Defining Categories to Select Representative Attack Test-Cases

Mohammed S. Gadelrab
LAAS-CNRS
Université de Toulouse, France
gad-el-rab@laas.fr

Anas Abou El Kalam
IRIT/ENSEEIHT
Université de Toulouse, France
anas.abouelkalam@enseeiht.fr

Yves Deswarte
LAAS-CNRS
Université de Toulouse, France
yves.deswarte@laas.fr

ABSTRACT
To ameliorate the quality of protection provided by intrusion detection systems (IDS) we strongly need more effective evaluation and testing procedures. Evaluating an IDS against all known and unknown attacks is probably impossible. Nevertheless, a sensible selection of representative attacks is necessary to obtain an unbiased evaluation of such systems. To help in this selection, this paper suggests applying the same approach as in software testing: to overcome the problem of an unmanageably large set of possible inputs, software testers usually divide the data input domain into categories (or equivalence classes), and select representative instances from each category as test cases. We believe that the same principle could be applied to IDS testing if we have a reasonable classification. In this paper we make a thorough analysis of existing attack classifications in order to determine whether they could be helpful in selecting attack test cases. Based on our analysis, we construct a new scheme to classify attacks relying on those attributes that appear to be the best classification criteria. The proposed classification is mainly intended to be used for testing and evaluating IDS although it can be used for other purposes such as incident handling and intrusion reporting. We also apply the Classification Tree Method (CTM) to select attack test cases. As far as we know, this is the first time that this method is applied for this purpose.

General Terms
Security

Keywords
Intrusion Detection Systems, Attack, Evaluation, Test, classification.

1. INTRODUCTION
Known evaluations of Intrusion Detection Systems (IDS) suffer important shortcomings and often produce misleading results for several reasons [1, 2]. These defects are mainly due to (1) unsystematic approaches, (2) non-representative test cases, (3) incorrect metrics and (4) absence of sensitivity analysis on test datasets. It is worth noting that the aforementioned critiques are globally true whatever the context of the evaluation: as benchmarking with respect to a specific environment, as conformance test, as self-assessment of operational IDS, as test during development cycle, etc.

In order to provide a reasonable, complete solution and to obtain an unbiased evaluation, these problems should be eliminated or at least reduced. To resolve the problem of misleading ad-hoc evaluations, we derived a systematic methodology to evaluate IDS in [3]. In this paper, we focus on the second problem: the selection of representative test cases.

Intrusion Detection Systems should aim at detecting all attacks that either already exist or will be created in the future. In order to determine the superiority of an IDS and its quality of detection, it should be evaluated and tested against these classes of attacks. However, the attack space is too large to be enumerated and used as a dataset, whatever it is, for practical reasons. With this huge number of possible inputs to the IDS (i.e., attacks), the challenge resides in: (1) selecting a reasonably small number of attack test cases that provide a good representation of all possible inputs; (2) ensuring a good coverage to activate and exercise the different parts of the IDS.

In this work we borrowed two important concepts from software testing: the equivalence classes and the input parameter modeling. The underlying assumption of these techniques is that input instances from the same category have similar effects and thus generate similar outputs or results. Therefore, tests could be performed against only one or few representative samples from each equivalence class instead of the whole class.

To achieve that, we have conducted a thorough analysis of existing attack and vulnerability classifications to figure out their strengths and weaknesses. We studied the attributes of each taxonomy to determine how they could be relevant to our purpose (i.e., to produce an IDS evaluation-oriented classification). We eliminated the meaningless ones and combined several attributes from different classifications to produce a new classification. Moreover, based on this classification scheme, we applied the Classification Tree Method (CTM) [4, 17] in order to get an easy, semi-automatic selection of attack test cases. For this purpose, we initially used a tool called CTE [4], which can be used for generic application of the classification tree method. Then we began the development of a security specific tool similar to CTE which is also aware of existing attacks and exploits.

Such work should produce several benefits: first, it will reduce the attack test cases included in test datasets and make the test process more manageable. Second, it enhances the test dataset coverage of the attack domain space, as we can know those attack types against which the IDS is tested or not tested. Third, it yields a better knowledge of the domain as well as it enhances the understanding of new attack instances.

