Proceedings of the 5th WSEAS Int. Conf. on SIMULATION, MODELING AND OPTIMIZATION, Corfu, Greece, August 17-19, 2005 (pp250-255)

Decision tree based learning and Genetic based learning to detect network intrusions

Filippo Neri
University of Piemonte Orientale - DSTA
via Bellini 25/G, 15100 Alessandria AL, Italy

ABSTRACT
Abstract - The detection of intrusions over computer networks (i.e., network access by non-authorized users) can be cast to the task of detecting anomalous patterns of network traffic. In this case, models of normal traffic have to be determined and compared against the current network traffic. Data mining systems based on Genetic Algorithms can contribute powerful search techniques for the acquisition of patterns of the network traffic from the large amount of data made available by audit tools.

We compare models of network traffic acquired by a system based on a distributed genetic algorithm with the ones acquired by a system based on greedy heuristics.

KEY WORDS
Genetic algorithm, intrusion detection, machine learning.

1 Introduction

The raise in the number of computer break-ins, virtually occurring at any site, determines a strong request for exploiting computer security techniques to protect the site assets. A variety of approaches to intrusion detection do exist [2]. Some of them exploit signatures of known attacks for detecting when an intrusion occurs. They are thus based on a model of virtually all the possible misuses of the resource. The completeness request is actually a major limit of this approach [7].

Another approach to intrusion detection tries to characterize the normal usage of the resources under monitoring. An intrusion is then suspected when a significant shift from the resource's normal usage is detected. This approach seems to be more promising because of its potential ability to detect unknown intrusions. However, it also involves major challenges because of the need to acquire a model of the normal use general enough to allow authorized users to work without raising alarms, but specific enough to recognized unauthorized usages [8, 4, 10, 3].

Our approach follows the last philosophy for detecting intrusion and we describe here how it is possible to learn a model of normal use of a network from logs of the network activity. A distributed genetic algorithm REGAL [5, 13] is exploited for mining the network logs searching for interesting traffic patterns.

We are well aware that many aspects of deploying in practice learning system to acquire useful traffic patterns are still open including: selecting or building informative data representations, improving recognition performances (i.e., reducing both the rate of false alarms and of undetected intrusions), representing the traffic models for real world deployment (real-time classification of packets), and dealing with the shift in the patterns of normal use of the resources [9].

We concentrate here on the first two issues and we report our findings concerning the impact of different learning methods and of alternative data representation, with respect to the ones used in previous works, on the detection performances. As learning methods, we exploited two rule based systems: a heuristic one, RIPPER [1], and an evolutive one (based on genetic algorithms), REGAL [5, 13]. The first system has been selected because of its previous use [10]; it will thus act as benchmark. The second system has been selected because we believe that its intrinsically stochastic behavior should allow the acquisition of alternative robust and simpler models [13].

In the following, a description of the system REGAL (Section 2) and of the experiments performed in the Information Exploration Shootout (IES) and DARPA contexts (Section 3 and Section 4) are reported. The possible combination of classifiers obtained by REGAL and RIPPER, through meta learning is suggested to overcome each learner's limitations (Section 5). Finally, the conclusions are drawn.

2 THE SYSTEMS REGAL AND RIPPER

For space reason we will provide here an abstract description of both learning systems REGAL and RIPPER as their full descriptions have already been published. The system REGAL is available for free from the author.

REGAL [5, 13] is a learning system, based on a distributed genetic algorithm (GA). It takes as input a set of data (training instances) and outputs a set of symbolic classification rules characterizing the input data. As usual, learning is achieved by searching a space of candidate classification rules.

The language L used to represent classification rules is a Horn clause language in which terms can be variables

or disjunctions of constants, and negation occurs in a restricted form [12]. An example of an atomic expression containing a disjunctive term is *color(x,[yellow, green])*, which is semantically equivalent to *color(x,yellow) or color(x,green)*. Such formulas are represented as bitstrings that are actually the population individuals processed by the GA. Classical genetic operators, operating on binary strings, with the addition of task oriented *specializing* and *generalizing* crossovers are exploited, in an adaptive way, inside the system (for details see [5].

