Evaluation of Intrusion Detection Systems

Lei WEI

lwei026@ec.auckland.ac.nz

University of Auckland

Department of Computer Science

23 October 2007

Abstract

This paper proposes a group of criteria for evaluating Intrusion Detection Systems (IDS for short). In this article, an existing evaluation strategy of intrusion detection system is going to be briefly introduced as a case to be assessed against our proposed criteria. Intrusion Detection Systems are critical components to the primary defence of computer system and network security. An evaluation of IDS provides valuable data to improve the tested system. Through analysing the data, scientists are able to figure out the advantages and disadvantages of the system. In addition, evaluating Intrusion Detection Systems provides a way to study the attack mechanisms of some malicious programs. Developers of an intrusion detection system can update it according to the information, and hence new versions of the system have the ability to detect the same type of intrusions.

Acknowledgement

With thanks to Professor Clark Thomborson (University of Auckland) for his guidance on the research topic and recommendation to the relevant materials.

Section 1. Introduction

As the Internet is widely used today, malwares spread abroad rapidly. More and more computers are under the threat of being attacked by malwares. There are many kinds of malwares and their attack methods might be totally different.

Intrusion detection systems are created to enhance computer system protection and network security. Intrusion detection systems and Firewalls are often used together to protect computer systems such as the server and the database of a company. Intrusion

detection systems detect some attacks and record information about this, while firewalls prevent attacks. Intrusion detection systems are able to detect some attacks that are not able to be discovered by Firewalls. They raise warning messages, whenever malicious actions are detected. These warning messages remind operators taking steps to avoid risks and may also be used by Firewalls, so that the Firewalls could prevent the data packs coming from the suspicious sources that were believed safe.

However, new malwares are created every year. Compared with old malwares, new malicious applications are much more sophisticated on attack. This is a big challenge for intrusion detection systems to enhance computer security by detecting these new malwares.

Evaluating intrusion detection systems is very important on enhancing the computer security. It provides essential data and conclusions to help developers improving their IDS and enable users to know the capability and limitations of the IDS which is in use. In the real world, most Intrusion Detection Systems are implemented based on some unproven assumption concerning system performances [1]. Installing an IDS program without carefully evaluation may bring potential risks, since people may relax vigilance on those assumptions and neglect to construct some effective security posture that make use of detection and prevention mechanisms [1].

Evaluating intrusion detection systems enable scientists to study the way an intrusion detection system detects, monitors and, possibly, prevents attacks in run-time [1]. In addition, they could collect the result of attacks in the experiment of an evaluation. This is helpful for scientists to find out the methods about repairing damages of computer system; they can also study attack mechanisms of the malwares [1]. The result of the research is useful for developers to develop much more advanced Intrusion Detection Systems in the future. Developers update the current version of intrusion detection systems; so that the new versions of these IDS programs have bigger possibilities to warn systems avoid future attacks of the same type [1].

This paper aims to propose some points that should be considered when people are designing an evaluation of IDS and is going to discuss an existing evaluation framework of IDS according to these criteria. An evaluation of intrusion detection system should provide more pertinent data and analyzed results to help developers create powerful IDS,

to remind users being vigilant on limitations of IDS and to enable computer scientists finding out more helpful conclusions related to system security such as repairing damages and figuring out the attack mechanisms of new malwares. In section 2, we are going to list out the criteria about an evaluation framework and things needed to be considered in evaluation. An existing IDS evaluation strategy will be introduced as an example in section 3. Section 4 assesses this evaluation framework against our criteria. Finally, we would make a conclusion.

Section 2. Criteria to Evaluating IDS

This section proposes criteria to evaluate IDS. People who evaluate an intrusion detection system aim to know the system and to find out its capabilities and limitations. In addition, this is a way to monitor and study the attack mechanisms of new malwares and the way to repair damages.

As an evaluation framework of IDS, it should be feasible, less expensive, accurate and unbiased.

An evaluation of intrusion detection system should be feasible. A situation that is infeasible to evaluate is obviously not appropriate for the work. We would not get any expected results from infeasible evaluations.

A less expensive evaluation is preferable, compared with the one that is more expensive, as long as it satisfied the minimum requirements for accuracy and bias.

An accurate evaluation is obviously preferable to an inaccurate one, since accurate data and results are closer to the true value and reduce the possibility that researchers made wrong conclusions. To estimate accuracy, a usual way is to evaluate the IDS several times and then to calculate a “standard error of measurement”.

