Implementation of Honeypots

Vishal Bankar, Suvarna Zinjurke
Students of Separtment of Computer Engineering,
Sau. Sundarbai Manik Adsul Poly Chas, Ahmednagar
kiranbankar11lic@gmail.com,
suvarnazinjurke58879@gmail.com

Akshay A. Somwanshi
Department of Computer Engineering
Sau. Sundarbai Manik Adsul Poly Chas, Ahmednagar
akshay9somwanshi@gmail.com

Abstract—**Providing security to the server data is the greatest challenge. Security activities range from keeping intruders out of the network or system, preventing the interception of information sent via the Internet and damage caused by computer viruses. Countermeasures are developed to detect or prevent attacks. Most of these measures are based on known facts, known attack patterns. Countermeasures such as firewalls and network intrusion detection systems are based on prevention, detection and reaction mechanism. But they are not able to gather much information about attackers. An Intrusion Detection System (IDS) is a device or software application that monitors network and system activities for malicious activities or policy violations. Then it produces reports to a Management station. Intrusion prevention is the process of performing intrusion detection and attempting to stop detected possible incidents.**
The main problem with current intrusion detection systems is high rate of false alarms. Use of honeypots provides effective solution to increase the security and reliability of the network. Honeypots are easy to use, capture the required information and mainly used by the corporate companies to secure their networks from the online hackers and unauthorized users.

I. INTRODUCTION

Public and private organizations transfer more of their information through the Internet. Today, attacker or intruder to the system is the biggest problem for the safety of the network. Criminals have more opportunity to gain access to sensitive information through the Web application. The first step in protection against online attacks is to understand the nature and tools of the attacks.

To provide security to server data, it is efficient to implement fake services using honeypot. Honeypot is nothing but a fake server that provides emulated services similar to the real services running on the actual server. So whenever attacker tries to attack actual server, attacker is redirected towards the fake server that is honeypot and eventually is trapped in the honeypot. Honeypot then gives the valuable information regarding the intruders. This information can be used to block the attacker and it can be used to take the legal actions against them [6].

To detect anomalous or inappropriate activities, already there are some methods such as IDS, Firewalls etc. But they have several limitations of anomaly detections such as high rate of false alarm, alerts generated does not contain sufficient detailed information for analysis etc.

Honeypot provides a platform by which online attacks can be investigated [7]. or client applications and browsers, malware collection, attack diversion etc.

II. EVOLUTION

The concept of honeypots was first described by Clifford Stoll in his book. This honeypot already used multiple virtualized systems hosted on a single hardware component. Since then, the development of sophisticated honeypots and honeynets has continued. In recent years, honeypots have become steadily more flexible. Honeyd can create a number of virtual hosts on a network and can be flexibly configured to run arbitrary services. Then redirection approach and a dynamic forensic system are used for specific investigations and to minimize false-positives. In this type of honeypot network traffic which seems to be anomalous is redirected to shadow servers and a forensic module collects useful information about executed attacks.

Then a dynamic honeypot system is proposed which adapts itself to the current network surroundings by passively observing network traffic [3]. This approach helps to automatically fit the honeypot seamlessly into the network in which it is located. This idea is improved by adapting honeypots dependent on other hosts in the network using active network port scans instead of passive traffic analysis. This way, the dynamic honeypot can autonomously integrate into a continuously changing computer network which is especially interesting for virtualized networks consisting of VMs and being subject to continuous changes. These approaches deal with the autonomous and dynamic integration of honeypots in constantly changing network surroundings.

III. CLASSIFICATION OF HONEYPOTS

A. Low-Involvement Honeypots

A low-involvement honeypot typically only provides certain fake services. Attackers can only scan and connect to several ports. On a low-involvement, honeypot there is no real operating system that an attacker can operate on. The information about the attackers and the risk is limited since the attacker's ability to interact with the honeypot is limited [4].