REGAL is a distributed genetic algorithm that effectively combines the Theory of Niches and Species of Biological Evolution together with parallel processing. The system architecture is made by a set of extended Simple Genetic Algorithms (SGA) [6], which cooperates to sieve a description space, and by a Supervisor process that coordinates the SGAs efforts by assigning to each of them a different region of the candidate rule space to be searched. In practice this is achieved by dinamically devising subsets of the dataset to be characterized by each SGA.

In other words, REGAL does include a form of meta-learning, i.e. the ability to combine several classifiers into a single one. Such a form of meta-learning can be easily realized by means of a cooperative genetic algorithms [14, 5]. That is exactly what the Supervisor process does.

The system RIPPER [1] is based on the iterated application of a greedy heuristic, similar to the Information Gain measure [15], to build conjunctive classification rules. At each iteration, those training instances correctly classified by the found rules are removed and the algorithm concentrate on learning a classification rule for the remaining one. The system outputs an ordered list of classification rules (possibly associated to many classes) to be applied in that same order to classify a new instance. An interesting features of the method is that it exploits on-line rule pruning while incrementally building a new classification rule to avoid overfitting.

3 INTRUSION DETECTION IN THE INFORMATION EXPLORATION SHOOTOUT CONTEST

An evaluation of REGAL over an intrusion detection task, by exploiting data from the Information Exploration Shootout Project (IES), is reported in this section. The IES made available network logs produced by 'tcpdump' for evaluating data mining tool over large set of data. These logs were collected at the gateway between an enterprise LAN and the outside-network (Internet). In the IES context, detecting intrusions means to recognize the possible occurrence of unauthorized ('bad') data packets interleaved with the authorized ('good') ones over the network under monitoring. The IES's project makes available four network logs: one is guarantee not to contain any intrusion attempts, whereas the other ones do include both normal traffic and intrusions attempts. In the IES context, no classification for each data packets is requested, instead an overall classification of a bunch of the network traffic, as containing or not attacks, is desired.

Table 1. RIPPER using the raw data

Dataset	interlan	incoming	outgoing
normal	0.04	0.04	0.04
intrusion1	0.23	0.07	0.04
intrusion2	0.09	0.07	0.05
intrusion3	0.08	0.14	0.04

Table 2. RIPPER using compressed data

Dataset	interlan	incoming	outgoing
normal	0.02	0.05	0.04
intrusion1	0.11	0.11	0.21
intrusion2	0.03	0.13	0.12
intrusion3	0.11	0.21	0.12

An approach to intrusion detection, based on anomaly detection, has been selected. We proceed as follows. IES data can be partitioned, on the base of their IP addresses, into packets exiting the reference installation (Outgoing), entering the installation (Incoming) and broadcasted from host to host inside the installation (Interlan). Three models of the packet traffic, one for each direction, have been built from the intrusion-free dataset. Then, these models have been applied to the three datasets containing intrusions. We expect to observe a significant variation in the classification rate between intrusion-free logs and logs containing intrusions because of the *anormal* characteristics of the traffic produced by the intrusive behavior. If this would actually occur, we could assert that the learned traffic models correctly capture the essential characteristics of the intrusion-free traffic. Experiments have been performed both with RIPPER and REGAL.

When RIPPER is applied to the IES data, the classification rate appearing in Table 1 (Experimental results of applying RIPPER to IES datasets using the raw data representation) becomes evident [10]. This results have been obtained by applying RIPPER to the data as available from the tcpdumped files (see Appendix A). No preprocessing over the data, such as feature construction, has been applied. The experimental findings shows that the acquired models do not exhibit very different classification rate when applied to logs containing intrusions with respect to intrusion-free logs. These findings may suggest that the exploited data representation is too detailed with respect to the capability of the learning system. In turn, this causes the learned models to miss the information characterizing intrusion-free traffic.

Following this observation, we develop a more compact representation for the packets that consists in mapping a subset of feature's values into a single value, thus reducing the cardinality of possible features values.

