Network Intrusion Detection by Applying Various Testing Techniques

Kurundkar G.D[1] Naik N.A.[2]
Dr. Khamitkar S.D.[3] Dr.Kalyankar N.V.[1]

GJCST Computing Classification
B.8, D.4.6 & K.6.5

Abstract- **Intrusion detection is the process of identifying unauthorized usage of a Network. It is important task to protect system from unauthorized users on the network and it is very difficult task to protect system from Intruder. For protecting such network-based system from unauthorized users, we have to implement some prevention measures. With the help of prevention measures, we can give protection to our network. It is necessary to test that prevention measures are working correctly or not, by applying various testing methods on Network. This paper describes methods of Software testing which are applicable for security measures and try to describe approach for on-line attack identification and uses similarity rules for generalization of attack signatures. We can immediately protect the system from many variants of previously known attacks. The architecture uses the comparison of outputs from diverse applications to provide a significant and novel intrusion detection capability. It also initiates attack diagnosis and blocking unauthorized user. The methodology consists of using scripts to generate both background traffic and intrusions with provisions for multiple, interleaved streams of activity.**

Keywords- Intruder; hacker; cracker; Intrusion detection; anomaly detection; verification; validation.

I INTRODUCTION

The ideal goal of any security expert in making a system secure enough to avoid any possible intrusion attempt is to provide the testing methods which are applicable to the known attacks and think about the future attacks which can be harmful to the network. Now a days the Network Protocols, Standalone systems or the single program is much more complex than the previous one. Time required for protecting the system from the intruder is much more than expected. It practically impossible to create a totally secure system. The use of firewall now a days is not enough for protecting your network from the outsiders. In fact, the defense layer itself can contain errors, or it can be bypass with the intelligent internal users.

Manuscript received "Thu, Jan 21, 2010 at 2:04 PM GMT"

Mr. G.D.Kurundkar is working as Lecturer with Dept. of Computer Science S.G.B. College Purna
E-Mail: gaju_k_2001@yahoo.com
Mr. N.A.Naik is working as lecturer with Dept. of Computer Science and IT , Yeshwant College Nanded
E-Mail: nitinnaik@live.com
Dr. S.D. Khamitkar is working as Director, School of Computational Sciences S.R.T.M.University , Nanded.
E-Mail: s_khamitkar@yahoo.comDr. N.V.Kalyankar is working as Principal, Yeshwant College Nanded.

It is not an intelligent selection to base the whole security process only on prevention techniques. A more reliable process must also take into account how to detect and identify failures. A security process that relies only on prevention without any detection mechanism can be very unstable, since it can create a false sense of protection.
A good prevention is still the first line of any security. Datasets used to test IDS can be described by two main characteristics: the type of intrusion detection technology used (signature-based or anomaly-based) and the location of the IDS (host-based or network-based). Network-based IDS are significantly different from those needed by an anomaly host-based IDS. [1] An attacker can explore a series of potential vulnerabilities until successful, and when successful, he can repeatedly use the exploit that provided success against the same system or other systems with the same vulnerability.[2] Malicious characteristics of computer and network attacks, additional innovative techniques are required. One area of research, which has been applied to this domain by others, is the study of Byzantine faults.[3] The methodology includes techniques and approaches that we adapted from the general field of software testing. Neither the methodology nor the software platform is specific to particular IDS, they can be used to test and compare several different Intrusion Detections. IDS developers can use the platform and methodology to supplement their own approaches to testing. [4] Classification of ID techniques are misuse detection, anomaly detection, and a specification-based approach.[5]. The testing technique is based on an automated mechanism to generate a large number of variations of an exploit by applying mutant operators to an exploit template. The mutant exploits are then run against a victim system where the vulnerable applications and/or operating systems are installed. The attacks are analyzed by a network- based intrusion detection system.[14]

II INTRUDERS

Broadly saying the intruder is an intelligent compuser who is good enough to break into others system or present system to do some unauthorized things. There are many ways of manipulating, illegally updating, or damaging IT Networks and of preparing an attack on IT Network
Intruders are classified into three groups. .[6]

A. Masqueraders

Masqueraders are the persons from the outside who have not right to log into system to use it but they are smart persons who crack the password of the authorized users and then penetrates into system.

B. Misfeasor

These users are internal users from the network where they are blocked to gain access of the particular area but as they are also smart compusers they anyhow access the things they want from the system.

