CS395/495 Computer Security Project #2

Winter 2005

Important Dates

Out: 1/19/2005

Due: 2/15/2005 11:59pm

Project Overview

Intrusion Detection System (IDS) is a common tool to detect the malicious activity launched by hackers. IDS can be classified as signature based IDS and statistics based IDS. In this project, you will be asked to use the signature based IDS and develop some parts of a tiny statistics based IDS.

Education Objectives

By the end of the project, you learned how to use IDS and how to develop IDS by your own.

Part I Signature based IDS

◆ Introduction

Snort is an open source network intrusion detection system, capable of performing real-time traffic analysis and packet logging on IP networks. It is a classical signature based IDS, which can perform protocol analysis, content searching/matching and can be used to detect a variety of attacks and probes, such as buffer overflows, stealth port scans, CGI attacks, SMB probes, OS fingerprinting attempts, and much more.

In many small businesses and medium size enterprises, Snort is many system administrators' favorite. So, in the first part of this project, you will be asked to learn how to use Snort.

◆ Specification

1. Detection

We will supply you 3 test datasets in tcpdump format. The 3 test datasets contain ICMP, TCP, and UDP traffic respectively. Please use Snort to detect the attacks in them, and name the three alerts file as alert.tcp, alert.udp, alert.icmp respetively. You can find the tarball of this part at /home/zli109/ cs395-prj2-partI.tar.gz

Note: please use the snort full log format, and only hand in the alert files of these three.

You can find Snort documents in the *reference* section.

2. Answer the following questions about your detection results

1) In the three datasets, how many alerts snort found in total, how many is TCP, and how many

is UDP, and how many is ICMP?

2) Which one is the most common alert in the TCP traffic
3) How many kinds of DOS attack does snort find? What are they?

Put the answer of these questions into a question_answer.txt, and include in your hand-in tarball

3. Add rules for snort to detect a new worm

Now suppose a new worm break out. The feature of the worm is:

1) It targets the TCP 8008 or UDP port 4004
2) It contains the signature "03 0E FE CC A0" follow by "PASS : RECV" within the 20 bytes of the first one.

The worm is coming from outside of our network (129.105.100.0/24). Do not alert if some machines inside our network to send the worm like traffic out. Write rules for this worm, to alert the network administrator with message "Worm Ditty!"

Put the new rules into a Ditty_rules.txt and include in your hand-in tarball

4. Add a firewall rule to block the new worm

Now you will be ask to add a firewall rule to block that worm. Suppose the firewall use this kind of rule format:

action	src	port	dest	port	flags	comment
allow/block	IPsubnet, use * to refer any host	port number or * (refer any)	IPsubnet, use * to refer any host	port number or * (refer any)	flag can be TCP, UDP	the description of this rule

Write firewall rules based on the above format to the Ditty worm traffic towards our computer science department (129.105.100.0/24).

Hint: assume that we do not have benign traffic on those services which the ditty worm rely on to propagate.

Put the rules you add to the firewall into a Ditty_firewall_rules.txt in your hand-in tarball

◆ Evaluation

The total points of this part are 30.

- ✓ Detection result (10)
- ✓ Answer questions (5)
- ✓ Snort rules for Ditty worm (8)
- ✓ Firewall rules for blocking Ditty worm (7)

◆ Deliverables and hand-in Instructions

The deliverables for this part are:

1) An ID.txt file which contain your names and your NU student ID.

2) The three alert files for the three datasets respectively
3) A question_answer.txt
4) A Ditty_rules.txt
5) A Ditty_firewall_rules.txt

To hand-in this part, please run the handin1.sh provide by us. Submission will be done through the dedicated webpage (which you can reach from the course site). You can re-submit the project as many times as you want before the deadline; simply follow the same procedure.

Note: you should submit the project2 part I and part II separately, just choose:

Project 2 -- PartI

to submit the part I.

◆ Reference

[1] Snort Users Manual http://www.snort.org/docs/snort_manual/
[2] SNORT FAQ http://www.snort.org/docs/FAQ.txt
[3] Jay Beale, Caswell, Snort 2.1 Intrusion Detection, 2nd edition. O'reilly Press. 2004. (no online version on the Internet, you may find it in the NU Library)

Part II Statistical based IDS

◆ Introduction

Comparing with signature based IDS, statistical based IDS uses the statistical metrics and algorithms to differentiate the anomaly traffic from benign traffic, and to differentiate different types of attacks. The advantage of statistical based IDS is that it can detect the unknown attacks.

