Banking Security using Honeypot

Sandeep Chaware
D.J.Sanghvi College of Engineering, Mumbai
smchaware@gmail.com

Abstract

New threats are constantly emerging to the security of organization's information systems infrastructure. Firewall and VPN cannot prevent all intrusions and do little to prevent attacks from within the organization itself. Intrusion detection plays a vital role in ensuring the integrity of a network's security. Network intrusion detection systems (NIDS) or Intrusion Detection Systems (IDS) have long been seen as the most effective means of detecting attacks. However they do have significant weaknesses. The increasing quantity and diversity of legitimate network traffic has resulted in ever increasing hardware costs and the large number of false positive alerts generated can be too much to analyze effectively. By relying on the search for known attack signatures NIDS are unable to detect new forms of attacks and the use of encryption prevents them examining traffic altogether. An additional approach is required to tackle such problems. In this paper, we proposed a secure system for banking application using honeypots. Using this system, at least data integrity can be ensured along with monitoring the interaction to detect possible attack.

Keywords- *IDS, Honeypot, Monitor, Server, Port Scanner, Attack Finder, File Watcher.*

1. Introduction

There are number of threats that can not prevented, rather detection will be best remedies. Various systems has been implemented for detection of an attack, one most popular is Intrusion Detection System (IDS). This system will take appropriate action, once the attack has been detected. In general IDS works as follows: It identify System, log traffic between source and destination, log all traffic from that source, log content of the packet from that source and take appropriate action.

A honeypot is an IDS. It works similar to IDS. A honeypot simulates a vulnerable computer on a network. It contains no critical data or applications, but has enough data to lure an intruder. When an intruder attacks the honeypot, the activities of the intruder are recorded in the log files. Later, these log files are audited; the attack signatures are identified and stored in the database for use by IDS.

There are two types of honeypots, production honeypots, it does not prevent intrusion, and rather it strengthens the security systems of an organization. Organizations normally have a large amount of log data to be analyzed. Hence, it is difficult for IDS to detect the exact time of intrusion. Due to this chances of missing an intrusion are high, i.e. generates of false positives. Also, IDS may generate false negatives. In such cases, it is useful. In this, the organization needs to deploy one of its servers as a Honeypot. It solves the problem of false negatives. Number of activities on the honeypot are far less because normal users do not use the honeypot. Each log created at the honeypot refers to an intrusive activity. By detecting intruders, it helps to reduce the risks of security breaches.

The other type of honeypot is research honeypot. It is useful in scenarios where the intrusion has already taken place. It is easy to trap intruders because the honeypot has limited data, which can be analyzed to find out who the intruder was? How the intrusion was carried out? Even after the intrusion is detected, the intruder is allowed to continue. The attack patterns are monitored and new policies are made to protect the computers from any future attempts. However, it is not completely safe to use a honeypot for research because after the intruder has access to the honeypot, the whole network of the organization is exposed.

The main advantages of honeypots are, first, all traffic to a honeypot system is considered as an intrusion. So, better detection rates. Second, honeypots can be deployed as vulnerable systems, misleading intruders to think that they can easily break into. Third, honeypot server may raise an alarm before the intrusion actually happens. However, there are some disadvantages like honeypot setup must be regularly monitored, if an intruder gets into a honeypot, whole system is exposed. Plus, he can launch new attacks to other networks from the honeypot. If the intruder does not attack the resources on a honeypot, then it is of no use. And, if no intrusion takes place, honeypot remains idle and is a waste of money.

2. Existing Systems: A Survey

There are no such existing systems and the work on honeypots may take a little bit longer to surface but there have been some serious attempts to deploy the most from the concept and the finest example to it may be the new honey monkeys that Microsoft are coming up with. The most of the attacks by a hacker would like to attack on the database concerning the username, the password and their respective account numbers. After acquisition of the same the hackers would very conveniently trespass the security walls of authentication and authorization and thereby making the transaction official.

