An Autonomic Security Mechanism Based on Novelty Detection and Concept Drift

Gesiel Rios Lopes, André L. V. Coelho, Raimir Holanda Filho
Master's Course in Applied Computer Sciences
University of Fortaleza - UNIFOR
Fortaleza - CE - Brazil
Email: gesielrios@edu.unifor.br, {acoelho,raimir}@unifor.br

***Abstract*—The growth in the number of elements and services in current computer networks and the integration of several technologies, in order to provide connectivity and services everytime and everywhere, have renderede the computing infrastructure complex and susceptible to malicious activities. In this sense, the use of autonomic computing principles arises as an alternative to face these challenges, in special to deal with the development of security mechanisms capable of infering malicious activities with the purpose of self-protecting the infrastructure against attacks.This paper presents and evaluates a novel model of an autonomic element which is based on the notions of novelty detection and concept drift. The element is able to infer malicious activities that may compromise the proper functioning of the network. Experimental results, extracted from real samples network traffic, indicate that the element autonomic element infact is very useful.**

***Keywords*—Autonomic network, autonomic element, novelty detection, concept drift.**

I. Introduction

In recent decades the industry has increasingly produced new technologies, products and services, thus satisfying the demands for computer systems with large storage capacity and fast communication. These advances, although they bring important benefits, they have produced highly complex and heterogeneous systems. As a result, ensure security, fault tolerance, design capabilities, integration and management are becomming critical factors [1]. Therefore, solving the problems and difficulties generated by this scenario is a big challenge, which will only be achieved when computer systems be designed and built to adjust to changing situations, treating and managing of their resources efficiently. These factors are the main motivations for developing a new management system, delegating such tasks to machines, thus enabling the removal of administrative staff from the cycle and putting it only in a supervisory position. This view has been referenced in the academic community as autonomic computing [2].

This paper addresses the challenge of implementing an autonomic element, with the following combination of properties: continuous unsupervised learning of new concepts, an approach based on statistical properties extracted from the initial packet payload of the TCP flows and the use of a sorting algorithm of low computational power, the clustering algorithm k-means [3]. Moreover, our work is not limited to submitting a proposal, because it evaluates an implementation of an autonomic element using real traces. It is expected that the autonomic element proposed in this work will enable self-protection, self-configuration and able to learn new concepts over time, based on flows generated by the traffic.

The paper is organised as follows: Section II presents the concepts of novelty detection, the continuous learning OLINDDA (OnLIne Novelty and Drift Detection Algorithm) algorithm used by the autonomic element proposed in this paper and the k-means clustering algorithm. Section III presents our proposed autonomic element. Section IV describes the procedure for collecting and labeling the data used in the validation of the autonomic element. Section V presents our results and analysis, and finally, on Section VI our paper is concluded with some remarks and future works.

II. Novelty Detection

The mining of data streams brings several challenges to machine learning techniques. Its importance increases with the need for real-time analysis of a large amount of data. In this scenario, the ability to identify emerging concepts is an important attribute, and the Novelty Detection technique can contribute toward that goal. In Machine Learning, novelty detection can be defined as the identification of patterns that differ somehow from those normally expected, keeping a certain similarity with the concept of outliers [4]. In the context of this work, we consider a novelty as a new concept, i.e., an abstraction of instances or examples that share a set of characteristics that differ from those previously identified.

If the characteristics of a new concept are similar to a concept learned initially we use the term concept drift. Within this paper, was used the OLINDDA, a cluster-based algorithm which considers the problem of detecting concepts from Internet Traffic and operates over a continuous flow of data [4].

A. The OLINDDA algorithm

In this subsection, we present the functions of the OLINDDA. In terms of the learning paradigm, it may be divided in two phases: a supervised learning phase, where a normal model is built from a set of examples that describe the data domain, and an unsupervised learning phase, where unlabeled examples arriving from a data stream are analized

 ISBN: 978-1-61208-134-2

and novel concepts are continuously learned [4]. The implementation of OLINDDA used in the proposed autonomic element made use of the k-means clustering algorithm [5].

1) The k-means algorithm: Cluster analysis aims to divide the elements of one sample, or population, in clusters so that the elements belonging to the same group are similar to each other based on variables (features) used for measuring them, and the elements in different clusters are heterogeneous with respect to these same characteristics.