Table 3. REGAL using a compressed data

Dataset	interlan	incoming	outgoing
normal	0.02	0.04	0.04
intrusion1	0.12	0.15	0.11
intrusion2	0.06	0.11	0.12
intrusion3	0.12	0.15	0.11

Original Value	New Value
0≤srcport<50	srcport=0
50≤srcport<100	srcport=0
<... skipped test ... >	<... skipped text ... >
srcport>20000	srcport=10
<... skipped text ... >	<... skipped text ... >
op contains ”DF”	op=1
op contains ”NXDomain”	op=2
op contains ANY OTHER VALUE	op=3

Table 4. Compression mapping applied on IES network data.

As an instance of reducing the range of the feature values, considers that the feature 'srcport' (see Appendix A for a description) may virtually assume any integer number from 0 to 65536. Also, the feature 'op' may assume hundreds of discrete values. Taking into account basic knowledge about the domain, we manually developed the reduction mapping shown in Table 4. This mapping is not to be considered as the best one but as a proof that a simple reduction of the feature values may positively impact over the recognition capabilities.

Exploiting this representation, RIPPER's performances become the ones reported in Table 2 (Experimental results of applying RIPPER to IES datasets using a compressed data representation) and REGAL's performances exploiting the same compact data representation appear in Table 3 (Experimental results of applying REGAL to IES datasets using a compressed data representation). The observed figures show a more stable classification behavior of the models across different traffic conditions. Also a more distinct classification performance between the intrusion-free log and the logs including intrusions is evident. A compression-based representation is then a valuable way of increasing classification performances without introducing complex feature that may involves additional processing overhead. An evaluation of the effect caused by the addition of complex features to the raw network data representation has been performed in [10].

For the sake of clarity, an example of rule characterizing intrusion-free Incoming packets, learned by REGAL, appears in Figure 1 (Example of a rule characterizing part of the incoming traffic. The rule describes 7349 incoming packets without confusing them with any outgoing or interlan packet). The Incoming packets are characterized in term of the values of the features from their TCP/IP header. This rule successfully covers 7349 Incoming packets without being fooled by any Interlan or Outgoing ones. A description of the predicates appearing in the rule is provided in Appendix A.

```
IF    srcprt(x,[[0,20],[40,100],[150,200],[>500]]) and
      dstprt(x,[>1024]) and flag(x,[FP,pt]) and
      seq1(x,[[100,150],[200,300],[500,5000],[>10000]]) and
      seq2(x,[[50,100],[200,300],[500,20000]]) and
      ack(x,[[0,3000],[5000,10000]]) and
      win(x,[[0,2000],[>3000]]) and
      buf(x,[<=512])
THEN  IncomingPacket(x)
Coverage: (Interlan, Incoming, Outgoing) = (0, 7349, 0)
```

Figure 1. Example of a rule for incoming traffic.

4 INTRUSION DETECTION IN THE 1998 DARPA INTRUSION DETECTION EVALUATION PROGRAMME

We also performed an additional evaluation of our approach over network logs from 1998 DARPA Intrusion Detection Evaluation Programme [11] whose objective was to survey and evaluate research in intrusion detection. A standard set of data to be audited, which includes a wide variety of intrusions simulated in a military network environment, was provided. We exploited data available from the KDD'99 Intrusion Detection Contest[1].

The raw training data was about four gigabytes of compressed binary TCP dump data from seven weeks of network traffic. This was processed into about five million connection records. Similarly, the two weeks of test data yielded around two million connection records. A connection is a sequence of TCP packets starting and ending at some well defined times, between which data flows to add from a source IP address to a target IP address under some well defined protocol. Each connection is labeled as either normal, or as an attack, with exactly one specific attack type. Each connection record consists of about 100 bytes. Attacks fall into four main categories:
DOS: denial-of-service, e.g. syn flood;
R2L: unauthorized access from a remote machine, e.g. guessing password;
U2R: unauthorized access to local superuser (root) privileges, e.g., various “buffer overflow” attacks;
Probe: surveillance and other probing, e.g., port scanning.
In practice two datafiles containing classified connections are available: one has to be used for acquiring a model of

[1]Information about KDD'99 Intrusion Detection Contest is available on-line at http://www.epsilon.com/kdd98/task.html.

the traffic and the other one for testing its performances. The distinction is important because the test file contains attack types not occurring in the learning file. This is intended to make the task more realistic.