2.1 Pitfalls in the current Architecture

2.1.1 There is the risk of detection:

Once the true identity of a honey pot has been identified, its value is dramatically reduced. Attackers can ignore or bypass the honey pot, eliminating its capability for capturing information. Perhaps even more dangerous is the threat that once identified, an attacker can introduce false or bogus information into a honey pot, misleading the data analysis.

2.1.2 There is the risk of disabling honey pot functionality:

This could be an attack against either data control or data capture routines. Attackers may want to not only detect a honey pot's identity, but disable its data control or data capture capabilities, potentially without the honey pot administrator knowing that functionality has been disabled. For example, an attacker may gain access to a honey pot within the honey net, and then disable data capture functionality on the honey pot. The attacker could then feed the honey pot with bogus activity, making administrators think data capture is still functioning and recording activity, when its not. Having multiple layers of data control and data capture helps mitigate this risk, as there is no single point of failure.

2.1.3 Catch all of remaining risk.

3. Proposed System Architecture

Figure 1 shows the basic functionality of the honeypot. The attacker uses his own techniques to get into the system to get some important information or data. There will be two kinds of interactions. First, low interaction honeypots, in which enough interaction is provided to attackers, through which, some interested attacks may be known to the system. Second, high interaction honeypots, in which through full interaction with the attacker, detail information is known.

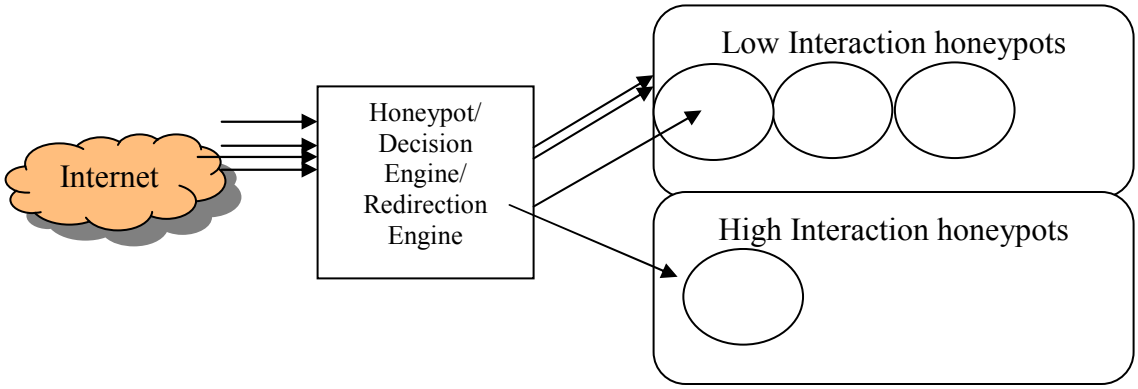

Figure 1: Honeypot System Architecture

It will basically segregate data into three layers i.e. namely user interface (open source), the dummy programs (to lure the attackers and to keep into system busy to help us to take steps) and the main part of the system. Along with the intruder detection it will also make us able to track them down as there will be an inbuilt system to maintain their data. The other users will access the user interface (UI) which will be an open source.
It will be followed by the layer of dummy programs which will be hidden from the actual UI, but the attackers can reach it easily. These programs will take a lot of time as it will create a delay and in the mean time the system will raise an alarm that will inform the system administrator that there is an intrusion going on and will give the respective authority the necessary time to take according actions. Also there will be a tracking program that will track the attackers and will help in getting hold over him. The system will keep track of the changes the attacker's make on the dummy data generated by the underlying dummy programs so as to make us aware of the manipulations the attackers were to make to the original data if it had been reachable, and also will help in taking necessary decisions as what part of the database is more crucial for it was attacked and need the almost care and protection from future threads.

The final layer that is the main database of interest will be most preciously protected can be accessed only with the password of administrator.