Let $x \in R^n$ be a vector of features with a known probability density p. We want to compute a partition of R^n, $\{R_1, R_2, \ldots, R_n\}$, representing each class by a vector $\hat{x}_i \in R_i$ that is called centroid. The choice of centroids and the partition is made to minimize the average distance of each vector to its centroid:

$$D = E\left\{d\left(x, r\left(x\right)\right)\right\} \tag{1}$$

The solution adopted in practice is to alternately estimate the centroid and the partition of space. The algorithm of Lloyd-Max is summarized in the following table [5]:

1) Initialization: Choose a static number of classes and initialize the centroids of each class according to some criterion.
2) Classification: Determine a partition of X, $\left\{X^1, X^2, \ldots, X^c\right\}$, associating each training sample to the class whose centroid is closest:

$$x \in X^k : k = \arg\min_i d(x, \hat{x}_i) \tag{2}$$

3) Update centroids: Calculate new centroids for each class by the equations

$$\sum_{x \in X_p} \frac{\partial}{\partial \hat{x}_p} d(x, \hat{x}_p) = 0, \qquad p = 1, \ldots, c, \tag{3}$$

where X_p means the data set assigned to the p-th class. If $d(x, y) = \|x - y\|^2$,

$$\hat{x}_p = \frac{1}{N_p} \sum_{x \in X^p} x, \qquad p = 1, \ldots, c \tag{4}$$

where $N_p = |X^p|$

4) Back to step 2 until there is a stop condition.

In the second step, each pattern is classified in the class of the closest centroid. During the third step, the centroids are recalculated, starting to occupy the midpoint of the standards in its class.

B. Distinction between normal and unknown

For construction of the normal concept, OLINDDA initially considers a set of examples that are used to represent the normal concept. The initial data are grouped into k clusters by the k-means clustering algorithm. For each cluster, we calculate the distance between the centroid and the farthest example. This allows us to establish a decision boundary for each cluster. The union of the boundaries of all clusters is the global decision boundary which defines the model. A new unseen example that falls inside this global boundary is consistent with the model and therefore considered normal; otherwise, it is labeled unknown. Otherwise, the example is marked as a member of an unknown profile and moved to a short-term memory for further analysis. That memory works like a FIFO queue to avoid its uncontrolled growth, eliminating old samples to permit the inserting of new samples [4].

C. Identification of novelty and concept drift

To monitor the formation of clusters in the short-term memory of unknown data, we use k-means to generate k candidate clusters. For a candidate cluster to be considered valid, which might indicate a concept in our approach, it must fulfill the following requirement. We use the in implementation of the proposed autonomic element, the sum of the squares of the distances between examples and the centroid divided by the number of examples as a validation criterion of the degree of cohesion, defined by:

$$d(x_j, \hat{x}_i) = \frac{\sum\limits_{x_j \in c_i} (x_j - \hat{x}_i)^2}{N_i} \tag{5}$$

where c_i represents the candidate cluster, $\hat{x}_i$ their respective centroids and N_i the number of examples of their respective centroids belonging in c_i.

Then, we compare this value to the mean distance between the centroid and examples of the candidate cluster. If the value for the candidate cluster is lesser than or equal to the one obtained for the model, the candidate cluster is considered valid. This restriction aims at selecting clusters whose density is not lower than that of the model. Once a candidate cluster has been validated: it may either represent a novel concept (novelty) or be an evidence that the normal concept is undergoing a change (concept drift).

To establish the limit between conceptual change and novelty, we calculate the global position of a centroid for the Normal template and then calculates the distance between this centroid and the overall centroid farther away, resulting in a decision boundary. If the centroid of the new cluster is beyond this boundary, the new cluster is considered a new and separate for further analysis. Otherwise, it is attributed to this new cluster a change of concept, and this information is used to update the standard model itself. An overview of the OLINDDA is shown in Figure 1.

III. Our proposed autonomic element

In this section, we present our proposed autonomic element (Figure 2), which aims to identify, by monitoring the network traffic, malicious activity resulting from deviations and possible changes in the normal concept.