An evaluation should be unbiased; it is preferable to a biased one. However, estimating bias is difficult. A usual approach is to cross-validate results of evaluations. If one is consistently higher than the other, then there are some factors that cause the bias in one of both of the evaluations. It is only possible to get to the “ground truth” in simple cases. For example, one wants to minimize the bias in a measurement of weight on a scale or balance, it is necessary to set the weight-measurement apparatus so that it reads 0 when nothing is on it.

To evaluate IDS, demands made on IDS should be considered in a situation. There are many demands; Stefan Axelsson listed out around seven of them in [2]. People should choose the items that are most interesting to design their evaluation strategy. It is not necessary to cover all of the demands made on IDS in an evaluation. We recommend evaluators pay attention on the following demands:

Effectiveness. An evaluation should assess the ability that IDS is able to detect attacks and the percentage of false alarms. An Intrusion Detection System is able to raise alarm whenever there is an intrusion, while the false alarm rate should be kept on a low level which does not over users' tolerance. Ideally, the attack detection rate should be 1 and the false alarm rate should be 0.

Efficiency. Good IDS should consume less time and memories to detect intrusions and to report warning messages. Intrusion detection system is only used to provide security services such as detecting intrusions for computer systems. If it takes up quite a lot of resource, the rest of the resources may not be enough for services which are provided to users.

Ease of use. An intrusion detection system should not be too hard to allow a user who is not a security expert to operate it. Otherwise, users may give up using the intrusion detection system and choose other one. No one using an intrusion detection system is just the same as that the system does not provide any protection for the system security.

Security. For some new malwares, their attack mechanisms are much more sophisticated. They no longer stay at the stage of using IDS evasion techniques. Some of them try to attack IDS and make the system break down. An example of the malware is Stick which is created by Coretez Giovanni [3]. Stick executes a large number of simulated attacks in a short time. This causes the IDS on a target machine gets overloaded and then the system may stop responding. An Intrusion Detection System that has the ability to defend itself is preferred.

Interoperability. An Intrusion Detection System is able to interoperate with another one in some extension. It is impossible that an intrusion detection system is able to detect all

sorts of attacks. More than one IDS work together may significantly enhance the intrusion detection rate; however, this may also increase the false alarm rate and spend more resources as well.

Collaboration. IDS may be brought with other security mechanisms together to enhance computer system security. We need to evaluate the way that the intrusion detection system collaborates with the other security mechanisms, such as Firewall. We need to ensure the combination does provide a better overall security.

If the data that were generated and collected in evaluation are saved, it would be better. Evaluators may need them for further analyse or rechecking.

Section 3. Example of an Existing evaluation strategy

In 2006, Frédéric Massicotte and his colleagues proposed a framework which is able to automatically evaluate Intrusion Detection Systems [4]. The following of this paper retells the content of the paper, as we are going to take their work as an example to assess against our criteria.

The evaluation framework generates data set of network intrusion detection systems which is signature-based [4]. In this framework, there are two subsystems. One simulates attack scenarios and collects data; the other is an IDS evaluation framework.

The first subsystem creates a virtual network to simulate attack scenarios, documents and records down all the traffic traces. Alarms of the evaluated IDS are collected as well. For every traffic trace, the subsystem documents four characteristics of it— the target system configuration, the VEP configuration, whether the vulnerabilities of the target system has been exploited by the Vulnerability Exploitation Program (VEP) and whether the attack is successful [4].

The second subsystem collects all the alarms made by the IDS and relevant traffic traces from a shared hard disk. It compares the two groups of data and finds out the number of correct alarms (True Positive) and silences (True Negative) as well as the number of wrong alarms (False Positive) and silences (False Negative).

The working procedure of the whole evaluation framework is represented by the following figure. The black arrows show the working process of the attack simulation and

traffic traces collection. The grey arrows indicate the work of IDS evaluation process. To get a report about the test result, there are around ten steps:

1. The process chooses a malware to attack the target system and sets configuration of the target system.
2. The process uses a sandbox, VMware, to build a virtual network for a test.
3. The evaluation system sets the attack configuration on the virtual attacker.
4. The virtual attacker machine attacks the target and, meanwhile, the traffic traces are documented and recorded by the system. Alarms raised by IDS are also collected.
5. The recorded data are saved into a data set in the shared disk.
6. The virtual attacker and target machines are restored to their initial configuration for next round of test.
7. IDS Evaluator picks out all the traffic traces of a test case from the data set.
8. The IDS Evaluator put those traffic traces to the tested IDS.
9. IDS Result Analyzer gets those traffic traces as well as the alarms generated by the IDS. It compares the traffic traces against the alarms to figure out the number of intrusions are detected by the IDS in the test scenario.
10. The evaluation framework generates the report.

