Network traffic intelligence using a low interaction honeypot

IOP Conf. Series: Materials Science and Engineering **263** (2017) 042096 doi:10.1088/1757-899X/263/4/042096

Network traffic intelligence using a low interaction honeypot

Tendai Nyamugudza[1], Venkatesh Rajasekar[1], Prasad Sen[1], M Nirmala[2] and V Madhu Viswanatham[1]
[1]School of Computer Science and Engineering, VIT University, Vellore-632014, Tamil Nadu, India
[2]School of Information Technology and Engineering, VIT University, Vellore-632014, Tamil Nadu, India

E-mail: vmadhuviswanatham@vit.ac.in

Abstract. Advancements in networking technology have seen more and more devices becoming connected day by day. This has given organizations capacity to extend their networks beyond their boundaries to remote offices and remote employees. However as the network grows security becomes a major challenge since the attack surface also increases. There is need to guard the network against different types of attacks like intrusion and malware through using different tools at different networking levels. This paper describes how network intelligence can be acquired through implementing a low-interaction honeypot which detects and track network intrusion. Honeypot allows an organization to interact and gather information about an attack earlier before it compromises the network. This process is important because it allows the organization to learn about future attacks of the same nature and allows them to develop counter measures. The paper further shows how honeypot- honey net based model for interruption detection system (IDS) can be used to get the best valuable information about the attacker and prevent unexpected harm to the network.

1. Introduction

An intrusion detection system (IDS) is a basic part in a network security systems condition. It empowers recognition of suspicious packets and attacks. With the assistance of IDs, all network traffic can be monitored. Now and again the IDS may likewise react to abnormal or malicious activity by making a move, for example, obstructing the client or source IP address from getting to the system.

Because of the way that the Internet and nearby systems have turned out to be ubiquitous, the quantity of interruption occasions has developed. A security strategy around these frameworks is fundamental. Its goal is to diminish the dangers identifying with: privacy, respectability, accessibility, and non-denial. Associations are steadily actualizing frameworks that monitor IT security. Since a few years, organizations have set up a few instruments set up several mechanism interruptions like; firewalls are filter inbound network system traffics, Antivirus used to stop proliferation of worm, validation with a specific end goal to control get to information and VPN innovation, even some encrypt the dataflow between the head office and other branches over the public network like internet.

Sometimes these mechanisms have set of rules that information system have configuration breaches that allow the attackers to bypass the security mechanism.

A firewall implements which traffic is permitted in and out a network system, is based on rules and characterized has been pre-assigned. The firewall assesses the headers but not the substance of information in data packets. Some exploits try to take advantages of the security weakness protocol that are accessed through the firewalls. Programmers will utilize your web server which has been compromised as a vulnerable to attacks on other inside servers.

That is the reason a secondary mechanisms line of barrier is essential, the interruption discovery framework (IDS). IDS have since a couple of years picked up a lot of interest, and they are an essential part of guarded measures shielding PC frameworks and systems and network from Abuse. In any case, that does not excluded organization to have an all-around characterized and connected security rules and policies, before executing IDS.

1.1 Classification of intrusion detection systems:

Interruption Detection is the specialty of recognizing unseemly or suspicious traffic against PC or systems network system. Today, it is hard to keep up PC network or systems gadgets up and coming, various ruptures are distributed every day. IDS monitor the consumption of such systems and detect the phantasm of insecure states. This uncertain state can be either an endeavor from inner clients to mishandle their benefits or outside clients (assailants) to adventure security vulnerabilities.

The detector takes out all superfluous data, figures out whether this activity can be considered as a side effect of an interruption, and makes a move (send cautions for instance).