Our autonomic element uses the general format of autonomic element, proposed by [2] and based on traffic subflows [6], which can be defined as a stream of packet being transmitted between a pair of hosts. Only TCP traffic was analyzed during this research, generating a set of 52 statistical variables candidates for discriminant, calculated based only on information obtained from the packet headers (for example:

 ISBN: 978-1-61208-134-2

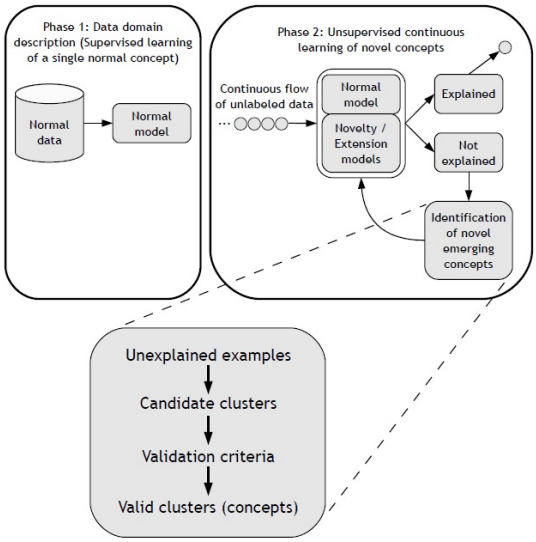

Fig. 1. Overview of the OLINDDA [4].

TABLE I
EXAMPLES OF CANDIDATE VARIABLES TO DISCRIMINANT

Candidate Variable
Total of packet (Client to Server)
Minimum Windows Size (Server to Client)
Mean of Inter-Packet Length Variation(Client to Server)
Maximum of Bytes in Ethernet Packet (Client to Server)

packet size and TCP flags), without the use of inspection of payload and port numbers. The variables are calculated for each direction of a bidirectional flow (client to server and server to client). Table I shows some of these variables.

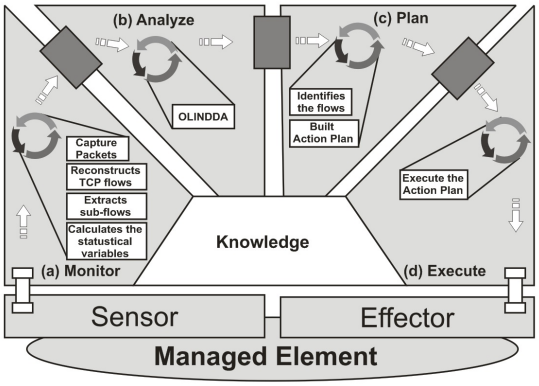

Fig. 2. Proposed Autonomic Element.

The implementation of the proposed autonomic element occurs in two phases: the first is called supervised phase and the second phase called unsupervised. In the first phase, the autonomic element determines the statistical variables that best discriminate the traffic considered normal(attack free) from a trace previously classified. At this stage the autonomic element use the mechanisms proposed to select the set of statistical candidates variables that best characterizes the normal traffic. Once you have determined the set of variables, the autonomic element builds the initial model, which will be used to identify potential malicious activities on the network and verifies the accuracy of the chosen variables. In the unsupervised phase, the proposed element performs the implementation of the autonomic cycle, as described below:

Monitor: The autonomic element (Figure 2(a)), works on the network traffic in promiscuous mode and reconstructs the TCP flows, extracts sub-flows, and calculates the statistical variables selected during the supervised phase, passing them to the analysis phase.

Analyze: Our autonomic element (Figure 2(b)), verifies if each sub-flow generated in the previous phase can be explained by the normal concept built during supervised phase. All sub-flows explained by the normal concept are labeled as normal to the next phase. The sub-flows not explained are temporarily stored in the short-term memory of unknown data. When the amount sub-flows stored in the short-term memory reaches a certain threshold, new clusters are generated, which will be labeled as a change of concept or novelty in accordance with the criteria used by OLINDDA and forwarded to the next stage.

Plan: Our autonomic element (Figure 2(c)), identifies the flows which were explained by the normal concept. These follow without any analysis in the next phase. The clusters labeled as concept drift or novelty are built into the normal concept. The information of the of remaining packets of sub-flows the clusters labeled as a novelty are stored, so they are discarded at the execute phase.

Execute: Our autonomic element (Figure 2(d)), in this phase execute the action plan established in the plan phase, i.e., all remaining packets of each sub-flows that were considered normal concept following without any changes and all the remaining sub-flows identified as novelty are discarded, because in our approach these packages represent some malicious activity.

Knowledge: stores the informations found by the autonomic element in all phases, such as the statistical variables, limits, the normal model, error rates, among other items.