The paper is organized as follows: In Section 2 we analyze different attack taxonomies and then discuss the reasons that make these classifications helpless for the selection of attack test cases,

in Section 3. We introduce our classification which is intended to be used for IDS evaluation and testing in Section 4. Then, we propose a complementary approach for selecting attack test cases in Section 5. Finally, Section 6 draws a conclusion.

2. ANALYSIS OF AVAILABLE CLASSIFICATIONS

The problem of attack classification has attracted many researchers in the domain. However, they did not share the same objective. In order to not re-invent the wheel, we firstly examined the existing taxonomies. In the next two sections, we analyze several of them to see if one can match our needs and can be employed for IDS evaluation and testing. The detailed description of each taxonomy is out of the scope of this paper (refer to the [5] for more details).

We begin by Bishop's Vulnerability taxonomy [6]: Although this taxonomy is intended for classifying vulnerabilities, it might be useful to present it briefly because of its useful attributes (or axes). It has six axes: The *nature* of the flaw, the *time of introduction* of the vulnerability, the *exploitation domain* (i.e., the consequences of exploitation), the *effect domain* (what is affected?), the *minimum number of components* essential to exploit the vulnerability and the *source of identification* of the vulnerability.

Another interesting work is the two-dimension taxonomy that was introduced in [7]. It extends Neuman and Parker's taxonomy that have only the technique dimension [8] by adding the result dimension. It was built around attacks experimented by internal users (students of computer science class). The weak point of this classification is that it considers attacks launched by students in an undergraduate class. Therefore it ignores an important part of the attack domain space which consists of more sophisticated techniques by more experienced attackers.

Kumar had classified attacks according to four attributes of pattern or signature: existence, sequence, interval and duration [9].

Weber's taxonomy is based on three dimensions: the required level of privilege to conduct the attack, the mean by which the attack proceeded (e.g., exploiting a software bug) and the intended effect (e.g., a denial of service) [10].

DARPA's taxonomy is a reduced version of Weber's taxonomy. It considers only the effect dimension. Attacks were divided into five categories: Remote to Local, (R2L), User to Root (U2R), probe or scan and Denial of Service (DoS), [11], [12].

Howard's taxonomy [13] is based on the attack process rather than the attack itself. The attack process was divided into stages:

1. *attacker* (who is she/he? a simple hacker or a terrorist),
2. *tool* (what the attacker uses; a script kiddy or a specialized tool),
3. *vulnerability* (through implementation, configuration or design vulnerability),
4. *access* (what access is obtained; unauthorized access to files, objects or processes),
5. *results* of attack (exposure or corruption of data) and
6. *attack objectives* (e.g., destroy data, obtain information).

Simon Hansman's taxonomy [14] has four dimensions: *attack vector* (i.e., attack type: virus, worm, DoS, etc.), *attack target* (e.g., OS, application, network protocol), *exploited vulnerability* and *effects of attack*. The attack vector consists of the means by which the attack reaches its target.

The so called defense-centric taxonomy was introduced in [15] to serve network administrators in defending their own systems. It classifies attacks according to attack manifestations in system calls as seen by anomaly HIDS (i.e., anomaly host based intrusion detection systems). The four features, or dimensions, of interest are: (1) *foreign symbol*: the system call that appears when an attack is executed and never appears in normal operation. (2) *Minimal formal sequence*: the manifestations sequence that appears in the attack and do not appear in the normal operation, albeit all its subsequences appear in the normal operation. (3) *Dormant sequence:* the manifestations that partially matches a subsequence in the normal operation. (4) *Non-anomalous sequence:* manifestations that fully match sequences in the normal operation.

The taxonomy found in [16] was created for the purpose of analyzing IDS. It classifies activities that could be relevant to IDS instead of classifying attacks directly. An analytic evaluation of IDS was later established on it to determine its detection capabilities in front of attack classes. The underlying model of the observable manifestations distinguishes dynamic characteristics from static characteristics of activities. Static characteristics are split further to separate the characteristics related to interface objects and those related to affected objects. Similarly, dynamic characteristics are developed into three sub-characteristics: communication features, method invocation characteristics and other additional attributes. Then, an attack could be described by five parameters: interface object, affected object, communication, invocation method and other minor attributes. In total, it contains 24 interface objects, 10 affected objects, 2 communication characteristics, 5 method invocations and 4 minor attributes.