C. Clandestine

An individual who have supervisory power to control the system and have power to turn off audit control or to suppress audit collections.(It is likely to be outsider/insider.)[7]

III INTRUSION DETECTION MECHANISM

Intruders are unauthorized user and tries to access authorized system. They try to access protected information such as password. Typically, s system maintains a file that associated with a password with each authorized user.

A. Statistical Anomaly Detection

It involves the collection of data relating to the behavior of legitimate users over a period of time.

B. Threshold Detection

This approach involves defining thresholds independent of users, for the frequency of occurrence of various events.

C. Profile Based

A profile of the activity of each user is developed and used to detect changes in the behavior of individual accounts.

D. Rule Based Detection

It is a set of rules that can be used to decide that a given behavior is that of an intruder.

E. Anomaly Detection

Rules ate developed to detect deviations from previous usage patterns.

F. Penetration Identification

An expert system approach that searches for suspicious behavior

G. Audit Record

It is a fundamental tool for intrusion detection, some record of ongoing activity by users must be maintained, to an intrusion detection system.

H. Learning and Detection

Maintaining log based analysis detecting abnormal activities of unauthorized users. Experimenting with unsupervised learning algorithms and detection mechanisms is made easy. We are trying the K-Means algorithms, as well known clustering algorithm .Log analyzer can handle logs in other formats gathered from different running environments. The sources of audit data can be a keyboard input, command-based logs, application-based logs, and network traffic. According to the type of the audit data collected, we can classify the IDS into two categories [11]

IV TESTING

Testing involves the protection of system from known and unknown attacks. The conditions should include both normal and abnormal conditions. Testing should intentionally attempt to make things go wrong to determine if things happen when they shouldn't or things don't happen when they should. Testing is a process of quality control using a group of defined methods and evaluation criteria, together with guidelines for their use.[8].. An important defect is one that from the customer’s perspective affects the usability or functionality of the application. The quality assurance aspect of testing is the degree to which the developers followed. Corporate standard processes or best practices are not responsibility of the testing team. The testing cannot improve quality but trying to find defect in developed application. They can only measure it, although it can be argued that doing things like designing tests before coding begins will improve quality because the coders can then use that information while thinking about their designs and during coding and debugging. Testing has three main purposes: verification, validation, and defect finding..The *verification* process confirms that the software meets its technical specifications..The *validation* process confirms that the software meets the requirements described by user and A *defect* is a variance between the expected and actual result. The defect’s ultimate source may be traced to a fault introduced in the specification, design, or development (coding) phase. The difficulty in measuring the detection rate is that the success of IDS is largely dependent upon the set of attacks used during the test. In addition, the probability of detection varies with the false positive rate, and IDS can be configured or tuned to favor the ability either to detect attacks or to minimize false positives [9]
Testers must not only have good development skills—testing often requires a great deal of coding but also be knowledgeable in formal languages, graph theory, and algorithms. Indeed, creative testers have brought many related computing disciplines to bear on testing problems, often with impressive results.[13]

A. *V-Model Of Application Testing*

V-Model of testing incorporates testing into the entire software as well as application development life cycle. In a diagram of the V-Model, the V proceeds down and then up, from left to right depicting the basic sequence of development and testing activities. The model highlights the existence of different levels of testing and depicts the way each relates to a different development phase. Like any model, the V-Model has detractors and possibly has deficiencies and alternatives but it clearly illustrates that testing can and should start at the very beginning of the assignment. In the requirements gathering stage the business requirements can verify and validate the business case used to justify the assignment. The business requirements are also used to guide the user acceptance testing. The model illustrates how each subsequent phase should verify and validate work done in the previous phase, and how work done during development is used to guide the individual testing phases. This interconnectedness lets us identify important errors, omissions, and other problems before they can do serious harm. Application testing begins with Unit Testing, and in the section titled "Types of Tests" we will discuss each of these test phases in more detail. [10]

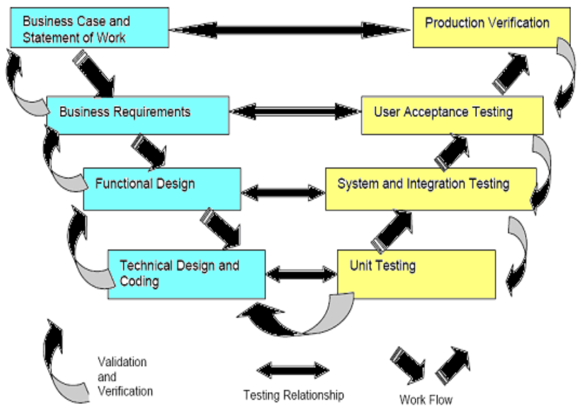

Fig.1.1 Show. The V-model of software/Application

Types of Software/Application Tests :

Phase	**Guiding Document**	**Test Type**
Development Phase	Technical Design	Unit Testing
System and Integration Phase	Function design	System testing and integration Testing
User Acceptance Phase	Business Requirements	User Acceptance Testing
Implementation phase	Business Case	Product Verification Testing
Regression Testing applies to all phases		

Table 1.1. The V-Model of testing identifies five software testing phases, each associated with type of test.

