by

Teja Kommineni

A dissertation submitted to the faculty of
The University of Utah
in partial fulfillment of the requirements for the degree of

Master of Science

in

Computer Science

School of Computing

The University of Utah

May 2017

The University of Utah Graduate School

STATEMENT OF DISSERTATION APPROVAL

The dissertation of **Teja Kommineni**

has been approved by the following supervisory committee members:

Robert Ricci,	Chair(s)	__ Date Approved
Kobus van der Merwe,	Member	__ Date Approved
Suresh Venkatasubramanian,	Member	__ Date Approved
__,	Member	__ Date Approved
__,	Member	__ Date Approved

by **Ross Whitaker**, Chair/Dean of

the Department/College/School of **Computer Science**

and by **David Kieda**, Dean of The Graduate School.

ABSTRACT

Blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah. Blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah. Blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah. Blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah.

Blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah. Blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah. Blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah. Blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah.

Blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah. Blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah. Blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah. Blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah.

Blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah. Blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah. Blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah. Blah blah blah blah blah blah blah blah blah blah blah blah blah blah blah.

For my parents, Alice and Bob.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

NOTATION AND SYMBOLS

α	fine-structure (dimensionless) constant, approximately $1/137$
α	radiation of doubly-ionized helium ions, He++
β	radiation of electrons
γ	radiation of very high frequency, beyond that of X rays
γ	Euler's constant, approximately 0.577 215 ...
δ	stepsize in numerical integration
$\delta(x)$	Dirac's famous function
ϵ	a tiny number, usually in the context of a limit to zero
$\zeta(x)$	the famous Riemann zeta function
...	...
$\psi(x)$	logarithmic derivative of the gamma function
ω	frequency

CHAPTER 1

THE FIRST

This is a chapter.

CHAPTER 2

BACKGROUND

In this chapter we review Machine Learning and Data Mining techniques. ML/DM methods have become prevalent in the area of networking to gather insights from the collected network data.

Machine Learning is a method of generating rules from past data to predict the future. A Machine Learning approach usually consists of two phases: training and testing. The following steps are usually performed:

- Identifying attributes (features) and classes from training data.
- Choosing a subset of the attributes for classification.
- Choosing a ML algorithm to create a model using the training data.
- Use the trained model to classify the unknown data.

Data Mining is the process of extracting implicit, previously unknown and potentially useful information from data. Data mining is generally considered as a step in the process of Knowledge Discovery from Data (KDD). KDD usually consists of the following steps:

- Selection of raw data from which knowledge has to be extracted.
- Preprocessing the data to perform cleaning and filtering to avoid noise.
- Transforming the data so that all the attributes in the raw data are aligned towards achieving the common goal.
- Applying Data Mining algorithms to find rules or patterns.
- Interpret the observations of the above step and validate them.

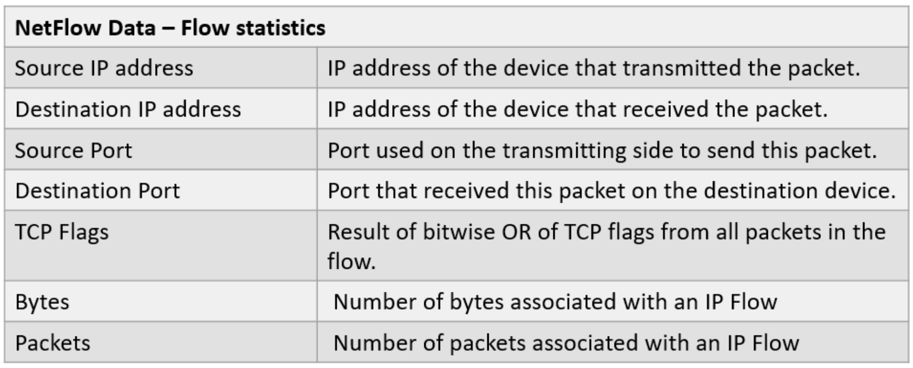
There is a significant overlap between these two techniques. ML/DM approaches come in 3 forms: unsupervised, semi-supervised and supervised. In unsupervised learning problems, the main task is to find patterns, structures or knowledge from unlabeled data. If a part of data is labeled then the problem is called semi-supervised learning. If all the data is completely labeled then it is called supervised learning.