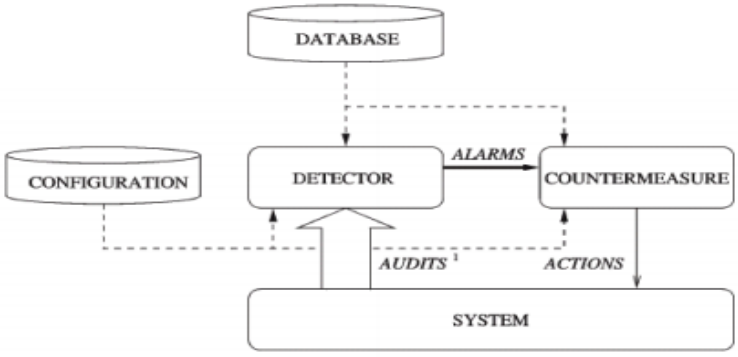

Figure 1. Simple intrusion detection system

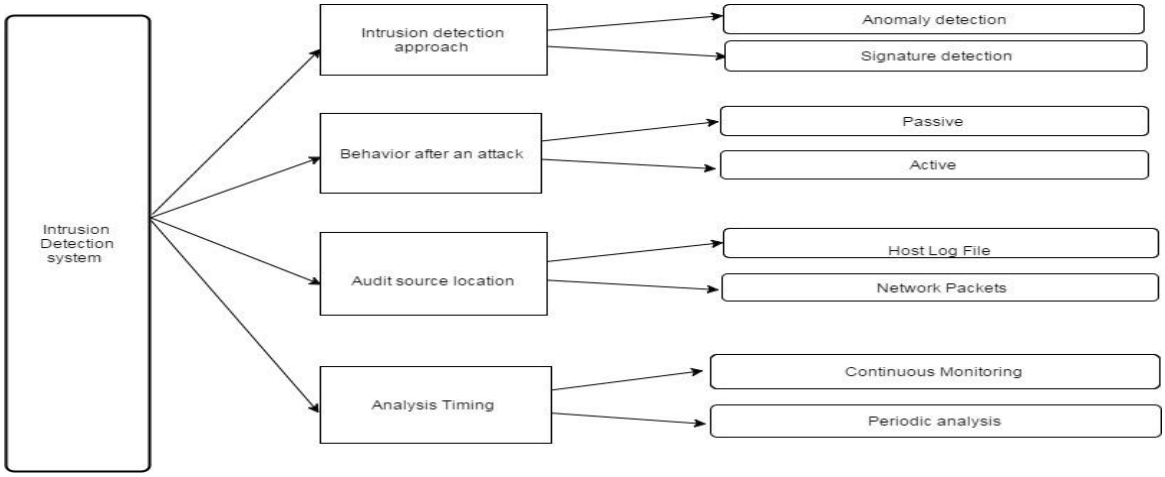

Figure 2. Characteristics of Intrusion Detection System

1.2 Honeypot

As described in [5] a honeypot copies the behaviour of honey ant in which the food repository is stored in them and often raiders attack them to loot their food. This idea has been translated to modern computer networks where intruders often invade networks to perform malicious activities. A honeypot is dummy machine in a network with the purpose of distracting an attacker from the organization's valuable machines. They can be configured to provide information about an attack earlier before they exploit the network at the same also allowing network administrators to gain knowledge about attack trends.

There are two types of honeypot low interaction and high interaction:
High interaction honey -These are complex system which have capacity to capture large amount of information and allows attackers to interact with real systems, thereby allowing organizations to gather more information [9].

Low interaction honeypot -A low interaction honeypot if limited in the way which it interacts with the attacker. It provides emulation of services in which the attacker is limited by the way the honeypot interacts with him [9].

This paper describes the components, setup, operation and implementation of low interaction honeypot which captures network malicious traffic in a network system. The paper is organized as follows section III describes existing works related to honeypot, section III describes the design and implementation and Section discusses the implementation results

2. Related work

In [1], they used hop-by-hop trace back mechanism in honeypot. In this technique they got accurate signature of the attack. It's called honeypot back-propagation. The roaming honeypots will receive packet and it'll initiate a trace back process by altering autonomous systems (ASs) and alert triggers an AS-level input-debugging-like process. The main use of this honeypot back-propagation is to distinguish attack packets from legitimate packets and enables aggressive action.