Figure 3 describes temporally the stages of implementation of the proposed functioning of the autonomic element during the unsupervised phase. Initially, at time t_0, the autonomic element has only the model for normal traffic. In between times t_0 and t_1, the proposed autonomic element is performed in the presence of normal flows and classes of attacks. During this period abnormal sub-flows are stored in short-term memory of unkown. At time t_1, temporary memory is filled and moves the analysis of sub-flows into candidate clusters, validating and classifying them into innovations and changes in concept. From the time t_1, the autonomic element has proposed the concept of related clusetrs and normal clusters validated and classified as concept drift and novelty.

 ISBN: 978-1-61208-134-2

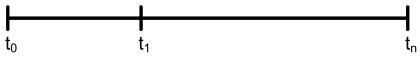

Fig. 3. Incorporation of novelty.

IV. Data and Measurements

Considering the difficulties to obtain actual samples of malicious traffic properly identified, we used artificially generated attacks and inserted into traces that are supposedly free of attacks to validate our experimental autonomic element. Thus, it is possible to verify whether the attacks can be correctly detected by our autonomic element.

To represent the normal traffic were used samples collected from a network gateway at the University of Fortaleza, during the period April 26-28, 2010. The captured packets were temporarily stored and the flows were reconstructed. Each flow was labeled with an application class. The process of labeling each flow was performed in a semiautomated manner through the use of the payload inspection tool OpenDPI [7]. Aiming to ensure that normal traffic really was free of malicious traffic, was also applied to each flows the tool Snort NIDS [8], with rules dated from November, 2010.

Our autonomic element was trained and validated through the use of 3 traffic datasetsts (referred as T1, T2 and T3). Each dataset was collected during periods of 1 hour (morning, afternoon and evening) and they contain the network traffic of the following classes: HTTP(browsers) and Snort Alerts (Alerts of possible flows of malicious activities identified by Snort [8]). Also were added to each dataset, 1250 attacks flows: Denial of Service and Brute Force, artificially generated.

The process of generating of the attack traffic, was conducted in a controlled manner in the laboratory. Figure 4 shows the topology used to generate the attack traffic.

In this scenario, we used three computers: the attacker, the victim and sniffer. The attacker has the role of sending traffic from attacks against services like FTP, SMTP and HTTP, to a victim. To generation the Denial of Service Attack and Brute Force Attack, were used the tools Heyenae [9] and Brutus [10], respectively.

Table II describes the classes, applications and total flows found within each dataset.

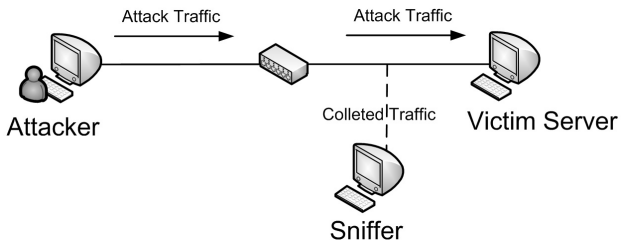

Fig. 4. Network topology used in the generation of artificial attacks.

V. Results and Discussion

In this section, we present the experimental results obtained with the implementation of our autonomic element using the process of cross-validation with 10 partitions [11]. To measure the accuracy of our autonomic element, we use the average rate of false-positive error e_{FPos}, which measures the average error committed when examples of sub-flows of attacks are considered as normal and average of false-negative error e_{FNeg}, which measures the average error committed when normal traffic flows are labeled as attacks. These metrics are given below:

TABLE II
Summary of Datasets.

Class	Number of Flows		
	T1	T2	T3
HTTP	6835	5284	2588
Snort Alerts	1845	1296	697
Denial of Service	1250	1250	1250
Brute Force	1250	1250	1250

$$e_{FPos} = \frac{\textit{Total of false-positive sub-flows}}{\textit{Total of normal sub-flows}} \quad (6)$$

$$e_{FNeg} = \frac{\textit{Total of false-negative sub-flows}}{\textit{Total of attacks sub-flows}} \quad (7)$$

In the supervised phase in order to determine the set of statistical variables that best characterizes the traffic to be considered normal, we used 50% of HTTP flows and 50% of Snort Alert flows of T1, T2 and T3. The remaining HTTP flows and Snort Alert, along with attack flows of T1, T2 and T3, were used in the unsupervised phase, to verify the learning ability of the our autonomic element.