Before presenting our own classification in section 5, we will discuss in the next section the limitations of the previous classifications and analyze their attributes in order to select only the evaluation-relevant ones.

3. DISCUSSION

The previous attack taxonomies may have different viewpoints. One can notice that they are generally based on attributes of attack and/or vulnerability. Although using inconsistent attribute names, we can distinguish and cite the following ones amongst the used attributes:

1. *Type of attack*: virus, worm, trojan horse, DoS (Denial of Service), etc.;
2. *Detection technique*: pattern matching, statistical approach, etc.;
3. *Signature*: observed attack pattern, attack sequence pattern;
4. *Tool*: physical, user command, script, tool kit, etc.;
5. *Target* : OS, network protocol, application, service, process;
6. *Results*: Data corruption, exposure of information, denial of service;

7. *Gained Access*: root access, user access;
8. *Preconditions*: existence of particular versions of software by example;
9. *Vulnerability*: buffer overflow, weak password, inappropriate configuration,
10. *Objective* : terrorism, political gain, financial gain, self proving;
11. *Attacker* location: external, internal;
12. *Security property*: confidentiality, integrity, availability.

Taxonomies could be established upon a single attribute. Then the resulting categories will contain attacks that are widely different and share neither clear features nor strong relationships. For this reason, multi-dimension taxonomies have been formed by combining several attributes to obtain more distinctive categories that regroup more similar attacks. Almost all attack taxonomies suffer from the problem of mutual exclusivity. It was interpreted by the nature of sophisticated attacks as being composed of several blended attacks. However, this interpretation overlooked the real cause. If we examine classifications' attributes/dimensions, they are often not clearly defined and hence properties of attacks are not clearly separated. By example, putting both buffer overflow and DoS as classes bellow the same attribute will inevitably lead to a mutual exclusive problem because a buffer overflow attack may cause a DoS.

We also noticed that most of the aforementioned classifications are attacker-centric where they take the attacker viewpoint. Thus, they usually ignore or mask significant attack features, as seen by an IDS itself or system owners. By contrast, both the taxonomies of [15] and [16] are IDS-centric or defense-centric.

The categorization found in [16] was principally created for the analysis of IDS. It considers more details about attacks in terms of IDS characteristics. Although it seems to be more adapted for use in IDS evaluation and testing, it has some limitations. First, it has been uniquely focused on the manifestations of attack activities that could be observable by the IDS while ignoring completely other descriptive attributes that could be operationally so important such as: the consequences, the privileges required or obtained and the source of attacks. Such attributes are necessary for the configuration of evaluation platform and also to determine where/how the generated attack test cases will be injected. Second, it contains very fine grained dimensions even though the level of detail attained has minor significance for the tested IDS. For instance, the dimension *interface object* –that contains 24 types- considers 5 distinctive types related to the application layer:

App. layer-connectionless;

App. layer-single connection-single transaction;

App. layer-single connection-multiple transaction;

App. layer-multiple connection-single transaction;

App. layer-multiple connection-multiple transaction.

As a result of this fine granularity, it is not amazing to find classes with one or two attacks. We can relax this attribute by considering that an IDS is whether aware of and can analyze the activities at the application layer or not and whatever it is a multiple connection or single transaction. The worst case that this might affect the number of generated alerts. This could be treated when we analyze the evaluation results. Thus it does not worth to be included in the classification scheme with the advantage of much less classes. To explain this, we can theoretically obtain about 9600 test cases (i.e., all possible combinations), since any arbitrary combination of activity characteristic can be used. The number could be reduced to 8000 if we merged the classes related to the application layer into one class.

The conclusion of our analysis and discussion is that the currently existing attack classifications and taxonomies are not appropriate for the evaluation and testing of IDS for several reasons. First, they often take the attacker not the IDS (the defense) viewpoint and have attributes out of the scope of the IDS. Second, they have ambiguous, inconsistently defined attributes and hence problem of mutual exclusion. Third, they have a huge number of classes. Fourth, there is no accompanying scheme for test case selection.

