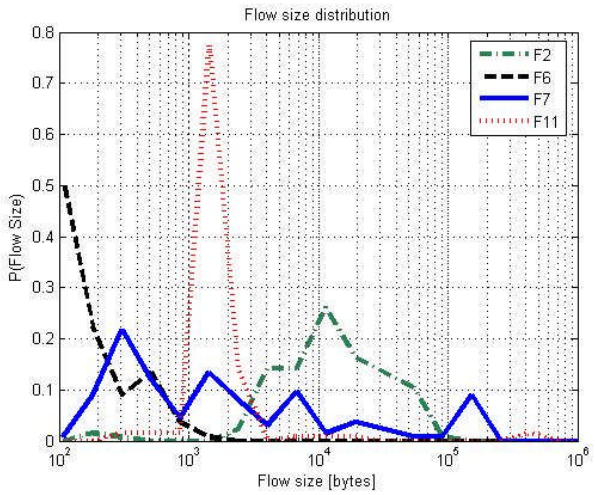

Fig. 3. Example of flow attribute value distribution (rx flow size)

Obtained Results

Experimental data was captured from production network DMZ connection to public internet. Argus Linux tool [9] was used to record bidirectional flow data from remotely mirrored switch port. Data was cleaned and inserted into MySQL database [10] using a shell script. The database is accessed by Matlab [11] using ODBC drivers and by Weka [12] using built-in JDBC driver. Twenty most active endpoints (Table 1) have been chosen for the experiment. Graphically examining flow data from

different sets (Fig.4) no visible distinction can be found. Values from different classes partly or completely overlap and classical classification algorithms correctly classify only about 25% of instances.

Table 1. Captured traffic covering 20 active endpoints

I	$i_1 / i_2 / i_3$
1	xx.yy.zz.230/udp/1234
2	xx.yy.zz.163/tcp/80
3	xx.yy.zz.166/tcp/80
4	xx.yy.zz.230/udp/4956
5	xx.yy.zz.230/udp/4374
6	xx.yy.zz.200/udp/53
7	xx.yy.zz.230/tcp/1234
8	xx.yy.zz.164/udp/678
9	xx.yy.zz.207/tcp/25
10	xx.yy.zz.164/tcp/678
11	xx.yy.zz.219/tcp/443
12	xx.yy.zz.163/udp/53
13	xx.yy.zz.204/tcp/22
14	xx.yy.zz.235/tcp/22
15	xx.yy.zz.236/tcp/22
16	xx.yy.zz.9/tcp/22
17	xx.yy.zz.4/tcp/179
18	xx.yy.zz.199/tcp/22
19	xx.yy.zz.205/tcp/22
20	xx.yy.zz.230/tcp/1723

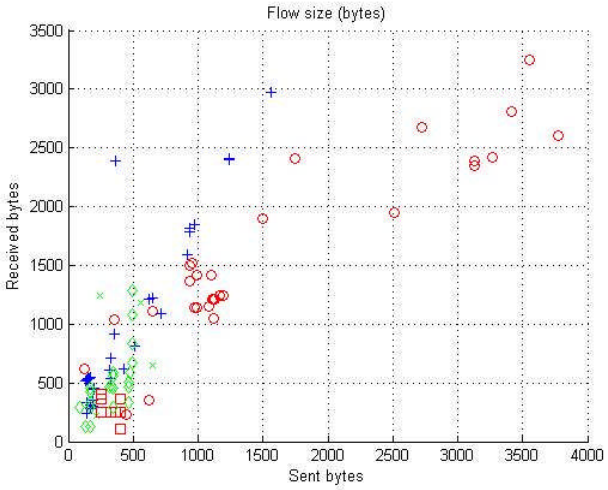

Fig. 4. Flow size attribute values from different overlapping sets

Table 2. Distance matrix covering 10 first endpoints

I	1	2	3	4	5	6	7	8	9	10
1	0.00	1.00	1.00	0.00	0.00	0.27	0.56	0.67	1.00	0.90
2	1.00	0.00	0.80	1.00	1.00	1.00	0.72	1.00	0.63	0.65
3	1.00	0.80	0.00	1.00	1.00	1.00	0.64	1.00	0.86	0.67
4	0.00	1.00	1.00	0.00	0.00	0.24	0.53	0.48	0.94	0.90
5	0.00	1.00	1.00	0.00	0.00	0.21	0.52	0.49	0.94	0.86
6	0.27	1.00	1.00	0.24	0.21	0.00	0.59	0.57	0.84	0.66
7	0.56	0.72	0.64	0.53	0.52	0.59	0.00	0.42	0.22	0.01
8	0.67	1.00	1.00	0.48	0.49	0.57	0.42	0.00	0.80	0.50
9	1.00	0.63	0.86	0.94	0.94	0.84	0.22	0.80	0.00	0.22
10	0.90	0.65	0.67	0.90	0.86	0.66	0.01	0.50	0.22	0.00

