PROFICIENT AND SECURE PATH FOR DATA COMMUNICATION USING HONEY POT METHOD AND ENCRYPTED IP ADDRESS WITH RANDOMIZING IN CRYPTOGRAPHICAL ALGORITHM

R. James Marianathan & Dr. A. Arul Lawrence Selvakumar

Abstract:

Core of the organizations data is available in the back end server. Securing the data from unauthorized people and transmitting the data to the authorized person through a secure route is the aim of the organization. A protective model for back-end servers is accomplished by the novel combination of honeypot deployment, honeypot masquerading, and strict authentication for back-end server access. In our approach, a solution that has been catching on in the network security and computer incident response environment is to employ “Honey Pots.” Honey Pots, also known as deception systems, honey-pots or tar-pits, are phony components setup to entice unauthorized users by presenting numerous system vulnerabilities, while attempting to restrict unauthorized access to network information systems. These features include suspicious event alerts to a management workstation for visual and audible notification, the ability to capture the unauthorized user’s keystrokes and send it to a remote server, various customized logging and bogus system files and information to have the unauthorized user waste time as the security administrator prepares a counter measure.

Honeypot

A honeypot is a decoy computer system designed to look like a legitimate system an intruder will want to break into while, unbeknownst to the intruder, they are being covertly observed. Honeypots are effective precisely because attackers do not know if they are there and where they will be. However, honeypots are also a controversial technique; they essential bait and capture intruders skirting the fine line between keeping attackers out of a network versus inviting them in. Little legal precedent has been established. Some see them as unfair entrapment tools while others see them as an effective data gathering and deterrence mechanism.

A honeypot is a program that takes the appearance of an attractive service, set of services, an entire operating system, or even an entire network, but is in reality a tightly sealed compartment built to lure and contain an attacker (a sandbox where intruders cannot harm production systems or data) – effectively shunting an intruder safely from production systems for covert analysis. Like a hidden surveillance camera, a honeypot monitors and logs every action an attacker makes including access attempts, keystrokes, files accessed and modified, and processes executed.

We are implementing in our project in Java, based on client and server technology. We are also deployed the cryptographic procedure for maintains the security. In this concept, the alternate path selecting is main factor for eliminate the intruder in the network and also utilize the network in better manner. Honey Pots are considered "set and forget" IDS sensors, because they are composed of a single system or network of devices whose sole purpose is to capture unauthorized activity.

As a result, they are able to download free tools that will scan many different networks looking for those easy-open entry points. Employing Honey Pots takes advantage of these traditional issues and uses it for its enticing benefit. They are constructed to sting hackers, not just keep them out. Honeypot technology has concentrated on emulating vulnerable operating systems, applications and services to capture attackers and analyses their methodologies. In recent years, the surge of Internet worms has created a new use for this technology. Many experts predict that attacks targeting Windows file sharing will increase significantly. Current Intrusion detection system have the ability to detect intruders once the attack has already occurred. This often gives little information on what has actually been done. Honeypots, while being a relatively new technology, can generate very good data on attacks. All traffic occurring on and around a properly set up honeypots is logged. Current weakness in misuse intruder detection system include the fact that new attacks are not detected until someone has generated a rule or signature that pick up that specific attack and the most attack, need only be slightly altered in order to bypass existing rules. Honeypots used to detect and report on new attacks. It may be automated, saving time and catching new attacks as they happen.

A honeypot is a fictitious vulnerable IT System deployed to be probed, attacked, exploited and compromised. Generally it consists of a computer, data or a network site that appears to be a part of a network but which is actually isolated and protected, and that which seems to contain information or a resource that would be of value to attackers. An example of a honeypot is a system used to simulate one or more network services that you designate on your computer's ports. An attacker assumes that you are running vulnerable services that can be used to break into the machine. This kind of honeypot can be used to log access attempts to those ports including the attacker's keystrokes.

Implementation Issues

Several issues arise with deployment of this protocol in practice. While the protocol is light to each client, inducing only slight overhead to assign permissions and forward authenticated client session information to the router Rm, this overhead across all clients on the implementation in the ns-2 network simulator was determined to be 10% of the traffic load required to balance this overhead. A second issues is the deviation of system resources to the handling of honeypot communication. It is possible for an attacker to flood the honeypot with request messages from different IP address sources. It is the duty of Rm instead of H to update the AS with misbehaving traffic alerts should the messages overflow the H queue. The flexibility provided by the use of the masquerading router

ISSN: 2278-5183
www.ijcdsonline.com

International Journal of Computers and Distributed Systems
Vol. No.2, Issue 1, December 2012

21 | a g e  w w w . c i r w o r l d . c o m
provides a greater possibility for deployment and success of the honeypot as a decoy device as well as better insulation for the true back-end server from malicious traffic. Typically, a honeypot involves simulation of either a network server or port containing valuable or sensitive information that would be attractive to an attacker. Honeypots are not intended for use by legitimate network traffic, so any traffic received can immediately be recognized as malicious and further action can then be taken. Such honeypots can interact with the attacker to a pre-defined extent to gather information on the types of attacks being deployed against the network.

The power of these honeypots lies in their exploitation and ability to gather information on attacks. Static honeypots have the weakness of detectable location which can allow an attacker to simply avoid the honeypot and continue to a potentially significant target. Alternate forms of honeypots have also been proposed, ranging from honeynets to the virtual honeypot scheme. Honeynets are simply a collection of honeypots deployed throughout a network to increase the likelihood of an attacker reaching one on an unfocused search. In a similar idea to honeypots, the concept of a honeytoken has also been proposed as a trap for malicious clients; honeytokens are programs or files that attract the attention of malicious clients on a smaller scale than honeypots, taking the form of enticing links or e-mail messages. This is a lightweight system that simulates multiple low-interaction honeypots on a single network card, capable of simulating the stack of multiple operating systems, a key component by which attackers can determine the signature of a system and therefore continue to press attacks which can later be analyzed. However, the low-interaction of each simulated honeypot can be detected and avoided just as a physical honeypot.