2.1 Netflow

NetFlow is a proprietary protocol for collecting IP flow packets information on networks. Each record in a NetFlow is called a flow. Each packet that is exchanged within a router or switch is examined for a set of IP attributes. These attributes serve as the identity of that packet.All packets with the same source/destination IP address, source/destination ports, protocol interface and class of service are grouped into a flow and then packets and bytes are tallied. This form of aggregating packets to flow has an advantage because it decreases the amount of information needed to be stored in the database.

The captured flows are exported to NetFlow collectors at frequent intervals. The flows that are to be exported are determined based on the following rules: 1) when it is inactive for a certain time without receiving any new packets. 2) If the flow is active than max threshold time which is configurable. 3) If a TCP flag (FIN / RST) indicates the flow is terminated. NetFlow provides a powerful tool to understand the network behavior and is widely used for network monitoring. There are different tools that have similar flow approaches and a common standardization is done within the IETF IPFIX working group.

There are different variants of NetFlow with each advanced version giving additional information about the flow. The fields that we used in our experiment and that are supported across the versions are in figure 2.1.



NetFlow Data – Flow statistics	
Source IP address	IP address of the device that transmitted the packet.
Destination IP address	IP address of the device that received the packet.
Source Port	Port used on the transmitting side to send this packet.
Destination Port	Port that received this packet on the destination device.
TCP Flags	Result of bitwise OR of TCP flags from all packets in the flow.
Bytes	Number of bytes associated with an IP Flow
Packets	Number of packets associated with an IP Flow

Figure 2.1. NetFlow Fields captured in a Flow

CHAPTER 3

DESIGN OVERVIEW

Detecting the behaviors implicit in the netflow data and using them to build systems that make network management easier is the overarching premise of this work. In this chapter we describe the design of our system, and how the different components fit into the architecture.

3.1 Components

Our design is comprised of four essential components: Netflow Collector, Feature Engineering, Pattern Detector, and applications as shown in figure 3.1.

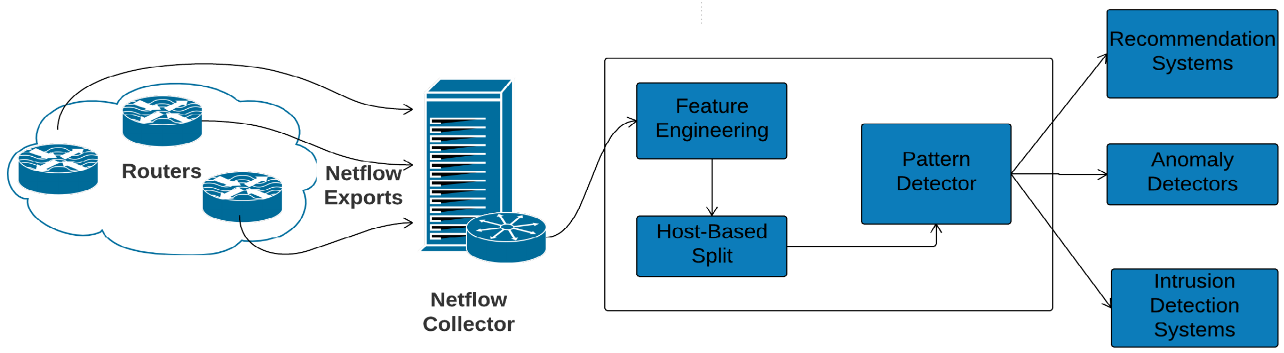


Figure 3.1. Architecture.

3.1.1 Netflow Collector

The raw data collected at the routers is exported to Netflow collectors as described in chapter 2. NFDUMP tools are used to process the netflow data. They support netflow v5,v7 and v9. The two tools of primary importance are nfcapd (netflow capture daemon) that reads the netflow data from the network and stores the data into files. nfdump (netflow dump) reads the netflow data from the files stored by nfcapd. Using nfdump we import the data on to our local machines from the Netflow Collectors. This imported data forms the primary source for our experiments. We collect real network traffic from a campus network with almost XXX users.