In [2], they analyzed the data of two high-interaction honeynets which are deployed in a corporate and an SME (Small to Medium Enterprise) environment and a distributed honeypots deployment. For the network and forensic procedures they have implemented a systematic approach. In this paper they analyzed the times between attacks on different hosts, operating systems, networks and geographical locations.

In [3], they used honeypots to give protection ICS (Industrial Control System) because the ICS systems with network communication remotely. They create leeway to prevent maximum and effective concentrative cyber-attacks. They distinguished the attacks in the three different stages like "Information Gathering Time", "Free Attacking Time" and "Cover Up Time". So the authors proposed a Camouflage Net that changes its own configuration when it detects a signal of certain illegal activity.

In [4], the authors did a study on honey net which in deployed in private organizational network. They said that these firewalls, IDS and IPS are very much effective in known attacks but it'll fail to detect the unknown attacks. The low-interaction honeypot which is deployed to capture the data and characterized into three major classes 1) legitimate traffic, 2) traffic due to system misconfiguration and 3) traffic due to worm propagation or misconfiguration.

2.1 Existing system honeypot products

This segment reviews some of the existing honeypot like Honeyd, HoneyBot and Specter.

2.1.1 Honeyd

Honeyd is a virtual honeypot which works in linux/unix based operating systems. Honeyd can construct multiple virtual hosts on the network using a single physical host. Since it's a virtual honeypot it is capable of monitoring multiple operating systems and give specific messages depending with operating system[8].

2.1.2 HoneyBoT

HoneyBoT is a windows based honeypot, it is a medium-interaction honeypot implemented by atomic software solutions. It initially started as an endeavor to identify by the code Red. HoneyBoT grand the access to attackers to upload malicious file to a quarantined area in order detect rootkits and trojans [7].

2.1.3 Specter

SPECTER is another intelligent honeypot based IDS. Specter inspects all common internet services like (SMTP, FIP, POP3, HTTP and TELNET) this are the areas were the attacker can easy trap the data. Furthermore specter will automatically take decision when the attacker tries to break into some network and it collects evidence of the attack. Specter makes decoy data which can be downloaded by attackers. These data files leave identification marks on the attacker's computer as proof. Specter will immediately attempt to collect data about each attack. [6]

2.1.4 Analysis of the existing system

The above systems are high interaction and medium interaction honeypots. The inherent problem with these is that they have higher maintenance. They require network administrators to constantly monitor the honey pot and closely see what's happening. In addition to that analyzing an incident make may take time since they provide a fully featured emulation of the real system. Another problem which emanates from the fact that they provide a fully featured system is that an attacker may be able to gain access to other non-honeypot system through the honeypot system. Scaling the system may also present a lot cost to the organization.

3. Proposed work

This paper describes implementation of a low interaction honeypot which mainly focuses on detecting network intrusions. Since it's a low interaction honey pot it has limited functions and the general aim of the implementation is to capture source of traffic coming to the honeypot. The implementation should be deployed on a machine that is solely used for the purpose of capturing and identifying malicious traffic.

3.1 Analysis of the proposed system

Multiple protocols: The Honeypot can be used to interact with traffic from multiple protocols. It is designed in such a way that protocols can easily be added and removed.

Multiple client per connection: the Honeypot can listen to any number of clients. It has a multithreaded design so can interact with any number of clients at the same time.

Logging: The application maintains a log file of all packets send and receives for each connection.

Graphical Interface: The application provides a simple GUI to allow the user to control the application.

Resistant to DOS: The application is designed with two key features which makes it resistant against denial of service attacks

Automatic connection time out: This feature guards against a DOS attack where an attacker tries exhaust operating system network resources by opening multiple ports and leaves them idle. This attack can prevent other users like admin or attackers from connecting to the honeypot. The application automatically times out each connection there by effectively closing the connection making sure no connection is left idle.

Waiting time for opening multiple connections: This guards against an attack in which an attacker rapidly open connections there by consuming more operating resources. This attack can disturb the capturing capability or starting new connections for other attackers. The application is designed in such a way that there is waiting time between simultaneous connections the same protocol.