The shadow honeypot is a means of detecting and deterring anomalous traffic by rerouting it to a state limited production environment that can be rolled-back. The design for this system is based on the possibility that some traffic will be mislabeled as anomalous and can therefore be serviced transparently provided it does not exhibit malicious behavior. While this system is effective for combining the honeypot decoy system with anomaly detection, it requires a great deal of configuration and adaptation based on network traffic patterns. It also requires the use of a separate controlled simulation of the production environment, which may be an expensive undertaking depending upon the complexity of back-end services.

Two requirements provide protection against DoS attacks in this scheme; First all traffic is dropped when a server switches from active to inactive (acting as a honeypot) and second any traffic that hits an inactive honeypot is filtered from further communication within the network [10]. There are several limitations to this scheme. First, the back-end servers must be able to seamlessly switch states with each other, meaning this type of approach is only suitable for systems without a large possibility for state change and easy TCP/IP migration from server to server. In the absence of this, the complete server image would have to be migrated to the new server connection, inducing tremendous client delay. Secondly, there is a significant waste of resources by allowing active servers to act as decoys. Third, this scheme actually violates the definition of honeypot usage, which mandates the uninterrupted operation of the network with the honeypot attached.

There is additional risk in the possible compromise of an active back-end server while it is acting as a honeypot. While this protocol provides for the periodic cleanup of source code on the back-end server, a complete server re-image would be required at each switch from the inactive state to the active state, termed an epoch; this would again induce a tremendous amount of client delay. This paper does suggest the idea of logical roaming without further expansion; it is upon this idea that the proposed system herein is developed.

Uses of Honeypots

- **Preventing Attacks:** Honeypots can prevent attacks in several ways. For example, Honeypots can be used to prevent automated attacks such as those launched by worms or auto-rooters. These attacks are based on tools which scan the entire networks looking for vulnerable systems. If any such systems are found they will launch the attacks and take over the systems. Honeypots can be used in such cases to defend such attacks by slowing the scanning process, potentially even stopping it. This can be done by using a variety of TCP tricks, such as using a window size of zero which puts the attacker into holding pattern. Honeypots can also be used to protect organizations from Human attacks (Non-automated attacks). The concept is based on deception or deterrence. The main idea is to confuse the attackers, making them waste their time and resources interacting with honeypots. Meanwhile the organization is able to detect the attackers activity and has time to respond and stop it.

- **Detecting Attacks:** The main difficulty with other technologies like Intrusion Detection Systems (IDS) and system logs in detecting the attacks is that they log too much of data and because of large false positives they are unable to detect new attacks. Honeypots address these issue by reducing false positives by capturing only small data sets which are of high value. By detecting attacks the organization can be able to quickly react to them, stopping the damage.

- **Responding to Attacks:** The main problem with responding to attacks is that it is difficult to take the systems which are being compromised offline and analyse the data. Honeypots address this problem as they can be quickly be taken offline without impacting day-to-day business operations. Also because the only activity the honeypot captures is the unauthorized or malicious activity, this makes hacked honeypots much easier to analyze.

- **Using Honeypots for research purposes:** Honeypots are used to gain extensive information of threats. Organizations can use this information for a variety of purposes, including analyzing trends, identifying new tools or methods, identifying attackers and their communities, ensuring early warning and prediction, or understanding attackers motivation.

**Types of Honeypots**

There are two types of honeypots, low interaction and high interaction. The main difference between these two is the complexity and interaction they allow an attacker. Low Interaction Honeypots:- By emulating operating systems and other services, low interaction honeypots do not give attackers much control. The main advantage is their simplicity that allows easy deployment and maintenance and the low risk factor because they do not work with real production system. (Examples :- Honeyd, Spectr )

High interaction Honeypots :- They differ in that they involve real operating systems and applications. Here nothing is emulated. The advantage of these is that they give the attackers real systems to work with, you can capture a wide range of information and learn new techniques being used. Secondly, honeypots don't assume what action the attacker might take. By giving an open environment to work with, they can learn methods the attackers wouldn't normally use. One downside of these high interaction
systems is that because of their complexity they can be somewhat difficult to deploy. Even more concern is the possibility that an attacker could gain control of the honeypot. (Example: A real FTP server on a Linux system with full interaction.)

**Future Work**

The Duplications of files take too much time to avoid this router has been implemented. The Implementing FTP protocol we can download the image files also Giving options to user for choosing the file by browsing after enter into the original server or duplicate server and maintain the separate database for every user in the network. The network is designed with mesh topology. In future the network can be designed for various topologies apart from the mesh topology. Presently the time complexity is in the order of the square root of the existing clustering method. This can be improved to higher orders also. Various combination of algorithm can be used for this network design to attain higher performance. The algorithm whichever can be combined include the ant colony, simulated annealing, clustering algorithm etc.

7. Marco Dorigo and Luca Maria Gambardella, “Ant Colonies for the Traveling Salesman Problem”.
11. Zhongzhen Yang & Bin Yu Transportation College, Dalian Maritime University, Dalian, 116026, China & Chuanmin Cheng Civil Engineering Department, Dalian University of Technology, China, “A Parallel Ant Colony Algorithm for Bus Network Optimization”.