3.1.2 Feature Engineering

Feature Engineering is the process of transforming raw data into features that better represent the underlying structure to the models so that they increase the accuracy of the model when applied on unseen data. This step comes before modeling and after seeing the data. Features extracted from the data directly influence the models generated and predictions made out of it. The better the features are the better the results will be. The results that we achieve when we are using ML/DM approaches are a factor of the data set in hand, the features generated , the prediction model and the way the problem is framed. Most of the times even simpler models perform better if the features describe the structure inherent to the data. Feature Engineering encompasses of:

- Feature Scaling/Feature Normalization, a method used to standardize the range of independent features of data. Failure to standardize variables might result in algorithms placing undue significance to variables that are on a higher scale. There are different ways to do feature scaling namely, Min-Max scaling, Variance scaling, L2 normalization etc.. The choice depends on the data and different statistics of it such as Mean, Variance, L2 norm. Scaling is not advisable in all instances we might loose valuable information if applied without proper thought.

- Transforming Non-normal distribution to Normal, many ML/DM tools perform well on normalized data and having skewed data will give inaccurate results. Log transformations are generally used to normalize the skewed data. After transforming the data if it doesn't capture the essence of original data we could look at other options such as square root, cube root. BoxCox is also another technique that changes the distribution of variables from non-normal to normal or near normal.

- Missing data is a common issue that every data science experiment has to deal with. We applied rules such as consecutive rule (that says that there can be no missing values between the lowest and highest values for the attribute, and that all values must also be unique) and null rule (that specifies the use of blanks, question marks, or other strings that may indicate the null condition when a value for a given attribute is not available, and how such values should be handled.) to handle missing data.

- Features could be either numerical or categorical variables. When dealing with algorithms that work on numerical data we have to make sure that the categorical variables are handled. For example, in our case a category, Transport Protocol, has levels such as TCP, UDP and others. One approach is to encode them as 0, 1, and 2, respectively converting them to numerical variables. This could pose a problem when using distance measuring algorithms as the distance between the encoded attributes, like 1 (UDP) minus 0 (TCP) doesn't mean anything. Another approach would be to create a feature for each value of the categorical variable and mark it as 0 or 1 based on if the value is present or not. There are both pros and cons to this method. The pros are if we choose to aggregate the values on all features this conversion gives a numerical value to work with. The cons are scaling and standardization could be affected.

- Feature Selection refers to the process of selecting a subset of relevant features to model the data. It helps in shorter time periods of execution, overcoming the overfitting problem and making the solution more generic and avoid curse of dimensionality. The central premise of this process is to remove redundant or irrelevant features that don't make much sense to the problem we ought to solve. Feature selection and dimensionality reduction are two confusing terms. Though, both methods seek to reduce the number of attributes in the dataset, but dimensionality reduction does it by creating new combinations of attributes, where as feature selection methods include and exclude attributes present in the data without changing them. Principal Component Analysis, Singular Value Decomposition are examples of dimensionality reduction methods. The techniques used to do feature selection depend on text, numerical variables and they are three general classes of them Filter, Wrapper, Embedded methods.

3.2 Host-Based Split

The main theme of our work is to learn more about behavior of the network. Packet based inspection could be detrimental for this and that is why we have chosen NetFlow as our source for investigation as it bundles packets into flows and alleviates the problem of looking into each packet. But, if we carefully examine a flow is an instance of communication between two hosts. There could be multiple such communications that could happen between the same hosts over the day and all these could be recorded as different flows as explained in the chapter 2. So, the netflow records of a day can consist of multiple entries for the same hosts communicating. Hence, to get the whole picture of a host we aggregate the data based on source IP address. This is conducive in 2 ways. Firstly, With a small memory imprint we will be able to comment about behaviors of the host. Secondly, we deviate from the traditional approach of classifying based on applications or based on good/bad users.

3.3 Pattern Detector

3.4 Applications

CHAPTER 4

RELATED WORK

In this chapter, we review works that relate to usage of Machine Learning/Data Mining techniques in the area of networking and discuss in detail how researches are using these methods for different purposes. We also put in to the context how our problem statement and solution differ from the existing work and advances the state of the art.

First, we review the different ML/DM approaches that are being used for traffic classification and intrusion detection in the area of cyber security.

Second, we survey the vast body of work that does packet based inspection and also analyze the shift of research focus towards inspecting at higher granularities specifically the flow based inspection techniques.