In the rest of the paper we will present a new classification of attacks that aims to avoid the previous shortcomings. It builds on the previous work by keeping only the attributes judged to be relevant for IDS testing, giving them meaningful names and consistent definitions. We also avoided the ambiguous attributes and eliminated the attributes that are out of the IDS scope. Finally, we combined the new classification with a simple scheme for test case selection (i.e., the classification tree method) to get relevant test cases of representative attacks.

4. TOWARDS A NEW CLASSIFICATION

Amongst the critiques of DARPA's evaluations, which hold also for almost all subsequent IDS-evaluations, was the criteria according to which attacks was selected as test cases [1].

In this section we state the required characteristics of good classifications followed by a suggestion of a new taxonomy of attacks. Having a good classification of attacks that takes the evaluator's viewpoint will be extremely useful for several reasons:

First, it will reduce drastically the number of necessary test cases. Second, more comprehensive evaluation could be obtained because selective generation of test-cases according to a good classification will provide better coverage of attacks. To explain that, let us consider the random selection of attack test-cases. Evaluators usually test their systems in an ad-hoc manner using few attack scripts available in their hands or on security mailing lists. However, the available attack scripts do not reflect the attack distributions or even do not cover some critical attack types. Some IDS evaluations such as [11] and [18] were accompanied by some kind of taxonomy but they are either superficial or reporting-centric taxonomies that are less suitable for IDS evaluations.

Third, expressing the results of the evaluation in terms of attack types will provide a more precise image of results with respect to particular types of attacks. For example, a misunderstanding could arise from the generalization of conclusions when expressing the results for all attacks included in test-cases whereas the tested IDS is weak in detecting certain type of attacks and strong in detecting another.

Before proceeding, it is worth to mention here that we use the terms: class, type and category as synonyms. The terms: attribute, axe and dimension are used interchangeably to signify the feature or the criteria of classification.

4.1 Classification Requirements

In order to obtain a good classification, there are some general requirements that should be satisfied. Such requirements of a reasonable classification were stated in [7], [14], and [16]. The most important ones are:

1. *Completeness/exhaustive:* it means that a categorization scheme should take into account all possible attacks (e.g., known and unknown).
2. *Clear and unambiguous criteria:* if each dimension has a number of *distinct classes*, an attack can be classified by picking up one and only one distinct class from each dimension.
3. *Mutually exclusive:* to ensure that an attack is placed at most in one category, a dimension has only mutual exclusive distinct classes.
4. *Repeatable:* The clear steps followed to classify an attack ensure that it should be placed always in the same category.
5. *Compliance with existing standards and terminology:* since vulnerability databases and dictionaries had become de facto standard in security, it was included in our taxonomy. This dimension has a great importance because vulnerabilities have a tight relation with attacks.
6. *Extensible*: when new attack classes appear, the categorization scheme should be able to classify them. In our scheme, new dimensions can be added and existing dimensions can be extended. For instance, the target and the carrier dimensions could be widened to contain more targets and more carriers respectively. Therefore, even theoretical attacks, that do not exist yet or not known yet could be considered.

Knowledge about attacks is continuously increasing, but it still practically insufficient to establish such taxonomy that satisfies all the aforementioned requirements. In this regard, we follow a pragmatic approach to do so and assign it a moderate priority in our interest. On the other hand, our classification is fully supporting the following requirements.

Evaluation-related requirements: In addition to the general requirements, we can identify two more requirements that are important from the evaluation perspective:

1. It should have a complementary scheme for attack selection because multidimensional classifications are more complicated and usually have thousands of classes. Therefore, a classification scheme should be complemented by a clear approach for wise selection of attacks.
2. It should consider attack generation aspects: It should be kept in mind that attacks are classified and consequently test cases are selected in order to be generated during the evaluation process.