B. Medium Involvement Honeypots

A mid-involvement honeypot provides more to interact with, but still does not provide a real underlying operating system. The fake daemons are more sophisticated and have deeper knowledge about the specific services they provide. At the same moment, the risk increases. The probability that the

Proceedings of Second Shri Chhatrapati Shivaji Maharaj QIP Conference on Engineering Innovations
Organized by Shri Chhatrapati Shivaji Maharaj College of Engineering, Ahmednagar
In Association with Novateur Publications JournalNX-ISSN No: 2581-4230
February, 22ⁿᵈ and 23ʳᵈ, 2019

attacker can find a security hole or vulnerability is getting bigger because the complexity of the honeypot increases. A compromise of this system is still unlikely and certainly no goal as there are no security boundaries and logging mechanisms built for this kind of events. Through the higher level of interaction, more complex attacks are possible and can be logged and analysed. Generally speaking, the attacker gets a better illusion of a real operating system and has more possibilities to interact and probe the system. Developing a mid-involvement honeypot is complex and time consuming. Special care has to be taken for security checks as all developed fake daemons need to be as secure as possible. The developed versions should be more secure than their real counterparts, as this is the main reason to substitute these with fake variants. The knowledge for developing such a system is very high as each protocol and service has to be understood in detail [8].

C. High Involvement Honeypots

A high-involvement honeypots are far more complex than other type of honeypots. They involve the deployment of a real operating system and applications. This leads to a much higher risk as the complexity increases rapidly. At the same time, the possibilities to gather information, the possible attacks as well as the attractiveness increase a lot. By allowing the attackers to interact with real systems, the full extent of their behaviour can be studied and recorded. A high-involvement honeypot does offer such an environment. As soon as a hacker has gained access, his real work and therefore the interesting part begins. Unfortunately the attacker has to compromise the system to get this level of freedom. He will then have root rights on the system and can do everything at any moment on the system [9].

IV. EXISTING METHODOLOGY

Server computing environments are distributed in nature. Hence they can be easily targeted and exploited by the intruders. Intruders can pretend that they are the legitimate users and can use the cloud services maliciously. Providing security in a distributed system requires more than user authentication with passwords or digital certificates and confidentiality in data transmission [2].

IDS can be used to provide additional security measures for these environments by investing configurations, logs, network traffic, and user actions to identify typical attack behaviour. IDS must be able to monitor each and every node in cloud environment and this node must be able to alert other nodes in the environment. This type of communication requires compatibility between heterogeneous hosts, various communication mechanism, and permission control over system maintenance and environments. In cloud this feature is provided by the middleware, hence the IDS system is offered at middleware layer.

Fig. 1. Shows the architecture of cloud computing Intrusion Detection. Following are the components needed in construction of the system.

1] Event Auditor: Audit data that describes environment's state and the messages being exchanged is required in order to detect an intrusion. Event Auditor can monitor the data. It monitors message exchange between nodes as well as the middleware logging system. Audit data is sent to the IDS service core.
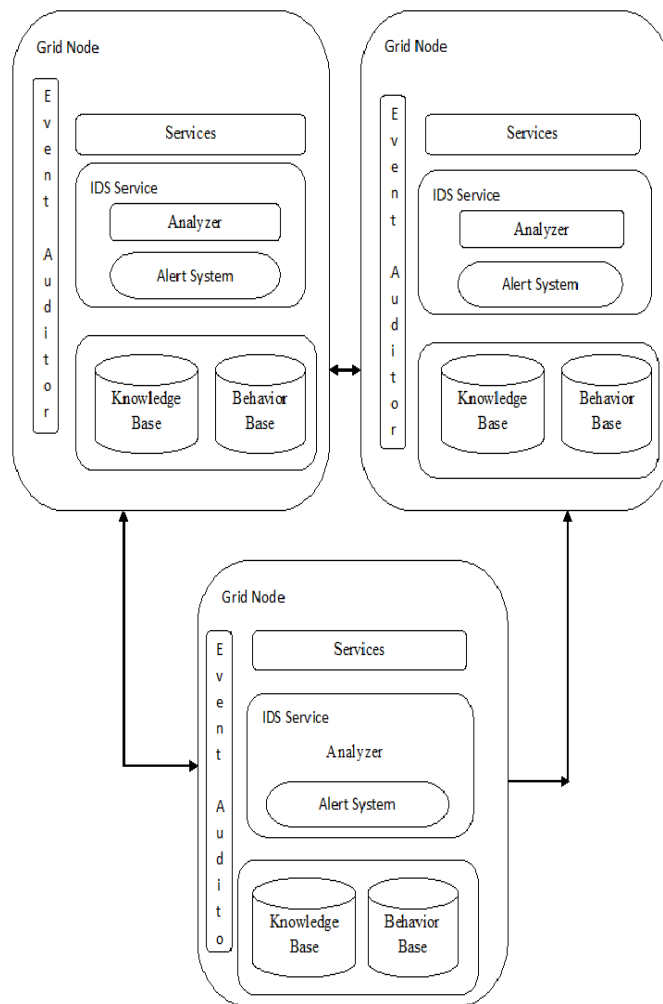