Finally, we discuss how unsupervised approaches are being used to look at network data at higher granularities and compare our solution with the prior work.

4.1 Data Mining and Machine Learning Techniques for Intrusion Detection

There are many papers published in the area of cyber security describing the different ML/DM techniques used. Buczak et al [4] provided a survey of the popular techniques used and thoroughly describe ML/DM methods which provides a good starting point to people who intend to do research in the area of ML/DM for intrusion detection.

The survey by Buczak et al [4] concentration has been mainly on explaining the ML and DM methods where as Bhuyan et al. [3] in his paper described different techniques used for network anomaly detection in detail. Garcia's work [15] also explained different intrusion techniques in detail. He extended his work to use Machine Learning techniques for anomaly detection with focus on signature based intrusion-detection. Narayan et al. [20] proposed a hybrid classification method utilizing both Naive Bayes and decision trees for intrusion detection. While their findings had higher accuracy applying these supervised techniques to our dataset was not feasible as we had very less labeled data.

Wired networks generally have multiple layers of security before the intruder enters the network. But, the wireless networks are more vulnerable to malicious attacks. They add a whole new semantics to the network security such as dynamically changing topology, different authentication techniques and ad-hoc formation of networks. The process followed in this paper works in the context of both wired and wireless networks. Zhang et al. in his work [26] provides a perspective focused only on wireless network protection.

4.2 Machine Learning for Traffic Classification

Apart form using ML/DM techniques for intrusion detection one other field where these have been extensively used is to classify the network data. The survey [19] lists the prominent ML/DM techniques used to classify the data and describes the need for traffic classification. In order to provide the promised Quality of service and be liable to the government laws network administrators should be able to distinguish the traffic on their network.

Traditional well-known port number based classification is not sufficient on it's own to distinguish different applications as different services are using http protocol to send their data and obfuscate from the traffic classifiers. Also, papers such as choicenet [25] point out that to develop an economy plane for the internet similar to the implicit content based billing architecture in mobile architecture there is need for network administrators to classify the network data.

Naive Bayes form is the simplest technique used for this type of classification and has been explained in greater detail in the work of Andrew and Denis [18]. This work was extended by applying Bayesian neural network approach [1] for increasing the accuracy. Though these classification techniques proved to be efficient in differentiating network data they have a drawback that they can only distinguish the network data into known classes which is in contrast to our goal of identifying implicit behavior which is not known in advance.

Renata[2] looked at unsupervised techniques for traffic classification. In contrast to the existing approaches he followed a principle of early detection. Accordingly, he looked at the first few packets of tcp flow and classified them into different applications. The underlying logic behind this method is that during handshake process each application behaves in a specific way exchanging a particular sequence of messages. He used K-Means algorithm to form clusters. Initially, training data collected over a period is used to generate a model which gives a set of clusters. When new data arrives simple Euclidean distance is used to map this flow to a cluster. The flow belongs to the cluster which it is closest to. Though, this approach had 80 accuracy it has few drawbacks, If we cannot capture initial few packets of a service it's effectiveness is compromised. If a flow doesn't fall under any cluster the behavior is undefined. Renata's [2] approach was similar to us as we also explored unsupervised techniques in our system but they differ in the point that they examine the packet data and map their clusters to only known classes of traffic.

Jeffrey and Mahanti et al. [12] explored hybrid techniques for traffic classification. The intuition behind this exploration is that not all data available will be labeled and when data from new services get appended to the existing dataset the supervised learning techniques are falling short in recognizing them and they map the new data set to one of the existing classes. To overcome this shortcoming they approached the problem in two steps. In the first step the dataset containing labeled and unlabeled data is passed to a clustering algorithm. In the second step the labeled data is used to classify clusters and this is done as follows: Within each cluster all the labeled data is considered and the label with majority forms the label of this cluster. Clusters without any labeled data are classified as 'Unknown'. When new data arrives it will be assigned based on the Euclidean distance to the closest cluster similar to [2] . This has combined advantages of supervised and unsupervised learning. It also decreases the training time because of few labeled data. It's uniqueness comes from being able to map the data from new applications and services to Unknown cluster or to their respective clusters if they are simple variation of existing application's characteristics. A slight variation of this approach has been used by us during our experiments and the following issues have been noticed. First, there were cases in which we have seen a cluster mapping to multiple labels as both turned out to be major labels in the cluster differing by a small value. In this case labeling a cluster just based on majority doesn't suggest the clusters behavior. Second, the amount of labeled data could be negligible compared to the size of the cluster. Third, two clusters could turn out to have similar labels. Lastly, the time consumed for this two step approach is high as we had to iterate through the data twice. Noticing that this technique isn't inline with our goals we have embraced a totally unsupervised approach for our problem.