A. Supervised phase

When considering the use of discriminating variables, it is essential to have measured in the sample elements, variables that can really distinguish the population, otherwise the quality of the classification will be compromised. A very common mistake is to think that increasing the number of discriminators, a better solution is reached. The Java implementation of the Wrapper [12] evaluator found in Weka [13] was used for selection of features that better characterize the flows associated to normal concept. Wrapper evaluates features using precision estimations produced by the learning algorithm that will be used on the classification. In this case, the Naïve Bayes and Best First were selected as search method for the Wrapper evaluator.

Figures 5 and 6 show, respectively, the results of the rates of e_{FPos} and the HTTP flows considered unknown in datasets T1, T2 and T3 for the set features selected by Wrapper for N initial packets.

As we can observe on Figures 5 and 6, the initial 11 packets utilization of each flow, was the result with lower average rates of e_{FPos} unknown HTTP flows considered by normal concept generated and so used by the autonomic element in the unsupervised phase.

 ISBN: 978-1-61208-134-2

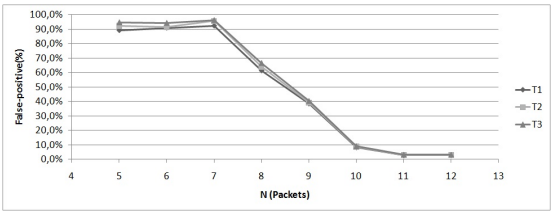

Fig. 5. Rate of false-positive error.

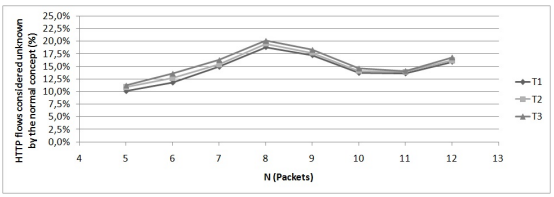

Fig. 6. HTTP flows considered unknown by the normal concept.

B. Unsupervised phase

The tabulated data below represent the results found by applying our autonomic element over 50% remainder set of HTTP flows 50% of Snort Alert flows and Denial of Service and Brute Force flows.

TABLE III
PERCENTAGE OF FLOWS CONSIDERED NORMAL.

Dataset	Class			
	HTTP	Snort Alert	Denial of Service	Brute Force
T1	66,11%	4,03%	0%	0%
T2	69,11%	3,12%	0%	0%
T3	67,51%	3,87%	0%	0%

Table III presents the percentage of flows considered normal by the autonomic element. We can see that on average 67.59% of HTTP traffic was correctly classified as normal, ie, were explained by the model correctly formed during normal supervision. We also had an average of 3.67% of false positive, flows of Snort Alert class that were classified as normal trafic. No Denial of Service and Brute Force attack, were classified as normal traffic flows.

TABLE IV
PERCENTAGE OF FLOWS CONSIDERED AS UNKNOWN.

Dataset	Class			
	HTTP	Snort Alert	Denial of Service	Brute Force
T1	33,89%	95,97%	100%	100%
T2	30,86%	96,88%	100%	100%
T3	32,49%	96,13%	100%	100%

Table IV presents the percentage of flows considered as unknown by the autonomic element. We can see that on average 32.15% of HTTP flows were classified as unknown. It is noteworthy that the initial error rate may indicate a slight concept drift of such flows. We also see that our autonomic element, obtained an average accuracy of 96.33% of Snort Alert flows and 100% in classes of attacks, which implies a high degree of discriminantion of the model generated during the supervised phase.

Tables V and VI show the Concept Drift and Novelty percentage of candidates clusters generated by autonomic element after analysis on the short-term memory.

TABLE V
CONCEPT DRIFT PERCENTAGE OF CANDIDATES CLUSTERS.

Dataset	Class			
	HTTP	Snort Alert	Denial of Service	Brute Force
T1	100%	0,83%	0%	0%
T2	100%	0,78%	0%	0%
T3	100%	0,81%	0%	0%

Table V presents the percentage of candidates clusters endorsed by our autonomic element of the type concept drift. We can observe that from the flows HTTP classified as unknown, 100% of there are classied concept drift, which indicates a slight concept drift of these flows relative to the normal concept determined during supervised phase. We also see that we had an average rate of 0.81% of Snort Alert flows regarded as the type concept drift, which may explain the rate of 3.67% of false positive found by our autonomic element. Table V also shows that none of the flows of the classes of attacks Denial of Service and Brute Force, generates clusters of the type concept drift.