3.2 Programming language

Java was used as the programming language for the development of the application. The language was chosen because the developers have experience in java, the language provides an easy to use high level socket programming and also it has an exceptional thread library implementation. These factors allowed the developers to concentrate on developing the application rather than learning the language and the multithreading libraries made implementation threading in the application easy.

3.3 Multithreading

This was done so as to allow the application to interact with multiple protocols and can have multiple clients on each protocol simultaneous. Multithreading gives the application capability to have multiple hackers at the same time, without multithreading the application capability as a honeypot would be compromised thus the application would only log single hacker at a time. The multithreading is based on the concept of supporting multiple clients in java socket programming. Each protocol running has its own thread listening for connections. The diagram below shows how multithreading is implemented in the application. Once a thread is connected, a worker thread that interacts with the connected client is started at the same time the main thread continues to listen for connections there by allowing the application to listen on multiple ports while interacting with multiple clients.

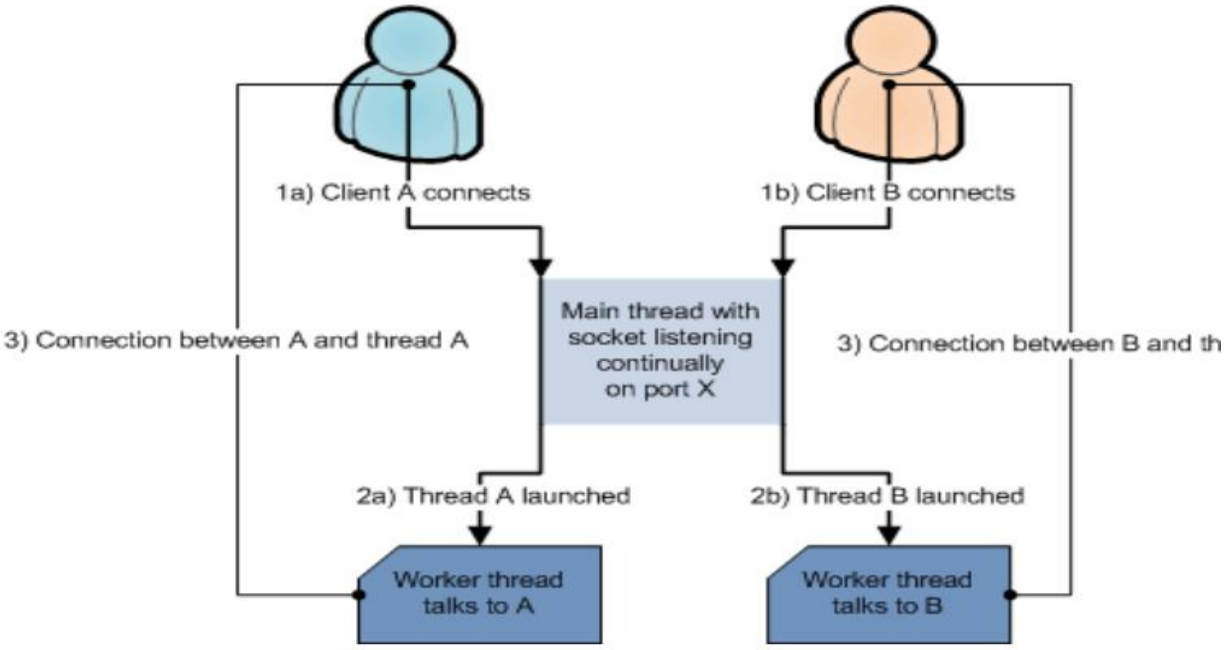

Figure 3. Multithreading architecture

3.4 Logging to plain text files

The application stores activity log files in text format in a local directory. Logs are stored as text so as to allow easy retrieval by other software that may need the files and also users can easily read them.

3.5 Support only for string based protocols

The application does not support transmission of binary data, only support string based protocols. This was done so as to limit the security risks associated with allowing users to upload binary files. With binary files an attacker may upload a file containing viruses. Since most protocols are string based the application will support most protocols a user would want to implement.