4.3 Usage of flow records for IDS/Classification

Traditionally Network data inspection is done by inspecting the contents of every packet. But, the granularity at which this is happening is changing based on the requirements. Earlier packet inspection is used to be a norm but inspecting the contents of every packet is prohibitive in this data-centric age. In this section we look at how flows (aggregated packet data) are being used as input for ML/DM algorithms in place of packets.

In our opinion, flow-based detection should not be seen as a replacement but should be treated as a complement to packet based inspection. We envision that a network administrator should be able to understand the behavior of his network through an aggregated view of network data (in our case flows) and should dive into packet data only when suspicious about a activity. This two step approach will ease the way the network administrators manage their network.

Work of Li et al. [13] and Gao et al. [14] are two examples of Dos detection using flows. In their approach a process tracks the presence of a flow in a specified time frame and an other process runs an anomaly-based engine that triggers if there is a sharp variation from the expected mean. A similar approach was proposed by Zhao et al.[27] to identify the IP addresses of the Scanning hosts and Dos attackers using the flow data. Our system though uses flows captures a broader view apart from identifying these specific attacks. Network Flows are also used in building Worm Detectors [6] [5], Botnet Detectors [22] [17] [16]. Specifically, the botnet detector built by Livadas et al. [17] is of interest as they build a model using the aggregated flows similar to ours.

REFERENCES

[1] T. Auld, A. W. Moore, and S. F. Gull, *Bayesian neural networks for internet traffic classification*, IEEE Transactions on neural networks, 18 (2007), pp. 223–239.

[2] L. Bernaille, R. Teixeira, I. Akodkenou, A. Soule, and K. Salamatian, *Traffic classification on the fly*, ACM SIGCOMM Computer Communication Review, 36 (2006), pp. 23–26.

[3] M. H. Bhuyan, D. K. Bhattacharyya, and J. K. Kalita, *Network anomaly detection: methods, systems and tools*, Ieee communications surveys & tutorials, 16 (2014), pp. 303–336.

[4] A. L. Buczak and E. Guven, *A survey of data mining and machine learning methods for cyber security intrusion detection*, IEEE Communications Surveys & Tutorials, 18 (2016), pp. 1153–1176.

[5] T. Diibendorfer and B. Plattner, *Host behaviour based early detection of worm outbreaks in internet backbones*, in Enabling Technologies: Infrastructure for Collaborative Enterprise, 2005. 14th IEEE International Workshops on, IEEE, 2005, pp. 166–171.

[6] T. Dubendorfer, A. Wagner, and B. Plattner, *A framework for real-time worm attack detection and backbone monitoring*, in Critical Infrastructure Protection, First IEEE International Workshop on, IEEE, 2005, pp. 10–pp.

[7] A. Einstein, *Eine Neue Bestimmung der Moleküldimensionen. (German) [A new determination of molecular dimensions]*, inaugural dissertation, Bern Wyss., Bern, Switzerland, 1905. Published in [8].

[8] ——, *Eine neue Bestimmung der Moleküldimensionen. (German) [A new determination of molecular dimensions]*, Annalen der Physik (1900) (series 4), 324 (1906), pp. 289–306. See corrections [9, 10]. This is a slightly revised version of Einstein's doctoral dissertation [7].

[9] A. Einstein, *Bemerkung zu meiner Arbeit: Eine Beziehung zwischen dem elastischen Verhalten. (German) [Remark on my paper: "A relationship between the elastic behavior ..."]*, Annalen der Physik (1900) (series 4), 339 (1911), pp. 590–590. See [11].

[10] A. Einstein, *Berichtigung zu meiner Arbeit: Eine neue Bestimmung der Moleküldimensionen. (German) [Corrections to my work: a new determination of molecular dimensions]*, Annalen der Physik (1900) (series 4), 339 (1911), pp. 591–592. See [8].