As we know, DOS attacks happen frequently in the Internet. SYN flooding attack is an important form of DOS attacks. Port scan is another main category of malicious intrusions. However, some kinds of port scan can be easily hidden in the SYN flooding attacks, and this noise always confuses the administrator and causes the wrong responses. How to differentiate the SYN flooding from port scan is a hot research topic in the intrusion detection research. In this project, you will be asked to develop some key parts of a tiny statistical based IDS system to differentiate the SYN flooding attacks and port scan attacks from benign traffic, and further differentiate the SYN flooding from port scan.

We will supply you the main framework, and the statistical model for automatically inferring the threshold for detection. You will be asked to find the best statistical metrics which can easily characterize and classify the traffic into three categories: *normal, SYN flooding, port*

scan, and calculate the metrics from the raw network traffic trace. Also, based on the statistical models we provide, you should be able to detect the SYN flooding attacks and port scan attacks.

This tiny IDS project includes the following steps:

1) Obtain the data set at /home/zli109/cs395-prj2/DARPA98*, which is the network traffic trace of the DARPA98 IDS evaluation data set [1].
2) Calculate the metrics you selected
3) Use the statistical metrics calculated from the training data set to training the statistical model.
4) Use the model from (3) and the statistical metrics calculated from the test data set to detect the attacks in the test data set.

**) Do not copy the whole data set to your home directory, since it's too large, just read the data directly from that directory.*

The DARPA98 IDS evaluation data set contains one training data set and one testing data set. For each detection trial, you select one to five statistical metrics and calculate for both training and testing data set. The HMM model can accept up to 5 metrics to detect the attacks and the Gauss model can only use one. You need to figure out the best metrics combination to get the best detection result. And you can try as many different metric combinations as you like. Use the statistical metrics calculated for the training data set, which has annotated attacks, to train the statistical model. For the training phase, we give the ground truth of the attacks to the statistical model, which can automatically infer the threshold for differentiating three categories (normal, SYN flooding, and port scan) based on the different statistical characteristics of your metrics for the three categories. Next, you can use the trained statistical model and the statistical metrics calculated from the testing data set to detect the attacks in the testing data set.

In this project, we will supply you two classical statistical learning models: the HMM (Hidden Markov Model) [2] and the Double Gaussian model. You can use the two models together or just use one model. For example, you can use the HMM model to differentiate the abnormal (including SYN flooding and port scan) from normal, and use the Double Gaussian model to differentiate the SYN flooding from port scan; or you can just use HMM model to classify to 3 categories (normal, SYN flooding and port scan).

For each detection, you can select 1 to 5 statistical metrics. For example, you can calculate the volume of SYN packet minus the volume of SYN ACK packet as a metrics, in every 5 minutes time interval.

Note: In the DARPA98 data set we will only supply you TCP traffic.

◆ Specification

You first need to modify some C programs (see the *readme* in the tarball for details) to calculate selected metrics of the training and testing data set, and then use the Matlab program to train the statistical model and do detection.

Note: You can find the tarball for this part at /home/zli109/cs395-prj2-partII.tar.gz

Basically you need to do the following:

A. You need to modify the parsing C programs for both training and testing data. For training data sets, the output of your program should be a list of metrics plus the annotated flags. Each line presents the metrics calculated from the 5-minute network traffic packet data. The same output should be for the program of testing data sets, except which does not have the annotated flag in each line, instead, you need to put the time stamp in each line. We will provide you most of the code. You only need to add some metric specific code to it, for the metrics you select.
B. Based on the outputs of cal_training and cal_testing, you need to use the Matlab program (TinyIDS.m) to train the statistical model for detection. The Matlab programs will compare your result with the ground truth to calculate your accuracy.

1. Getting familiar with DARPA98 data set

As mentioned before, DARPA98 data set includes two parts: the training data set and the testing data set. The training data set contains 7 week (35 days) Tcpdump data. The training data set we supply to you is in plain text format and includes 35 data files. Each of the data files contain one day's network traffic data, which is in Tcpdump format.

In Tcpdump format, each line represents a packet transferred on the target network. *e.g.*
897048008.080700 172.16.114.169.1024 > 195.73.151.50.25: S 1055330111:1055330111(0) win 512 <mss 1460>

The first column 897048008.080700 is the time stamp of this packet, which is in an absolute time format. 897048008 is the number of seconds since 1970, and .0870700 is the microseconds.