4. Design Implementation

The Honeypot comprise of two main modules, Set Honey and Honey Interaction, and two helper modules, Honey Interaction Thread and Honey Interaction Protocol. Set Honey being the main class that oversees Several Honey Interaction classes. Every Honey Interaction class implements Honey Interaction Protocol interface which provides communication logic with connected clients. Honey Interaction module starts the Honey Interaction Thread which communicates with the client upon connection.

4.1 SetHoney

The SetHoney class has modules that offers services that relate to management of other modules.
The SetHoney constructor creates a HashMap which stores the contained Interaction details and it maps a port number to one Honey Interaction module, this ensures that only a single module is added for each port. Upon HashMap initialization the log directory is also created. An instance of the HoneyInteraction is created and is passed to the SetHoney class through the register Service method and it is added to the HashMap. The registerParent() method gives it access to the logging directory. Once this process is done the application will start listening for connections. Fig 4 shows the launch and initialization.

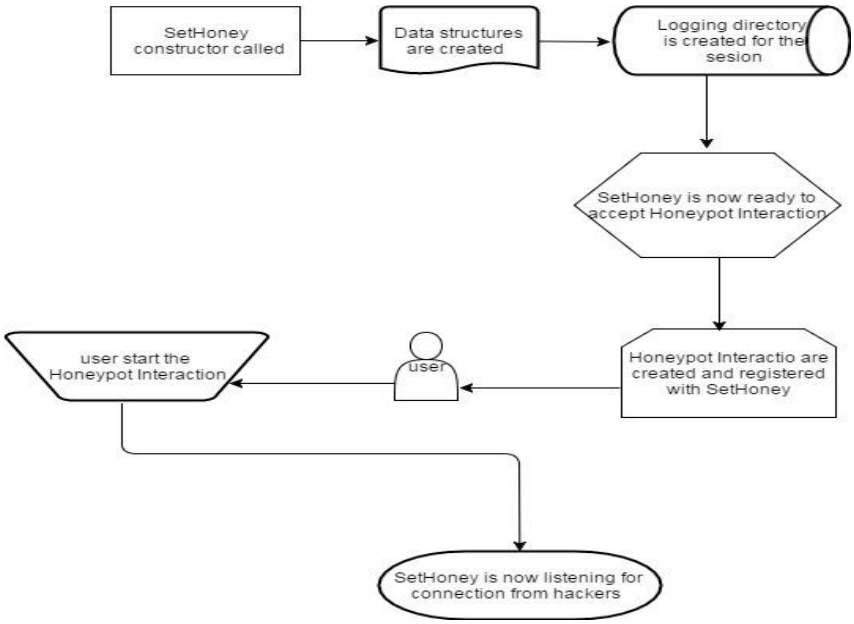

Figure 4. Application launch flow

4.2 Honey interaction

The Honey Interaction module handles the communication and logging relating to one protocol. Fig 5 shows the dataflow in the HoneyInteraction module. The Honey Interaction module is created by invoking the constructor that implements HoneyInteractionProtocol interface. The SetHoney class will then register it through the registerParent() method and the HoneyInteraction module will store the reference to the parent logging directory. Once the module is started by the user, it creates a thread and Socket that listen on the port contained in the HoneyInteractionProtocol.

If a client connects to the port HoneyInteraction starts a worker thread from HoneyInteractionThread, the thread communicates with the attacker while the HoneyInteraction continues to listen for new connections.