4.2 Suggesting New Classification

As stated previously, the purpose of this classification is to be used in the evaluation and testing of IDS. Therefore, attacks are viewed from the perspective of the IDS itself. We examined carefully the attributes mentioned in section 3 to determine which attributes are significant. Issues that are invisible for IDS or meaningless for it should be discarded. For example, dimensions such as *attacker's objective (intention)* will not be treated anymore within this classification since it is both hard and useless to reveal attackers intention. Beside that, we see any attack attempt or intrusion as a serious threat, whatever the objectives behind. Similarly, both the *type* and *detection technique* dimensions do not provide precise, clear cut categories.

While the *results* and *security property* (security threat) dimensions give an indication about the expected damage, it is out of the IDS scope according to the assumptions stated below. It can be investigated later by correlation and forensic tools. Furthermore, once attackers have hands over your system (especially if they managed to have root/administrative access), they can do what they like; they can steal, modify or destroy information and hence having a serious threat to the confidentiality, the integrity and the availability respectively.

Based on the analysis that we made in the previous sections and regarding the stated requirements, we have adopted a new classification inspired from the previous classifications [7], [12], [13], [14], [16].

Our taxonomy relies on two main assumptions. First, we define the task of an IDS as "to detect and to identify any *attack* or *intrusion* attempt, whether the attack was successful or not". The second assumption is that IDS is concerned mainly with atomic attacks. Composite and multi-stage attacks could be detected but in terms of individual attacks that comprise it. The correlation between alerts corresponding to the atomic attacks is supposed to signal a composite one or a scenario of attack.

Figure 1 shows our five-dimension taxonomy. The dimensions are selected carefully to cover attack manifestations, sources and origins. The dimensions are:

1. *firing source* that indicates the place from which the attacks are launched. It has two distinct classes: remote and local. This will determine the place from which an attack test case will be launched. It can help to decide which the placement and type of IDS is appropriate (e.g., which network segment, host-based or network-based). It is also important to evaluate the capacity of the evaluated IDS to detect remote as well as local attacks.
2. *privilege escalation*: regarding whether the attack results in promoting the access level. The distinct class *root* means that the attacker has gained a root/administration access. The distinct class *user* implies that the access level gained after the attack is a user access. *System is* the third distinct class for attacks that enables the execution of processes with system rights. The distinct class *none* covers attacks that do not need or do not result in any access to the system. This includes most of remote DoS attacks and reconnaissance or scanning attacks.
3. *vulnerability* dimension*:* was particularly added to express the relationship between attacks and vulnerability databases/dictionary and to precise the exploited vulnerability. It can point to the specific vulnerability that is exploited by the attacks that belong to this class. But for now, we precise only whether the vulnerability is due to configuration or design/ implementation flaws.

4. the *carrier* dimension explains the means by which the attack was carried out: either via network traffic or through an action performed directly on the machine and does not appear on the network interface.

5. the last dimension is the *targeted object.* Attackers may target the memory, the operating system, the network stack, a file system object or a process (which represent the distinct classes of the target dimension).