Distinction distance calculations have been done by Matlab scripts in several steps. First, import data from the database and split into the training and test sets. Do discretization on training values, calculating the probability (15) for each class and each interval. Second, use the adjusted naive Bayes classifier on each pair of classes and get the confusion matrixes. Third, mutual information (4) is calculated for each confusion matrix, therefore we can get the distance matrix (Table 2).

The distance matrix (Table 2) shows all three space equality, the matrix is symmetric, the diagonal is zero, and all values are positive in range between 0 and 1.

Flow Grouping by Cluster Analysis

Nonmetric distance usage limits available clustering methods [13] due to the absence of triangular inequality. In distance transformation, using triangle generating modifiers, it is possible to turn a semimetric space into a metric one [4], but this is a topic for another research.

Agglomerative hierarchical clustering is used to build the hierarchy from individual endpoints by progressively merging clusters. Inter-cluster distance can be calculated as complete linkage clustering (16) or single linkage clustering (17), because these methods do not require metric distances.

$$\max\{d(x,y): x \in A, y \in B\}, \quad (16)$$

$$\min\{d(x,y): x \in A, y \in B\}. \quad (17)$$

As the cluster progresses, the rows and columns of the distance matrix are merged, so the clusters are merged and the distances updated. The dendrogram (Fig.5) shows that similar services are grouped together even without taking into account the port numbers, protocols or host addresses. Remember though, that nonoverlapping clusters cannot be merged, because the distinction distance is not defined between such sets.

The vectors {13,19,14} represent open secure shell ports, {15,18} also represent ssh port but on a different operating system.

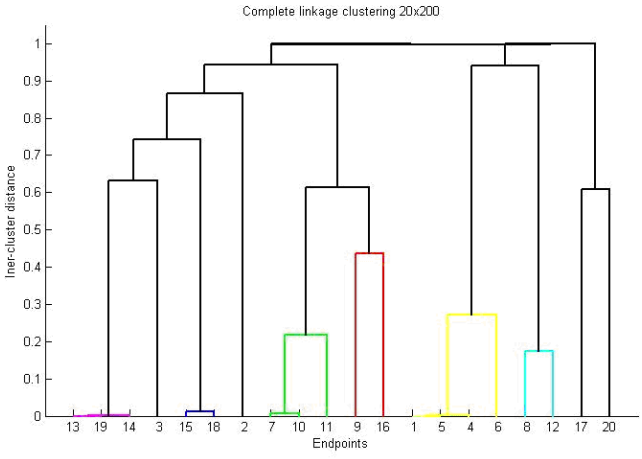

Fig. 5. Dendrogram, built using complete linkage clustering for 20 endpoints with 200 flow vectors in each

The vector pair {7,10} groups together a known bad host using uTorent P2P software on port 678 and an unknown host. A more accurate protocol analysis shows that this host also runs a P2P sharing application (bittorrent). The vector group {1,4,5} shows similar endpoints running on the same machine.

Conclusions

This research has shown that network services can be recognized by network flow data. Using distinction distance allows to compare these flows and use automatic classification mechanisms to group similar network services together. This solution allows recognize a network service even if different port numbers are used and the payload data is unavailable.

Using this mechanism, we can mark the traffic using a probe. Apply policies to a known service, and assign them to other services of the same type. This can be useful for detecting unauthorized services and virus infected machines, and for marking the traffic for service quality.

Acknowledgment

This work has been partly supported by the European Social Fund within the National Programme "Support for the carrying out doctoral study programm's and post-doctoral researches" project "Support for the development of doctoral studies at Riga Technical University".