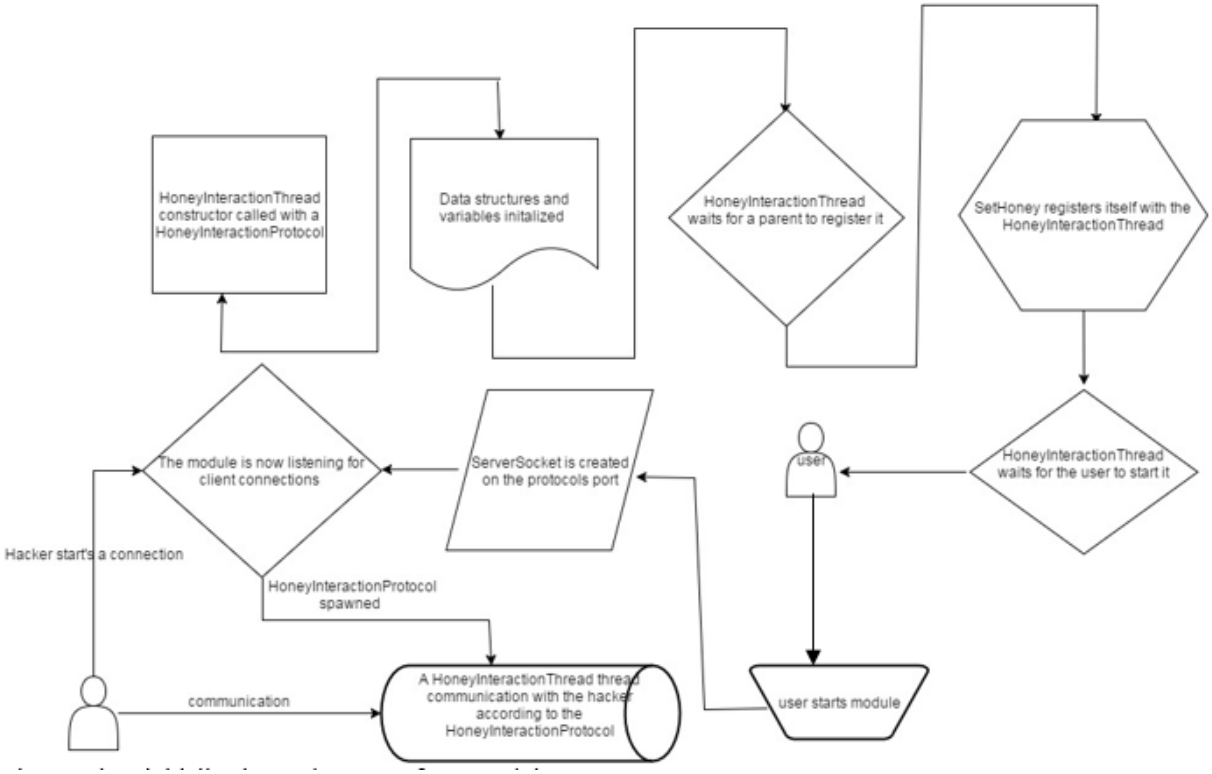

Figure 5. -The creation, initialization and startup of a HoneyInteraction

4.3 Honey interaction protocol

The HoneyInteractionProtocol has five methods which should be implemented by the protocol class:
processInput() - shown in the diagram below. It processes packet from client and returns response as string objects.

whoTalksFirst() – return a value showing who sends the first message.

getPort() - return the protocol to listen on.
toString() - returns a value used for naming log files and identify protocol in GUI.
isConnectionOver() – return the status of connection perceived by protocol.