B. *Various Testing Techniques*

i. Unit Testing

A series of stand-alone tests are conducted during Unit Testing. Each test examines an individual component that is new or has been modified. A unit test is also called a module test because it tests the individual units of code that comprise the application. It based on the technical design documents Unit tests focus on functionality and reliability, and the entry and exit criteria can be the same for each module or specific to a particular module. If a defect is discovered during a unit test, the severity of the defect will dictate whether it will be fixed before the module is approved.

ii. System Testing

System testing tests all components and modules that are new, changed, affected by a change, or needed to form the complete application. The system test may require attachment of other systems & it should be minimized as much as possible to reduce the risk of externally induced problems. For example, the system test for a new web interface that collects user input for addition to a database. System testing requires many test runs because it entails feature-by-feature validation of behavior using a wide range of both normal and erroneous test inputs and data. The Test Plan is critical here because it contains descriptions of the test cases, the sequence in which the tests must be executed, and the documentation needed to be collected in each run. When an error or defect is discovered, previously executed system tests must be rerun after the repair is made to make sure that the modifications did not cause other problems.

iii. Integration Testing

Integration testing examines all the components and modules that are new, changed, affected by a change, or needed to form a complete system. Where system testing tries to minimize outside factors, integration testing requires involvement of other systems and interfaces with other applications, including those owned by an outside vendor, external partners, or the customer. Integration testing also differs from system testing in that when a defect is discovered, not all previously executed tests have to be rerun after the repair is made.
Integration testing has a number of sub-types of tests that may or may not be used, depending on the application being tested or expected usage patterns.

iv. Compatibility Testing

Compatibility tests insures that the application works with differently configured systems based on what the users have or may have. When testing a web interface, this means testing for compatibility with different browsers and connection speeds.

v. Performance Testing

Performance tests are used to estimate and understand the application's scalability when, for example, additional users are added or the volume of data increases. This is particularly important for identifying bottlenecks in high usage applications. For data retrieval application, reviewing the performance pattern may show that a change needs to be made in a stored SQL procedure or that an index should be added to the database design.

vi. Stress Testing

Stress Testing is performance testing at higher than normal simulated loads. Stressing runs the system or application beyond the limits of its specified requirements to determine the load under which it fails and how it fails. A continuing performance slow-down leading to a non-catastrophic system halt is the desired result, but if the system will suddenly crash and burn it's important to know the point where that will happen. This test is possibly the most important test for mission-critical systems.

vii. Load Testing

Load tests are the opposite of stress tests. They test the capability of the application to function properly under expected normal production conditions and measure the response times for critical transactions or processes to determine if they are within limits specified in the business requirements and design documents or that they meet Service Level Agreements. For database applications, load testing must be executed on a current production-size database.

viii. User Acceptance Testing

User Acceptance Testing is also called Beta testing, application testing, and end-user testing. Software vendors often make extensive use of Beta testing. Regardless of their best efforts, though, they probably don't find all the flaws in the application.

ix. Product Verification Testing

Production verification testing is a final opportunity to determine if the software is ready for release. Its purpose is to simulate the production cutover as closely as possible and for a period of time simulate real business activity. it should identify anomalies or unexpected changes to existing processes introduced by the new application. The application should be completely removed from the test environment and then completely reinstalled exactly as it will be in the production implementation. Unlike parallel testing in which the old and new systems are run side-by-side, mock processing may not provide accurate data handling results due to limitations of the testing database or the source data.

x. Regression Testing

Regression testing is also known as validation testing and provides a consistent, repeatable validation of each change to an application under development or being modified. Each time a defect is fixed, the potential exists to inadvertently introduce new errors, problems, and defects. Regression testing is the probably selective retesting of an application or system that has been modified to insure that no previously working components, functions, or features fail as a result of the repairs. Regression testing is conducted in parallel with other tests and can be viewed as a quality control tool to ensure that the newly modified code still complies with its specified requirements and that unmodified code has not been affected by the change. It is important to understand that regression testing does not test that a specific defect has been fixed. Regression testing tests that the rest of the application up to the point or repair was not adversely affected by the fix.