[11] ——, *Eine Beziehung zwischen dem elastischen Verhalten und der spezifischen Wärme bei festen Körpen mit einatomigem Molekül. (German) [A relationship between the elastic behavior and the specific heat of solid bodies with monatomic molecules]*, Annalen der Physik (1900) (series 4), 339 (1911), pp. 170–174, 590. See remarks [9, 10].

[12] J. Erman, A. Mahanti, M. Arlitt, I. Cohen, and C. Williamson, *Semi-supervised network traffic classification*, in ACM SIGMETRICS Performance Evaluation Review, vol. 35, ACM, 2007, pp. 369–370.

[13] Y. Gao, Z. Li, and Y. Chen, *Towards a high-speed routerbased anomaly/intrusion detection system*, Technical Report, (2005).

[14] ——, *A dos resilient flow-level intrusion detection approach for high-speed networks*, in Distributed Computing Systems, 2006. ICDCS 2006. 26th IEEE International Conference on, IEEE, 2006, pp. 39–39.

[15] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Maciá-Fernández, and E. Vázquez, *Anomaly-based network intrusion detection: Techniques, systems and challenges*, computers & security, 28 (2009), pp. 18–28.

[16] A. Karasaridis, B. Rexroad, D. A. Hoeflin, et al., *Wide-scale botnet detection and characterization.*, HotBots, 7 (2007), pp. 7–7.

[17] C. Livadas, R. Walsh, D. Lapsley, and W. T. Strayer, *Usilng machine learning technliques to identify botnet traffic*, in Local Computer Networks, Proceedings 2006 31st IEEE Conference on, IEEE, 2006, pp. 967–974.

[18] A. W. Moore and D. Zuev, *Internet traffic classification using bayesian analysis techniques*, in ACM SIGMETRICS Performance Evaluation Review, vol. 33, ACM, 2005, pp. 50–60.

[19] T. T. Nguyen and G. Armitage, *A survey of techniques for internet traffic classification using machine learning*, IEEE Communications Surveys & Tutorials, 10 (2008), pp. 56–76.

[20] S. Peddabachigari, A. Abraham, C. Grosan, and J. Thomas, *Modeling intrusion detection system using hybrid intelligent systems*, Journal of network and computer applications, 30 (2007), pp. 114–132.

[21] S. Singh, *Fermat's Enigma: The Epic Quest to Solve the World's Greatest Mathematical Problem*, Walker and Company, 435 Hudson Street, New York, NY 10014, USA, 1997.

[22] W. T. Strayer, D. Lapsely, R. Walsh, and C. Livadas, *Botnet detection based on network behavior*, in Botnet detection, Springer, 2008, pp. 1–24.

[23] R. Taylor and A. Wiles, *Ring-theoretic properties of certain Hecke algebras*, Annals of Mathematics, 142 (1995), pp. 553–572. This paper is a companion to [24], providing the remedy for the flaw in Wiles' 1993 proof of Fermat's Last Theorem. See also [21].

[24] A. Wiles, *Modular elliptic curves and Fermat's Last Theorem*, Annals of Mathematics, 142 (1995), pp. 443–551. This paper contains the bulk of the author's proof of the Taniyama–Shimura conjecture and Fermat's Last Theorem, carried out at Princeton University. The companion paper [23] contains the solution to the flaw discovered in the proof that Wiles announced on June 23, 1993, in Cambridge, England. See also [21]. In March 2014, now Royal Society Research Professor Sir Andrew John Wiles of Oxford University was awarded the prestigious Abel Prize in Mathematics for this proof — an award that also carries a cash prize of six million Norwegian crowns, or about US$722,000.

[25] T. Wolf, J. Griffioen, K. L. Calvert, R. Dutta, G. N. Rouskas, I. Baldin, and A. Nagurney, *Choicenet: toward an economy plane for the internet*, ACM SIGCOMM Computer Communication Review, 44 (2014), pp. 58–65.

[26] Y. Zhang, W. Lee, and Y.-A. Huang, *Intrusion detection techniques for mobile wireless networks*, Wireless Networks, 9 (2003), pp. 545–556.

[27] Q. Zhao, J. Xu, and A. Kumar, *Detection of super sources and destinations in high-speed networks: Algorithms, analysis and evaluation*, IEEE Journal on Selected Areas in Communications, 24 (2006), pp. 1840–1852.