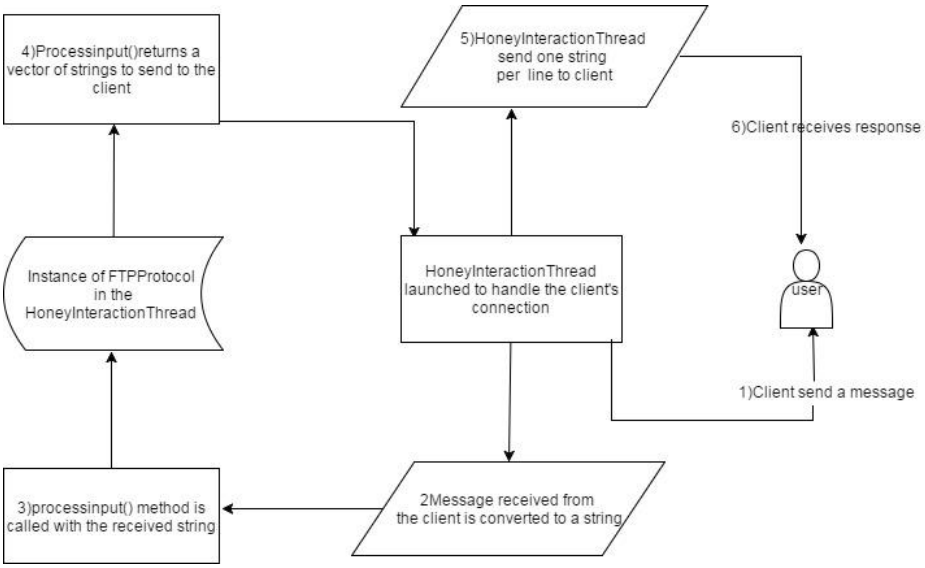

Figure 6. -The FTP protocol's communication with a client through theprocessInput() method.

5 Results and Discussion

The low interaction honeypot was tested in a local environment using localhost. Two protocols, IRC and FTP were added. The FTP protocol used the default pot 21 and IRC was configured to use pot 6669.

The FTP protocol was started and a connection was made to the protocol.Fig 7 shows one attacker connected through the FTP protocol.