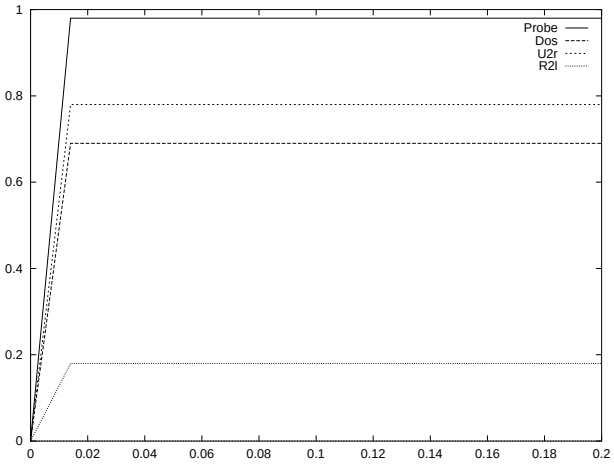

Figure 2. RIPPER plus Meta-Learnig.

In figure 2 (Detection performances exhibited by RIPPER plus Meta-Learnig on the DARPA test data. An extended representation of the data and a complex learning approach (meta-level learning) have been exploited) and figure 3 (Detection performances exhibit by REGAL on DARPA test data (no additional Meta-Learning has been used). A compressed data representation has been exploited), performances of RIPPER plus Meta-Learning (as used in [10]) and REGAL over DARPA's data are respectively shown by means of ROC curves. In the figures, the

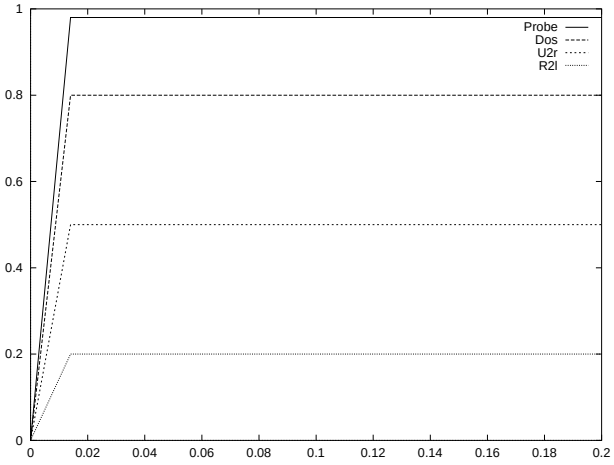

Figure 3. REGAL with compression.

x axis represents the false alarm rate, i.e. the percentage of 'Normal' connections labeled as intrusions, whereas the y axis represents the detection rate, i.e. the percentage of intrusions that have been correctly recognized. The reported performances are an average of the results obtained

Table 5. Average composition of a test set.

Instance Label	Size
Intrusion free	60593
Dos	223297
Probe	2377
R2l	5992
U2r	16

on three test files, whose average composition is reported in Table 5. The reported graphs show similar detection performances, between the models acquired by the systems, for Probe and Remote-To-Local (R2l) attacks types. Instead, REGAL's model performs slightly better on DOS type attacks but worst on User-To-Root (U2r) attacks.

Let consider, now, the modeling approaches exploited by the two systems. Lee and Stolfo [10] run RIPPER over an extended data representation of the tcp connection including, in addition to the basic tcp features, derived information such as: the number of connections to the same host in the past two seconds ('count'), the number of connections to the same service, as the current connection, in the past two seconds ('srv-count'). These features have been chosen on the basis of the authors expertise. A preprocessing of the raw network logs is required in order to exploits this features. Several classifiers (rule sets) for each attack type have been obtained. Eventually meta-learning, i.e. learning at the classifier level, has been applied to produce the reported performances.