Fig1. Architecture of Intrusion Detection System.

2] IDS service: The IDS applies two methods of Intrusion Detection that are Knowledge Based and Behaviour Based. It consists of two components that are Analyser and Alert System. The rules analyser receives audit packages and determines whether a rule in the database is being broken. It returns the result to the IDS service core. With these responses, the IDS calculate the probability that the action represents an attack and alerts the other nodes if the probability is sufficiently high.

3] Behaviour Analysis: This method dictates how to compare recent user actions to the usual behaviour. The audited data is sent to the IDS service core, which analyses the behaviour using artificial intelligence to detect deviations. The analyser uses a profile history database to determine the distance

Proceedings of Second Shri Chhatrapati Shivaji Maharaj QIP Conference on Engineering Innovations
Organized by Shri Chhatrapati Shivaji Maharaj College of Engineering, Ahmednagar
In Association with Novateur Publications JournalNX-ISSN No: 2581-4230
February, 22nd and 23rd, 2019

between typical user behaviour and the suspect behaviour and communicates this to the IDS service.

4] Knowledge Analysis: The knowledge-based method detects known trails left by attacks or certain sequences of actions from a user who might represent an attacker.

V. PROPOSED ARCHITECTURE

A. A. Design

This system is based on extraction of honeypot system to find suspicious flows.
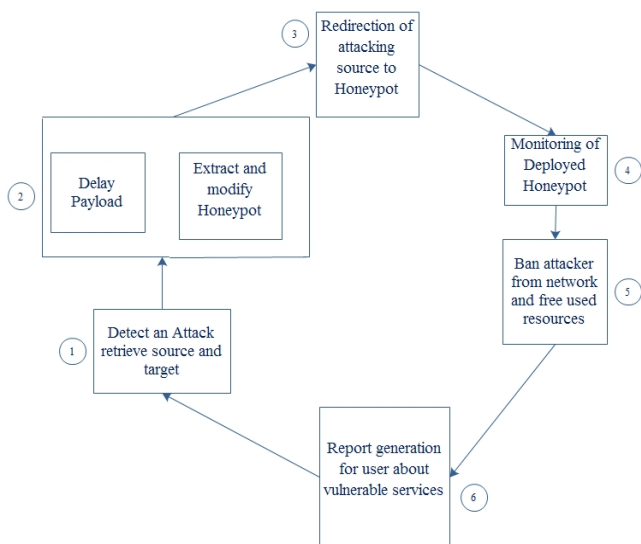

Fig. 2. Design Diagram

Fig. 2. shows the steps for extraction procedure of honeypots [11]:

1. Detect an attack retrieve source and target.
2. Delay payload, extract, and modify Honeypot.
3. Redirection of attacking source to Honeypot.
4. Monitoring of deployed Honeypot.
5. Ban attacker from network and free used resources.
6. Report generation for users about vulnerable services.

B. Algorithm

The main scope of the Honeypot system is to find out whether the incoming traffic flow is attack or normal flow. Based on the alarms generated by used Intrusion Detection System, a Tag is attached to the flow. If the Tag is found to be attack then flow is redirected towards the honeypot server else the flow is forwarded to normal destination server.

The redirection algorithm performs the per-flow treatment of each flow in the Flow List [1].