xi. Testing Using Disinfected Traffic Or Logs

In this approach, real background traffic is saved and then sanitized to remove any sensitive data. Then, intrusions are injected to the cleaned data. The main advantage of this approach is that tests can be publicly distributed and tests are repeatable. On the other hand, there are some difficulties facing this approach such as cleaning may end by removing needed data and cleaning could fail, thus leading to have some sensitive data disclosed to others. [12]

xii. Testing Using Simulated Traffic

Approach includes background traffic that is generated by complex traffic generators. The advantages of this approach are as follows:

• The data can be publicly distributed, since it does not contain any sensitive data.

• It could be repeatable, since the testing team can replay the same background data or using the simulator, testers can regenerate previous background traffic.

• It could be used to test both: possibility of detection and false-positives.

V CONCLUSION

The paper describes various software as well as application testing Techniques, which will adopt for purpose of testing IDS, We are trying to present the details of the methodology. Software testing is a critical element in the software development life cycle and has the potential to save time and money by identifying problems early and to improve customer satisfaction by delivering a more defect-free product We will perform results from searching various activities of unauthorized behavior patterns by log base experiments on the Network Security Monitor(NSM)for the testing of IDS The platform consist of windows, and in this research paper, we are trying to maintain unexpected activities of unauthorized users, from regular users of system by maintain log file of person behavior. Without adequate testing, however, there is a greater risk that an application will inadequately deliver what was expected by the business users or that the final product will have problems such that users will eventually abandon it out of frustration. . More formal, rigorous testing will go far to reducing the risk that either of these scenarios occurs.

VI References

1) Frédéric Massicotte, François Gagnon & Yvan Labiche, Automatic Evaluation of Intrusion Detection Systems.
2) James C. Reynolds, James Just, Larry Clough, Ryan Maglich ,*On-Line Intrusion Detection and Attack Prevention Using Diversity, Generate-and-Test, and Generalization,* Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS'03) 0-7695-1874-5/03 2002 IEEE.
3) M. Castro and B. Liskov, "Practical Byzantine Fault Tolerance," Proceedings of the 3rd Symposium on Operating System Design and Implementation, New Orleans LA, February 1999.
4) Nicholas Puketza,Mandy Chung,Ronald A.Olsson & Biswanath Mukharji, *A Software Platform for Testing Intrusion Detection Systems* , May 1997. University of California.
5) Krzysztof M. Brzeziski *On Common Meta-Linguistic Aspects of Intrusion Detection and Testing.* Journal of Information Assurance and Security 2 (2007) 167{178) .
6) Giovanni Vigna, William Robertson & Davide Balzarotti. *Testing Network based Intrusion Detection Signatures Using Mutant Exploits, CCS'04,* October 25–29, 2003, Washington, DC, USA.
7) Williams Stalling *Computer System Security,*3rd Edition.
8) D Hannan, M Ward An alternative approach to software testing to enable SimShip for the localisation market, using Shadow™ BY K Arthur.
9) Peter Mell,Vincent Hu, Richard Lippmann, Josh Haines, Marc Zissman. Richard Lippmann, Josh Haines, Marc Zissman . *An Overview of Issues in Testing Intrusion Detection Systems.* National Institute of Standards and Technology.
10) John E. Bentley, Software Testing Fundamentals Concepts, Roles, and Terminology Wachovia Bank, Charlotte NC, Paper 141-30.
11) A. Mishra, K. Nadkarni, A. Patcha, and V. Tech, "Intrusion Detection in Wireless Ad Hoc Networks", IEEE Wireless Communication, IEEE press, 2004.
12) Hadi Otrok, Joey Paquet, Mourad Debbabi and Prabir Bhattacharya.*Testing Intrusion Detection Systems in MANET*: A Comprehensive Study Fifth Annual Conference on Communication Networks and Services Research(CNSR'07) 0-7695-2835-X/07 2007
13) James A. Whittaker**,** What Is Software Testing? And Why Is It So Hard? Florida Institute of Technology J a n u a r y / F e b r u a r y 2 0 0 0 IEEE SOFTWARE
14) Mr. Nitin A. Naik, Mr. Gajanan D. Kurundkar, Dr. Santosh D. Khamitkar, Dr. Namdeo V. Kalyankar,Penetration Testing: A Roadmap to Network Security, Journal Of Computing, Volume 1, Issue 1, December 2009