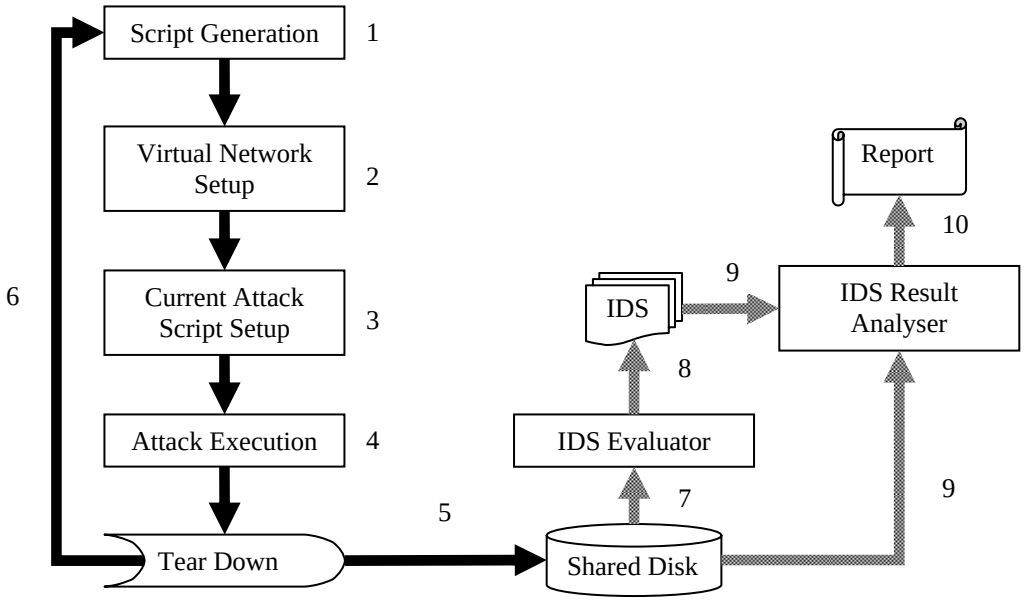

Figure1. The working process of Automatic Intrusion Detection System

In the report generated by the IDS Result Analyser, all the traffic traces are classified by two parameters, whether or not the attack is successful (S *or* $\neg S$) and whether or not the IDS has detected the attack (D *or* $\neg D$). Thus, there are four types of traffic traces, true positive ($S \wedge D$), true negative ($\neg S \wedge \neg D$), false positive ($\neg S \wedge D$) and false negative ($S \wedge \neg D$).

Frédéric and his colleagues proposed a 15-class taxonomy for all of the test results. These classes are formed by incorporating the four types of traffic traces which were collected in the test. For example, there are 242 out of 242 traffic traces are false positive in the test scenario that the malware, 0x82-w0000u_happy_new.c, attacks the target system with Snort 2.3.2 installed. The evaluation system concludes that the intrusion detection system, Snort 2.3.2, is alarmist when it is attacked by 0x82-w0000u_happy_new.c. Frédéric and his colleagues also evaluated two intrusion detection systems, Snort 2.3.2 and Bro 0.9a9, and made comparison between them. They found out the differences between the two systems.

Section 4. Assessing the evaluation strategy

In this section, Frédéric's evaluation strategy [1] is going to be assessed against our proposed criteria.

This evaluation is obviously feasible, as they successfully generated and collected documented data set and generated reports about analyzed results.

Compared with the evaluation frameworks which are done manually or semi-automatically, this evaluation strategy is less expensive. All of the tasks, attack simulation, data collection, comparison and report generation, could be worked out in one physical machine. Instead of requiring several physical machines to build a small local area network for testing, this evaluation uses a sandbox application on a physical machine to create a virtual network.

This evaluation framework could automatically repeat the test many times under the same scenario and generates the final report according to these results, as we could see from the tables in their report. This enhances the accuracy of the evaluation. The problem of the biased results is avoided by this strategy. When the system is evaluating an IDS program, it would finish all the relative tasks and generate the final report. During which,

people or other applications do not have to do other operations on the system, so the test results would not be affected by the outside world. In addition, the virtual target machine and the virtual attacker are restored to their initial configurations, after each round of test. Thus, the situation of next test is exactly the same as current test. Previous test would not have any impact to the results of next one.

If set the same situations for testing, the evaluation system can be used to compare different intrusion detection systems. In [4], two intrusion detection systems are evaluated and compared based on the exactly same situation.