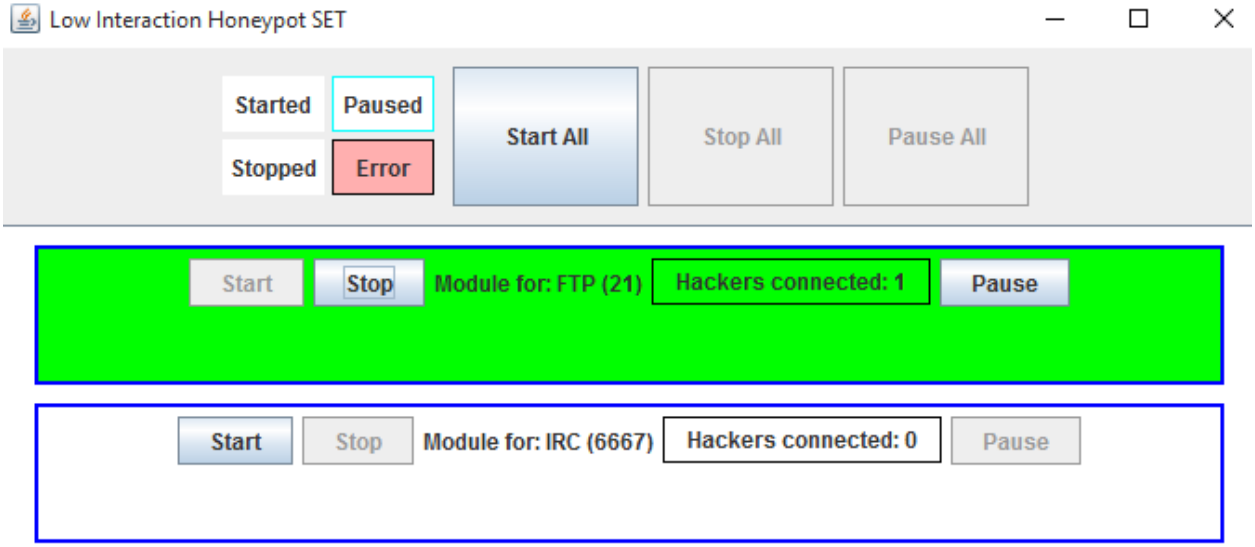

Figure 7. An attacker connected to FTP Module

Sample communication captured on the FTP protocol.

```
*********************************************
*****Started at: Tue Apr 18 10:19:59 IST 2017*******
TIMESTAMP,SRC_IP:PRT,DST_IP:PRT,PACKET
Tue Apr 18 10:19:59 IST 2017,0:0:0:0:0:0:0:1:17943,0:0:0:0:0:0:0:1:21,220 Service ready for new user.
Tue Apr 18 10:20:06 IST 2017,0:0:0:0:0:0:0:1:21,0:0:0:0:0:0:0:1:17943,user me
Tue Apr 18 10:20:06 IST 2017,0:0:0:0:0:0:0:1:17943,0:0:0:0:0:0:0:1:21,332 Need account for login.
Tue Apr 18 10:20:15 IST 2017,0:0:0:0:0:0:0:1:21,0:0:0:0:0:0:0:1:17943,USER ten
Tue Apr 18 10:20:15 IST 2017,0:0:0:0:0:0:0:1:17943,0:0:0:0:0:0:0:1:21,331 User name ok, need password.
Tue Apr 18 10:20:32 IST 2017,0:0:0:0:0:0:0:1:21,0:0:0:0:0:0:0:1:17943,PASSWORD 12345
Tue Apr 18 10:20:32 IST 2017,0:0:0:0:0:0:0:1:17943,0:0:0:0:0:0:0:1:21,501 Syntax error in parameters or arguments
*****Protocol FTP TIMED OUT talking to /0:0:0:0:0:0:0:1 using local port 21, connection closed.****
*****Stopped at: Tue Apr 18 10:22:32 IST 2017*******
*********************************************
```

Figure 8. Communication captured

The honeypot logs every connection into a separate file. The files are stored in a logging directory which can be configured in the application. It uses a timestamp for naming the files and folders. Every connection creates a new text file within the created directory. The text files are updated in real time, whenever a packet is send the log text file is updated.

5.1 Format of the log file

The log file contains a header which describes the time and date the connection was started. Afters this comes the log format comprising of timestamp, source ip, source pot, destination ip, destination pot and the packet. Every packet that was sent is logged in different line. The last section is the footer information showing how the connection was closed. This could be either timed out or finished talking.

6. Conclusion

Usually, Intrusion Detection Systems have been used in organization to dynamically screen network traffic. In today's networked environment with the advent of TSL and SSL more connections are encrypted, IDS are unable to detect encrypted connections. Honeypot are now an alternative to detect intrusions and malicious sources of traffic. Honeypot should be placed in an unused IP address so that no legitimate traffic connects to it. Honeypot allows the network administrators to understand the attackers thought. In this paper we have implemented a low interaction honeypot which captures every traffic that is send through a port being monitored. Further development can be done to the honeypot to enhance its functionality through adding modules to capture packets in binary form since the implemented version is limited to capturing text based packets. Additions can also be done to make it a high interaction honeypot which allows network administrators to get insight into an attack.

References

[1] SherifKhattab, Rami Melhema, Daniel Mosséa, TaiebZnatia 2006 Honeypot back-propagation for mitigating spoofing distributed Denial-of-Service attacks *Journal of Parallel and Distributed Computing* **66**(9) 1152-1164

[2] Robin Bloomfield, IlirGashi, AndreyPovyakalo, Vladimir Stankovic Comparison of Empirical Data from Two Honeynets and a Distributed Honeypot Network (Centre for Software Reliability, CityUniversity London, London, UK).

[3] Hidemasa Naruoka, Masafumi Matsuta, Wataru Machii, Tomomi Aoyama, Masahito Koike, Ichiro Koshijima, Yoshihiro Hashimoto 2015 ICS Honeypot System (CamouflageNet) based on

attacker's human factors *Procedia Manufacturing* **3** 1074 -1081
[4] Rupinder Kauri, Er. Sunil Nagpa J, Saurabh Chamotra 2015 Malicious Traffc Detection in a Private Organizational Network Using Honeynet System Annual IEEE India Conference (INDICON)
[5] RajaniMuraleedharan, and Lisa Ann Osadciw 2009An Intrusion Detection Framework for Sensor Networks Using Honeypot and Swarm Intelligence; IEEE
[6] Lance Spitzner: 2003; Specter: A Commercial Honeypot Solution for Windows
[7] http://www.atomicsoftwaresolutions.com;HoneyBOT is a medium interaction honeypot for windows.
[8] Niels Provos 2007www.honeyd.org Developments of the Honeyd Virtual Honeypot
[9] Christian Kreibich, Jon Crowcroft; 2004; Honeycomb – Creating Intrusion Detection Signatures Using Honeypots **34**(1)