The second column 172.16.114.169.1024 is the Source IP address 172.16.114.169 plus the source port 1024.

The third column > presents the direction of the traffic, which means the traffic is from source IP.port column 2 to the destination IP.port column 4.

The fourth column 195.73.151.50.25 is the Destination IP address 195.73.151.50 plus the destination port 25.

The fifth column S is TCP flag of the packet. S presents SYN.

The following columns are other TCP header fields. *Note*, ack flag may be in these fields. For more information about the format please ***man tcpdump***

In the training data set, we add one column at the beginning of each line to annotate which categories this line belongs to. 1 presents normal, 2 presents SYN flooding, 3 presents port scan.

The testing data set contains 2 weeks (10 days) data, in which we do not annotate the data, so you need to use the tiny IDS to detect the anomalies in it. Based on the ground truth* of testing data set, the TinyIDS.m program will output the error ratio of your detection program, so you can change or adjust your statistical metrics to get better detection result.

***Term:** ground truth --- The real attacks in the data set. If your detection results equal to ground truth, you will get 100 percent accuracy.*

Note: in the testing data sets, since we do not annotate which categories each line (packet) belongs to, the first column is the timestamp instead of annotation flags.

2. Metric calculation

Because the data sets are quite large, we recommend you to use C language to write the calculation program for high efficiency. You should write 2 programs (cal_training and cal_testing), one for the training data set and the other for the testing data set. Actually, we will supply most of the codes, what you should do is just add the metrics calculation code. In this program, we suppose you read the text Tcpdump trace data from *stdin*, and write the calculated metrics to *stdout*. Moreover, we will provide a *homenet.hosts* file which contains the all IP addresses of the attacked target network. Therefore, you should ONLY care about the traffic from these IP address or towards these IP address. Please use the following commands:

zcat <datasetfile> | cal_training ./homenet.hosts > metrics_training.txt
zcat <datasetfile> | cal_testing ./homenet.hosts > metrics_testing.txt

For the metrics calculation, we require you to calculate the metrics in every 5 minute traffic. For example, you may calculate the volume of SYN packets minus the volume of SYN ACK packets in every 5 minutes time interval. Use the time stamp to get the time.

For the metrics_training.txt file, we recommend the following format of each line:

Metric1 Metric2 … Annotation flag

Put the metrics you select in column 1 – n, if you select n metrics. The HMM system currently can simultaneously consider at most 5 metrics. So make sure n is no larger than five. Put the annotation flag in the last column. Please use one whitespace character to separate the different columns.

Note: In each 5-minute data, if any packet record is annotated as SYN-flooding, this record for the 5 minutes data should be annotated as SYN-flooding, otherwise if one packet record is annotated as port scan, the record for the 5 minutes should be annotated as port scan. The other records which do not have any SYN-flooding or port scan packet, should be annotated as normal.

For the metrics_testing.txt file, we require the following format:

Metric1 Metric2 … Timestamp

The timestamp can be used for comparing the ground truth with your detection results. Thus it is very important for grading. Please make sure you use the same form as the timestamp in Darpa98 data set.

Note: we provide a compathash.c and a compathash.h, which implement the hash functions in C language, feel free to use it.

Note, for debugging, try use

zcat <datasetfile> |head -100| cal_training ./homenet.hosts > metrics_training.txt

This command only feed your program first 100 lines of the tcpdump data, so it's easy for you to debug.

Note, in processing phase try use

zcat w.gz | cal_training ./homenet.hosts > metrics_training.txt*

3. Matlab program for training statistical model and detect the attacks in testing data set

We will supply you a set of Matlab files, including the following functions:

- ✓ TrainGaussClassiffier
- ✓ GaussClassiffy
- ✓ TrainHMM
- ✓ HMMInference

(For the detailed API specifications, see the comments in the *.m files)

You can use TrainHMM to train the HMM model and use HMMInference to detect the attacks in the testing data. Similarly, you can use TrainGaussClassiffer to train the Guassian Classifier, and use GaussClassiffy to detect the attacks. Be sure to put all the *.m

files we provide into the same directory as your Matlab files.