TABLE VI
NOVELTY PERCENTAGE OF CANDIDATES CLUSTERS.

Dataset	Class			
	HTTP	Snort Alert	Denial of Service	Brute Force
T1	0%	99,17%	100%	100%
T2	0%	99,22%	100%	100%
T3	0%	99,19%	100%	100%

Table VI presents the percentage of candidates clusters endorsed by our autonomic element considered novelty. We can observe that none of the HTTP flows generated candidate clusters like novelty, which indicates a high degree of discriminator the of chosen variables and the normal model, both made during supervised phase. We can also observe that our autonomic element found a rate of 99.19% of candidates clusters of the Snort Alert class as type Novelty and 100% accuracy in the classes of attack artificially generated, showing once again the high discriminat degree of the variables chosen and the quality of the model generated during normal supervision.

From the results presented, we can observe a high capacity of the our autonomic element to identify the traffic considered normal, with an average accuracy of 98.77%. It noteworthy also that all normal sub-flows considered unknown, generated concept drift clusters and all sub-flows of the classes of attacks, were correctly classified as unknown and generated novelty clusters.

 ISBN: 978-1-61208-134-2

VI. Conclusions and Future Works

In this paper, we presented an autonomic element to provide self-protection and self-configuration in a computer network, that makes use of OLINDDA, a novelty and concept drift algorithm in data streams. The OLINDDA uses a cluster-based approach, which considers the problem of detecting concepts from an one-class classification perspective.

The implementation of the proposed autonomic element occurs in two phases: the first is called supervised phase and the second phase called unsupervised. In the first phase, the autonomic element determines the statistical variables that best discriminate the traffic considered normal(attack free) from a trace previously classified. In the unsupervised phase, our autonomic element implementation executes the autonomic cycle, using the set of variables and the normal concept is determined during the supervised phase.

From the results presented, we can observe that the implementation of our autonomic element showed an average accuracy of 98.77% of all flows classified as malicious, which indicates a high degree of accuracy in the classification of flows and a strong learning ability of our autonomic element. Despite our autonomic element has been applied in a small set of attacks, it can be perfectly applicable to other types of attacks. We intend in future work, add new attacks and simulate more realistic scenarios.

References

[1] T. R. M. Braga, F. A. Silva, L. B. Ruiz, and H. P. Assunção, "Redes autonômicas. anais dos minicursos do 26 simpósio brasileiro de redes de computadores," *SBC*, pp. 159–208, 2006.

[2] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," in *IEEE Computer Society*. 36(1): 41-50, 2003.

[3] S. A. Mingoti, *Análise de Dados Através de Métodos de Estatística Multivariada: Uma Abordagem Aplicada*. Editora UFMG, 2007.

[4] E. J. Spinosa, A. P. de Leon F. de Carvalho, and a. Gama, Jo "Olindda: a cluster-based approach for detecting novelty and concept drift in data streams," in *SAC '07: Proceedings of the 2007 ACM symposium on Applied computing*. New York, NY, USA: ACM, 2007, pp. 448–452.

[5] S. P. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, pp. 129–137, 1982.

[6] T. Nguyen and G. Armitage, "Training on multiple sub-flows to optimise the use of machine learning classifiers in real-world ip networks," *Local Computer Networks, Annual IEEE Conference on*, vol. 0, pp. 369–376, 2006.

[7] "Ipoque's dpi software's open source version." http://www.opendpi.org (as of November, 2010), 2010.

[8] "Snort is an open source network intrusion prevention and detection system (ids/ips)." http://www.snort.org (as of November, 2010), 2009.

[9] Hyenae, "Hyenae project (hyenae)." http://hyenae.sourceforge.net (as of November, 2010), 2010.

[10] Brutus, "Brutus - the remote password cracker." http://www.hoobie.net/brutus (as of November, 2010), 2001.

[11] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 2nd ed. Pearson Education, 2003.

[12] M. A. Hall, "Correlation-based feature selection for discrete and numeric class machine learning," in *Proceedings of the Seventeenth International Conference on Machine Learning*, ser. ICML '00. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2000, pp. 359–366. [Online]. Available: http://portal.acm.org/citation.cfm?id=645529.657793

[13] E. F. Ian H. Witten, *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, 2005.

 ISBN: 978-1-61208-134-2