Meta-learning consists, in this case, in learning how to combine several classifiers' outcomes into a single one that be more effective. The basic idea is that by applying a set of classifiers, each one associated to a different attack type, to an instance, a vector of potential attacks (i.e. classifications) can be obtained. Then, from that same original training set, it is possible to set up a learning task that learns to predict the instances correct classification from the vector of potential attacks. In conclusion, an instance classification is obtained through a two step process: first a set of classifiers (one for each attack type) is applied and a vector of potential attacks is obtained, then a (meta-)classifier is applied to that vector to predict the instance classification.

On our hand, we used a different approach to the problem. REGAL has been run after applying a compression mapping to the feature values, as described in for the IES data (see the previous section). Only the basic features of a TCP connection have been considered such as: 'duration', stating the length (number of seconds) of the connection, 'protocol-type', stating the type of the protocol (e.g. tcp, udp, etc.), or 'src-bytes', stating the number of data bytes from source to destination. Then a classifier for each attack type has been learned. No additional meta-learning phase is necessary, as all the learned classifiers are applied in the same order as reported in Table 5 to label a connection. If the connection does not satisfy any rule,

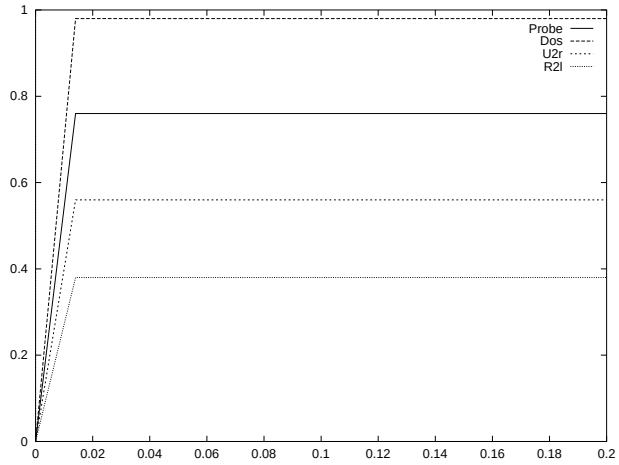

Figure 4. Detection performances of RIPPER.

the connection is considered to be Intrusion Free (default classification). If a connection gets a multiple classification, then it is considered to contain only the attack type associated to the first satisfied classification rule.

In summary, on the base of the experimental findings, we believe that the cooperating policy exploited by REGAL does positively compares against sophisticated learning approaches such as meta-learning and results in an easier to be deployed approach.

5 THINKING BEYOND: REGAL, RIPPER AND META-LEARNING

As additional question, we were interested in determining if the feature's compression mapping deployed with REGAL were system independent or not. Thus to test the effectiveness of this particular feature reduction, we performed an additional experiment where RIPPER is applied to the same data representation exploited for REGAL (discussed in the previous section), but without the meta-learning step. In figure 4 (Detection performances exhibit by RIPPER on DARPA test data. A compressed data representation has been exploited), the intrusion detection rate are reported.

The first observation is that the adopted feature compression result in quite good detection performance. Also we observe that the detection rates for the DOS and Probe attack type are swapped with respect to the ones appearing in figure 2. A possible explanation is that RIPPER select the DOS label as the default class and without the use of an additional meta-learning step this label is going to change the final classification performance. The other detection rates are as a matter of fact comparable to the ones obtained in the previous experiments.

Thinking beyond the current approaches, we think that would be interesting to apply meta-learning and classifiers learned by different learning system. This is theoretically simply to do but it will actually required a lot of engineering work. We believe, however, that combining classifiers learned by different learning methods, such as hill-climbing and genetic evolution, can produce higher classification performances because of the different knowledge captured by complementary search methods.

6 CONCLUSIONS

We shown the potentiality of two concept learners to the modeling of network data for detecting intrusions. Two different set-ups to deal with detecting intrusions have been explored.

We analyzed a data packet representation exploiting compression of the feature's values in the effort to reduce the complexity of acquiring model of the traffic. We believe this being an important requisite for the automatic modeling and the on-line deployment of intrusion detection system.