You need to modify the tinyIDS.m, in which you can finish the training and detection process based on the functions we provide. We will supply you the skeleton of the tinyIDS.m. The main purpose of the tinyIDS.m is to read the statistical metrics from the files, to train the model based on the training data set and detect the attacks in the testing data set. In addition, you can evaluate your results in tinyIDS.m.

4. Self-evaluation

After you finish the attack detection in the testing data set, based on the ground truth we provide to you, the tinyIDS.m will calculate the error ratio *E*.

Since most of your grade for this project will be based on your error ratio *E* result, you should try different metrics and combinations of the two statistical models to get as good results as you can. Try to minimize the error ratio *E*.

Note 1*: Here we have 3 categories:* ***Normal****,* ***SYN-flooding****, and* ***Portscan****. If you count any time interval into the wrong category, you error should be increased by 1. The total errors divided the total time intervals you get the error ratio E.*

Note 2*: A more complicated metric may not be better than a simpler one. Just consider how the attacks will affect the characteristics of network traffic (packets) to select the metric. Also, it is up to you how to use the two classifiers. Maybe only one classifier is Okay, maybe you need to use both. Ponder on the characteristics of these network attacks, and try as much as you can.*

Note 3*: In this project, our detection is a preliminary one. From the result, we only can know for each time interval whether it belongs to Normal, or SYN-flooding, or Portscan. But some attacks may last for many time intervals. Thus we do not really know how many attacks we detect and how many we mis-detect. Therefore, here we use the error ratio E to evaluate you results. Another reason for such metric is that we have 3 categories to classify each interval to. It is not a simple normal/abnormal question.*

5. Evaluation Report

You also need to write an evaluation report which includes the followings:

- ✓ Your comparative analysis about the statistical metrics chosen: why you decided to choose these metrics based on the characteristics of portscan and/or SYN-flooding?
- ✓ The best metrics and the detection results of the metrics.
- ✓ Some theoretical analysis, if possible, of why these metrics chosen by you are the best metrics.
- ✓ A list of some other metrics you tried but you do not think those are good ones. Please

include the results of those metrics as well, and why you think theoretically they are not as good as your best ones.

- ✓ A description of any important design decisions you made.

◆ Evaluation

The total points of this part are 70.

A. The metric and the accuracy of your result of testing data set (50) (we will test it separately)

B. Your evaluation report (20)

Note: if we cannot compile the program modify by you or cannot run the matlab code, 10 points will be deducted. Also, each warning message the C program generates will lose you 1 point.

You can enter the accuracy contest with your metrics and your implementation. Whoever has the highest detection accuracy will get 30 extra bonus points with a gift. If there are ties, then all the parties will get the 30 extra points plus gifts.

◆ Deliverables and hand-in Instructions

The deliverables for this project are:

1) An ID.txt file which contain your names and your NU student ID.
2) The source code of the metric calculation C programs
3) The source code of the detection Matlab programs (please denote which C and Matlab programs give you the best results so that we can test with them directly)
4) The metrics calculation result: *metrics_training.txt* and *metrics_testing.txt*
5) Your evaluation report (make sure to include your detection results in your report)

To hand-in this part, please run the handin2.sh provided in the tarball. Please submit it through the dedicated webpage (which you can find on the course website). You can re-submit the project as many times as you want before the deadline; simply follow the same procedure.

Note: you should submit the project2 part I and part II separately, just choose:

Project 2 -- PartII

to submit the part II.

◆ Reference

[1] Richard P. Lippmann, Robert K. Cunningham, David J. Fried, Issac Graf, Kris R. Kendall, Seth E. Webster, Marc A. Zissman, "Results of the DARPA 1998 Offline Intrusion Detection Evaluation," slides presented at RAID 1999 Conference, September 7-9, 1999, West Lafayette, Indiana.

[2] Lawrence R. Rabinner, A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," in Proceedings of the IEEE, Feb 1989.

◆ Appendix

In order to enlighten you a little bit about the statistical metrics you can use, in this section we will supply some of them as examples. But feel free to use anything else which is not on the list. *Note, the metrics in the list are just examples, which may or may not be good choices.*

- ✓ The traffic volume
- ✓ The packet volume
- ✓ The SYN packet volume
- ✓ The unresponded flows (The SYN packet volume – The SYN ACK packet volume)
- ✓ The mean or standard deviation of the packet volume per host/port in the target network
- ✓ The number of peers that one host connect with in a time interval
- ✓ The unresponeded flows/total packets in a time interval