However, the 15-class taxonomy which is proposed by Frédéric and his colleagues may not be a good idea. The classifications are not effective enough. This is only a qualitative analysis. For example, all the test cases that have both true positive and false positive and no other types of traffic traces are classified as alarmist and complete detections. However, the true positive rates of different intrusion detection systems are different. For an intrusion detection system, it is still acceptable, so long as the detection rate is high enough and the false positive rate does not over users' tolerance.

A quantitative analysis is preferable to the 15-class taxonomy. Through looking up information in a result analyze report, we could find out the amount of the four types of traffic traces. This enables us to easily obtain the percentages of true positive and false positive traffic traces, i.e. the correction detection rate and the false alarm rate. The two ratios are:

$$R_{correct} = \frac{number\ of\ true\ positive\ traffic\ traces}{total\ number\ of\ traffic\ traces} \quad (1)$$

$$R_{false} = \frac{number\ of\ \ false\ positive\ traffic\ traces}{total\ number\ of\ traffic\ traces} \quad (2)$$

Therefore, we know an accurate and unbiased value about the effect of an intrusion detection system against a malware.

Since the evaluation framework does not record the time and the space that were spent on intrusion detection, we are not able to figure out the efficiency of the tested IDS.

There is no way to assess whether the tested IDS is easy to use by using Frédéric's evaluation strategy. To know whether a tested intrusion detection system is easy to use, we must collect feedbacks from human users. However, the working processes of the evaluation are done automatically by a machine.

The evaluation framework allows vulnerability exploitation programs to use IDS evasion techniques. We could test effect of IDS under this situation. However, we are not sure what an intrusion detection system is going to do to defend itself, when an attacker uses the malware that would cause the intrusion detection system gets denial of service. This evaluation strategy does not provide us the answer.

This evaluation strategy does not provide any methods to assess the interoperability and the collaboration of a tested intrusion detection system. We do not know whether or not the tested intrusion detection system is able to interoperate with another one by the evaluation. We could not find out whether the tested intrusion detection system could collaborate with other security mechanisms either. No information about ways that malwares attack the target systems is provided.

Since all the documented traffic traces and IDS alarms are collected and saved, we are able to keep these data for further use.

Section 5. Conclusion

At the beginning of the term paper, we pointed out the significance to evaluate Intrusion Detection Systems. Evaluation of IDS supports essential information for improving those detection systems and hence, enhances system security. We proposed criteria for evaluation strategies and talked about those demands made on IDS. These demands are what users expect IDS could do and should be assessed in evaluation.

We retold an existing evaluation strategy and assessed it against our criteria. This evaluation strategy is able to accurately and economically help us figure out the effect of a tested intrusion detection system against some malwares, although it does not provide us ways to assess some demands on IDS.

In fact, it is not easy to design a perfect evaluation strategy. To get more data and valuable conclusions, an evaluation system would have to do more work and be more complicated. In other words, it is more expensive to build such a framework. It is not possible to ask an evaluation strategy providing good results that suit all the requirements without increasing the cost. We are not trying to force evaluation strategies cover all the requirements. The proposed criteria for evaluation of IDS are merely some guidance to help evaluators make appropriate strategy. Designers should try to evaluate the factors

that are most interested with them, while the unnecessary tasks could be done in the future.

Reference

[1] Richard Lippmann, Joshua W. Haines, David J. Fried, Jonathan Korba, Kumar Das, The 1999 DARPA Off-Line Intrusion Detection Evaluation, MIT Lincoln Laboratory, 244 Wood Street, Lexington, MA 02420-9108, USA, Available online 22 August 2000. http://www.sciencedirect.com/science?_ob=ArticleURL&_udi=B6VRG-411FRK94&_user=140507&_coverDate=10%2F31%2F2000&_rdoc=1&_fmt=&_orig=search&_sort=d&view=c&_acct=C000011498&_version=1&_urlVersion=0&_userid=140507&md5=d238fb8f84f3a778913d6341c046ca03

[2] Stefan Axelsson, “The base-rate fallacy and the difficulty of intrusion detection”, ACM Transactions on Information and System Security (TISSEC), Volume 3, Issue 3 (August 2000), Pages: 186-205, 2000, ISSN: 1094-9224

[3] "Stick"- A Potential Denial of Service against IDS Systems, (2007/10), Available: http://xforce.iss.net/xforce/alerts/id/advise74

[4] Frederic Massicotte, Francois Gagnon, Yvan Labiche, Lionel Briand, Mathieu Couture, "Automatic Evaluation of Intrusion Detection Systems," to appear in Proc. 22nd Annual Computer Security Applications Conference (ACSAC'06), pp. 361-370, 2006. Author's preprint available: http://cg.scs.carleton.ca/~mathieu/ACSAC06.pdf, 21 September 2007.