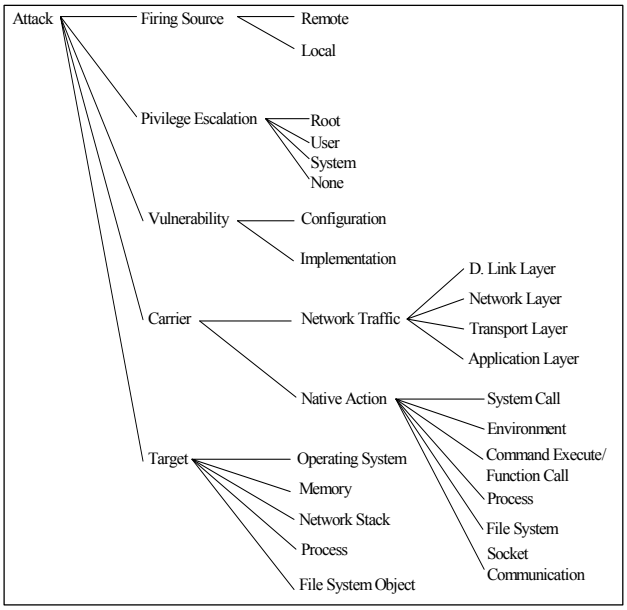

Figure 1. The suggested new taxonomy

Our taxonomy do not focus only on the observable characteristics of attacks like did the defense-centric [15] and the IDS-centric [16] taxonomies. The added value of this taxonomy is that it allows the classification of attacks regarding their characteristics observable by IDS while keeping the eye on the operational issues important for administrators. For example, the severity of attacks is reflected implicitly by privilege escalation dimension. Also, the source of danger (i.e., the firing source and the vulnerability types) could suggest how the danger could be alleviated by which counter measure (e.g., modify firewall rules to block a remote source or search a missed patch). Moreover, it does not ignore the evaluators' needs where it provides essential information for the generation of attacks and the analysis of test data. For example, the firing source dimension gives an idea about the place from which an attack should be generated, and the vulnerability dimension tells whether a particular configuration should be set/unset.

Ideally, an IDS should behave (i.,e., detect, undetect) in the same way against attacks of the same class. Thus, it will be sufficient to include a single attack from each class in the test case suite. To check this "strong" assumption, two cases can be distinguished. First: assuming that our classification is perfect: when we inject attacks of the same class (i.e., they have the same attributes, manifestations, etc.), the IDS will ideally behave similarly for all attack instances. Otherwise, if the IDS behave differently, we can conclude that the IDS has a problem of implementation and/or configuration. Consider signature-based IDS, by example, it may lack the corresponding signature of some attacks. The second case: assuming that neither the IDS nor the classification are perfect. As this is likely the case, we should search a compromise. To ensure the representativity of attacks in the test suite, we need to extend it with several instances of attacks of the same type and then make statistics on the detection /non-detection results. This will increase the number of test cases but at the same time it will enhance the quality and certainty of results.

Finally, there might be some empty classes, but we do not consider it as a limitation. Contrarily, this may demonstrates the extensibility of the classification as future attacks can fit into the empty classes.

5. A SCHEME FOR TEST-CASE SELECTION

Having presented the evaluation-oriented classification, how could it be employed for the evaluation and test of IDS? In this section we present an approach by which evaluators can select relevant test cases.

5.1 Classification Tree Method

The classification-tree method (CTM) was developed by Grotchamann and Grimm in [17]. By means of the CTM, the input domain of a test object is regarded under various aspects that are assessed to be relevant for the test. For each aspect, disjoint and complete classifications are formed. Classes resulting from these classifications may be further classified. The stepwise partition of the input domain by means of classifications is represented graphically in the form of a tree. Subsequently, test cases are formed by combining classes of different dimensions.

To construct the test-cases, a grid is drawn below the tree. The columns of the grid result from vertical lines that correspond to the leaves of the classification tree. A tester can construct a test case by selecting a single child class of each top-level classification. Each row of the grid indicates a distinct category of test case. However, not all test cases are legal or valid. Therefore, the tester should identify all valid test cases and eliminate invalid ones. This often could be done by applying the constraints stated explicitly or implicitly in system specifications.

A major advantage of the classification-tree method is that it turns test case selection and generation into a systematic process and making it easy to handle. Moreover, the systematic generation and analysis of test cases prevents the overlook that might occur for some areas of input. Thanks to its graphical representation, it allows the visualization of ideas and could be a good mean of communication between testers and developers.

In order to generate the possible test cases we used a tool called CTE (Classification Tree Editor) [4] which enables the automatic generation of test cases.

5.2 Generation of Attack Test Cases Using CTM

Given the attack classification tree and using the classification tree method (CTM), the CTE tool can produce all the possible combinations of the distinct subclasses from all the dimensions.

These combinations represent possible attack test cases.

The number of combinations may be in the range of thousands (more precisely 1920 test cases compared to 9600 in [16]) and exhaustively covers the attack space. However, the test cases can be reduced, regrouped and reordered to get only relevant test cases by applying constraints or generation rules in CTE.

The syntax for expressing the constraints within the program CTE is straight forward. In addition to the dimension or the attribute name, it uses the logic operators (AND (*), OR (+) and NOT) and the parentheses. For example, the following generation rule (i.e., constraint):

*Remote * (root +system) * configuration vul * Network traffic * (FS object + OS)*

will result in 16 test case categories, which represent remote attacks that provide root or system access by exploiting configuration vulnerability and could be observed in network traffic targeting the files system or the operating system (see appendix A).