References

1. **Richard W. Hamming.** Error Detecting and Error Correcting Codes // Bell System Technical Journal. – 1950. – Vol 26. – P.147–160.
2. **Damien Michel, Antonis A. Argyros, Manolis I. A. Lourakis.** Localizing Unordered Panoramic Images Using the Levenshtein Distance // Institute of Computer Science, Foundation for Research and Technology, Crete, Greece.
3. **Wehner S.** Analyzing worms and network traffic using compression. – April 2005.
4. **Tomáš Skopal**. Unified framework for fast exact and approximate search in dissimilarity spaces // ACM Transactions on Database Systems. – 2007. – Vol. 32, No. 4, Article No. 29.
5. **Рассел С., Норвиг П.** Искусственный интеллект. Современный подход // Издательский дом „Вильямс". – 2006. – С. 946–972.
6. **The Apache Software Foundation.** The Apache SpamAssassin Project. At: http://spamassassin.apache.org, last accessed January 2008.
7. **Kruegel C., Mutz D., Robertson W., Valeur F.** Bayesian event classification for intrusion detection // Computer Security Applications Conference. – 2003. – P. 14–23.
8. **Rennie Jason.** An Application of Machine Learning to E-Mail Filtering // CMU. – December, 1998.
9. **QoSient LLC.** Audit Record Generation and Usage System. – At: http://www.qosient.com/argus, last accessed January, 2008.
10. **MySQL AB**. The world's most popular open source database. At: http://www.mysql.org, last accessed January, 2008.
11. **MathWorks, Inc**. The Language of Technical Computing. At: http://www.mathworks.com, last accessed January, 2008.
12. **Ian H. Witten, Eibe Frank.** Data Mining: Practical machine learning tools and techniques, 2nd Edition. – San Francisco: Morgan Kaufmann, 2005.
13. **Bin Zhang, Srihari S.N.** A fast algorithm for finding k-nearest neighbors with non-metric dissimilarity // Proceedings of the Eighth International Workshop on Frontiers in Handwriting Recognition. – 2002. – P. 13–18.

Submitted for publication 2008 02 15

M. Ekmanis, V. Novikovs, A. Ruško. Unauthorized Network Services Detection by Flow Analysis // Electronic and Electrical Engineering. – Kaunas: Technologija, 2008. – No. 5(85). – P. 49–52.

There is no strong semantic structure in network traffic behavior so the most general abstraction query-by-example can be used to identify particular application. Automatic traffic grouping is also possible according to some similarity or dissimilarity distance, if such is defined. We propose a new distinction distance as a method to define the distance between network flows. Cluster analysis is done using distinction distance matrix calculated from real traffic flow dumps. The experiment shows the ability of algorithm to identify a traffic source by example and group similar sources together. Ill. 5, bibl. 13 (in English; summaries in English, Russian and Lithuanian).

М. Екманис, В. Новиков, А. Рушко. Определение несанкционированных сетевых приложений путем анализа потока сетевого трафика // Электроника и электротехника. – Каунас: Технология, 2008. – № 5(85). – С. 49–52.

В характеристиках сетевого трафика нет твердой семантической структуры, в общей абстракции для идентификации определенных приложений необходимо использовать запрос по примеру. Возможна также автоматическая группировка соответственно по любому сходству или различию дистанций, если таковы определены. В статье выдвигается новая отличительная дистанция как метод для определения дистанции между сетевыми потоками. Анализ кластера проведен путем вычисления матрицы отличительных дистанций для реальных данных потока трафика. Эксперимент показал способность алгоритма идентифицировать источник трафика по образцу и сгруппировать схожие источники вместе. Ил. 5, библ. 13 (на английском языке; рефераты на английском, русском и литовском яз.).

M. Ekmanis, V. Novikovs, A. Ruško. Neautorizuotų tinklo paslaugų nustatymas analizuojant duomenų srautus // Elektronika ir elektrotechnika. – Kaunas: Technologija, 2008. – No. 5(85). – P. 49–52.

Tinklo duomenų srautas neturi griežtos semantiškos struktūros. Bendruoju atveju konkrečiai taikomajai programai atpažinti galima naudoti pavyzdines užklausas. Taip pat galima automatiškai grupuoti duomenų srautus pagal tam tikrus atstumų panašumus arba skirtumus, jei tokie yra apibrėžti. Straipsnyje kaip atskiras metodas siūlomas naujas skiriamasis atstumas. Jis naudojamas atstumui tarp tinklo srautų nustatyti. Klasterio analizė atlikta skaičiuojant skiriamųjų atstumų matricos skaičiavimą realiems duotiesiems duomenų srautams. Eksperimentas parodė algoritmo gebėjimą identifikuoti srauto šaltinį pagal pavyzdį ir sugrupuoti panašius šaltinius. Il. 5, bibl. 13 (anglų kalba; santraukos anglų, rusų ir lietuvių k.).

DOI: 10.5755/j02.eie.11161