The experimental results support use of the compression of the feature values as a valuable method to increase detection performances while avoiding the use of derived and complex features that involve additional computational overhead.

A Appendix. The Information Exploration Shootout raw data representation

The IES data (available on line at http://iris.cs.uml.edu) have been collected by means of the TCPDUMP utility. Taking into account privacy concerns, the data portiong of each packet has been dropped. For each packet in the datasets the following attributes are available:
time - converted to floating pt seconds .. hr*3600+min*60+secs.
addr and port - (just get rid of x.y.256.256.port) The first two fields of the src and dest address make up the fake address, so the converted address was made as: x + y*256.
flag - added a "U" for udp data (only has ulen) X - means packet was a DNS name server request or response. The ID# and rest of data is in the "op" field. (see tcpdump descrip.) XPE - means there were no ports... from "fragmented packets".
seq1 - the data sequence number of the packet.
seq2 - the data sequence number of the data expected in return.
buf - the number of bytes of receive buffer space available.
ack - the sequence number of the next data expected from the other direction on this connection.
win - the number of bytes of receiver buffer space available from the other direction on this connection.
ulen - if a udp packet , the length.
op - optional info such as (df) ... do not fragment.
Particular attention has to be taken when dealing with fields like 'op' that contains a large amount of values.

References

[1] W. Cohen. Fast effective rule induction. In *Proceedings of International Machine Learning Conference 1995*, Lake Tahoe, CA, 1995. Morgan Kaufmann.

[2] D. Denning. An intrusion detection model. *IEEE Transaction on Software Engineering*, SE-13(2):222–232, 1987.

[3] S. Forrest, S. A. Hofmeyr, A. Somayaji, and T. A. Longstaff. A sense of self for unix processes. In *Proceedings of 1996 IEEE Symposium on Computer Security and Privacy*, 1996.

[4] A. Ghosh, A. Schwartzbard, and M. Schatz. Learning program behavior profiles for intrusion detection. In *USENIX Workshop on Intrusion Detection and Network Monitoring*. USENIX Association, 1999.

[5] A. Giordana and F. Neri. Search-intensive concept induction. *Evolutionary Computation*, 3 (4):375–416, 1995.

[6] D. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading, Ma, 1989.

[7] S. Kumar and E. Spafford. A pattern matching model for misuse detection. In *National Computer Security Conference*, pages 11–21, Baltimore, 1994.

[8] T. Lane and C. Brodley. An application of machine learning to anomaly detection. In *National Information Systems Security Conference*, Baltimore, 1997.

[9] T. Lane and C. Brodley. Approaches to online learning and conceptual drift for user identification in computer security. Technical report, ECE and the COAST Laboratory , Purdue University, Coast TR 98-12, 1998.

[10] W. Lee, S. Stolfo, and K. Mok. Mining in a data-flow environment: experience in network intrusion detection. In *Knowledge Discovery and Data Mining KDD'99*, pages 114–124. ACM Press, 1999.

[11] R. Lippmann, R. Cunningham, D. Fried, I. Graf, K. Kendall, S. Webster, and M. Zissmann. Results of the DARPA 1998 offline intrusion detection evaluation. In *Recent Advances in Intrusion Detection 99, RAID'99*, W. Lafayette, IN, 1999. Purdue University.

[12] R. Michalski. A theory and methodology of inductive learning. In R. Michalski, J. Carbonell, and T. Mitchell, editors, *Machine Learning, an Artificial Intelligence Approach*, volume I, pages 83–134. Morgan Kaufmann, Los Altos, CA, 1983.

[13] F. Neri and L. Saitta. Exploring the power of genetic search in learning symbolic classifiers. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PAMI-18:1135–1142, 1996.

[14] M. A. Potter, K. A. D. Jong, and J. J. Grefenstette. A coevolutionary approach to learning sequential decision rules. In *Sixth International Conference on Genetic Algorithms*, pages 366–372, Pittsburgh, PA, 1995. Morgan Kaufmann.

[15] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, California, 1993.