Let FL be the flow list of packets.
FDA: Destination Address of packet
FSA: Source Address of packet
NDA: Destination Address of Homogeneous Server
PDA: Destination Address of Active Server

The Redirection Algorithm

For a flow in FL
If (Tag = attack)
Parse the primary packet and search source and destination address (FDA and FSA)
    PDA = FDA
    NDA = PDA

A:

If (NDA = Destination address of honeypot)
Forward the packet to NDA
Else
    Replace NDA by destination address of honeypot and forward the packet to NDA
If (More Fragment = 0)
    Goto S
Else
    Parse next header of the flow for PDA
NDA = PDA
If (Tag = attack)
    Goto A
Else
    Goto B
Else
    Parse the primary packet and search source and destination address (FDA and FSA)
PDA = FDA
NDA = PDA

B:

If (NDA = Destination address of active FTP server)
    Forward the packet to NDA
Else
    Replace NDA by destination address of server
    Forward the packet to NDA
If (More Fragment = 0)
    Goto S
Else
    parse next header of the flow for PDA
    NDA = PDA
If (Tag = attack)
    Goto A
Else
    Goto B
S:
Stop

VI. CONCLUSION AND FUTURE WORK

Implementation of this system gave me an opportunity to study Honeypot and IDS system in detail. It is important for organizations to secure their digital assets by detecting and preventing vulnerabilities before they are exploited. Honeypots provide a valuable tool to collect information about the behaviours of attackers in order to design and implement better defences. The design of the architecture and details about the implementation of Honeypot System are presented.

Honeypot system implements Load Balancer. Results show that Load Balancer processes multiple requests in less time which increases speed of the system. If any server in the system fails, the performance will not be degraded as the requests will be redirected towards other servers. This increases the scalability of the system.

Honeypot system generates less number of alarms than IDS. Hence it can be concluded that combination of Honeypot and IDS system can be suitably used as most efficient system to provide security for servers.

ACKNOWLEDGMENT

REFERENCES

[1] Anjali Sardana and R. C. Joshi "An Integrated Honeypot Framework for Proactive Detection, Characterization and Redirection of DDoS Attacks at ISP level". Journal of Information Assurance and Security 1 (2008) 1-15.

[2] Babak Khosravifar, Jamal Bentahar, "An Experience Improving Intrusion Detection System False Alarm Ratio By Using Honeypot", Proc.IEEE of the 22nd International Conferrence On Aavanced Information Networking and Appliication, 2008.

[3] C. Hecker, K. L. Nance, and B. Hay. "Dynamic honeypot construction", In Proceedings of the 10th Colloquium for Information Systems Security Education, pages 95-102, June 2006.

[4] Hamid Mohammadzadeh.e.n, RozaHonarbakhsh, and Omar Zakaria, "A Survey on Dynamic Honeypots", International Journal of Information and Electronics Engineering, Vol. 2, No. 2, March 2012.

[5] Kleber Vieira, AlexandreSchulter, Carlos Becker Westphall, and Carla MerkleWestphall, "Intrusion Dtection for Grid and Cloud Computing".

[6] M. Balamurugan, B Sri ChitraPoornima, "Honeypot as a Service inCloud".

[7] M.Mokube and M. Adams, "Honeypots: concepts, approaches, and challenges". In Proceedings of the 45th annual southeast regional conference2007.

[8] N. Provos. "A virtual honeypot framework", In Proceedings of the 13th conference on USENIX Security Symposium - Volume 13, SSYM'04, page1, Berkeley, CA, USA, 2004. USENIX Association.

[9] Nithin Chandra S.R, Madhuri T.M, "Cloud Security using HoneypotSystems", International Journal of Scientific and Engineering Research Volume 3, Issue 3, March -2012 1 ISSN 2229-5518.

[10] Reto Baumann, Christian Plattner, "White Paper: Honeypots", February 26, 2002 International Conference on Web Services Computing (ICWSC)2011, Proceedings published by International Journal of Compute ApplicationsR(IJCA).

[11] Sebastian Biedermann, Martin Mink, Stefan Katzenbeisser. "Fast DynamicExtracted Honeypots in Cloud Computing".