6. CONCLUSION AND FUTURE WORK

This paper argued that an evaluation oriented classification of attacks is needed. We demonstrated that the existing classifications and taxonomies do not match all the needs of IDS evaluation and testing. To fill this gap, we proposed a new categorization scheme to be used by IDS evaluators and by network administrators to assess and test their IDS. Based on this classification and using the classification tree method, we introduced an approach to wisely select relevant attack test cases. Therefore, attack selection for IDS evaluation is no longer random or done blindly from the few attacks at hand. It can now be done with respect to the whole attack space. The next step is to classify the existing attacks and exploits in order to populate the test cases by real attacks. Then, we will proceed toward our ultimate goal as this enables the unbiased evaluations of Intrusion Detection Systems. This will be the subject of our next research.

7. REFERENCES

[1] John McHugh, Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory, *ACM Trans. on Information and System Security,* Vol. 3, No. 4, pp. 262-294, Nov. 2000.

[2] P. Mell, V. Hu, R. Lippmann, J. Haines, M. Zissman, An Overview of Issues in Testing Intrusion Detection Systems, *NISTIR 7007*, National Institute of Standards and Technology, August 2003.

[3] M. Gad El Rab, A. Abou El Kalam, Testing Intrusion Detection Systems: An Engineered Approach, *IASTED International Conference on Software Engineering and Applications, Nov. 2006.*

[4] Classification Tree Editor CTE: http://systematic-testing.com

[5] D. Lough, A Taxonomy of Computer Attacks with Applications to Wireless Networks, *PhD thesis*, Virginia Polytechnic Institute and State University, 2001.

[6] M. Bishop, Vulnerabilities Analysis, International Symposium on Recent Advances in Intrusion Detection (RAID'99), 1999.

[7] U. Lindqvist and E. Jonsson, How to Systematically Classify Computer Security Intrusions, *IEEE Security and Privacy*, pp 154–163, 1997.

[8] P. G. Neumann and D. B. Parker, A Summary of Computer Misuse Techniques, presented at 12th National Computer Security Conference, Baltimore, MD, 1989, pp. 396—407.

[9] S. Kumar, Classification and Detection of Computer Intrusions, *PhD thesis*, Purdue, 1995.

[10] D. J. Webar, A Taxonomy of Computer Intrusions, *Master Thesis*, Massachusetts Institute of Technology, June 1998.

[11] R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, and K. Das, Analysis and Results of the 1999 DARPA Off-Line Intrusion Detection Evaluation, *Third Intl. Workshop on Recent Advances in Intrusion Detection (RAID2000)*, Toulouse, France 2000.

[12] K. Kendall, A database of computer attacks for the evaluation of intrusion detection systems, Master's Thesis. Massachusetts Institute of Technology, MA, June 1999.

[13] John D. Howard. An Analysis of Security Incidents on The Internet 1989-1995, *PhD thesis*, Carnegie Mellon University, 1997.

[14] Simon Hansmann, A Taxonomy of Network and Computer Attacks, *Diplom Thesis*, University of Canterbury, New Zealand, Nov. 2003.

[15] Kevin S Killourhy; Roy A. Maxion, and Kymie M. C. Tan, A Defense-Centric Taxonomy Based on Attack Manifestations, *In International Conference on Dependable Systems & Networks (DSN-04)*, pp. 102-111, Italy, 28 Jun-01 Jul 2004.

[16] Dominique Alessandri, Attack-Class-Based Analysis of Intrusion Detection Systems, *Ph.D. Thesis*, Newcastle upon Tyne, UK: University of Newcastle upon Tyne, 2004.

[17] M. Grochtmann, J. Wegener, K. Grimm, Test case design using classification trees and the classification-tree Editor CTE, *In: Proceedings of the 8th International Software Quality Week*, pp. 1-11, 1995.

[18] Nicholas J. Puketza, Kui Zhang, Mandy Chung, Biswanath Mukherjee, and Ronald A. Olsson, A Methodology for Testing Intrusion Detection Systems, IEEE Trans. on Software Engineering, vol. 22, pp. 719--29, October 1996.

APPENDIX-A**: Test cases produced by the CTE Tool**