3.1 Proposed System Architecture

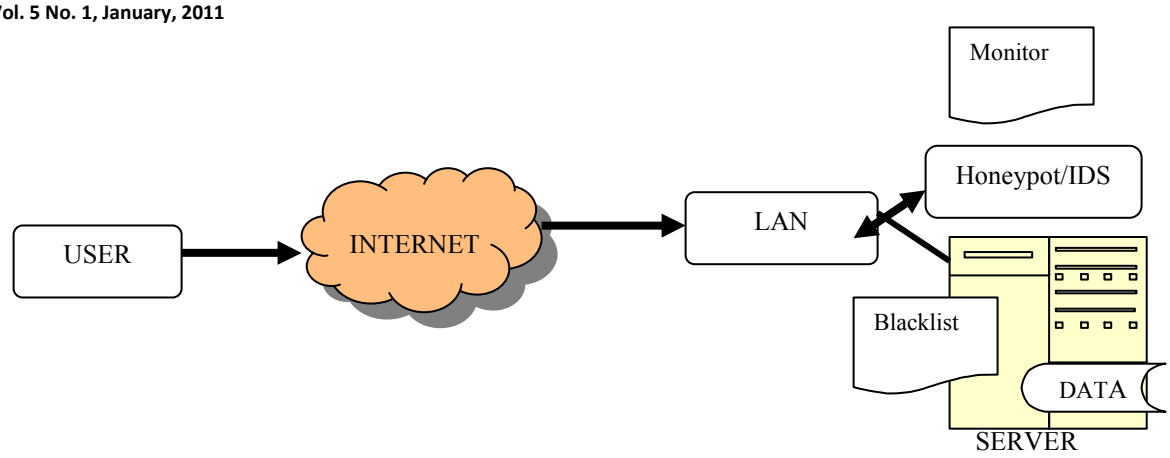

Figure 2: Proposed System Architecture using honeypot as IDS

Figure 2 shows the proposed system architecture using honeypot as IDS to protect a network. The users or attacker will access the network either Internet or direct. Within a LAN, IDS with honeypot and a centralized server with database layers as described above are being connected. Once the user will access the network, all its interactions low or high will be monitored by the IDS and make a log file for that user. IDS will decide to make a user as blacklisted or not, also server's data will be checked for integrity and identify the source of the user. Database layers also be checked for integrity by the system.

Our proposed system for banking system, which will divide internal database into three layers, first, the public database, which will have all the information for the public to view like new policies or schemes by the bank or some new promotion stunt for e.g. some added bonuses or credits for making a current or saving account with them in some specific time. The point being the data which helps bring business more than nuisance will be placed in the public database. Second layer will be the main database, which will be real data of various users for the system, like the database for the list of account holders or for that sake the policies that are on their way to the market but still not disclosed due to assent of some senior authorities or due to some current market conditions. The third layer is the dummy database, which holds the information having no relevance to the real life world. This data will be automatically generated by the system by shuffling of the existing system. This is the data that the system will offer to the attacker and it is to this that we propose to set our eyes on to monitor the attack.

With respect to above three layers of database, we have four modules in our system are described below.

3.1.1 Port Scanner: In this module, scanning of the open ports of the system is made. Open ports are the easiest and most convenient method of attacking. This module can work easily well if we were to increase the number of servers for effective handling the ever growing needs for performance and the originally assumed server will be able to check the open ports on the new server without actually deploying the entire software on the new machine.

The proposed algorithm is:

Step 1: Start with the UI for port Scanner.
Step 2: Enter IP Address of the Server.
Step 3: Enter the range of port numbers required to be scanned on the server machine.
Step 4: Click on "Start".
Step 5: If port is available for use then display the port number on the interface,
Else exit.

Figure 3: Port Scanner Algorithm

3.1.2 Attack Finder: In this module, in which actually blacklisting of the IP address of the attacker is made. The input being the server address and the port to be monitored there will be constantly monitoring the open ports and if in case there are any probability for any malicious activities and there is a constant request from the same particular IP then it will be blacklisted. We have assumed a threshold of 60 requests but it can be very easily altered according to the requirements of the hour.

The proposed Algorithm is:

Step 1: Enter IP address and port number of the client.
Step 2: Add to watcher list.
Step 3: Monitor the number of requests or packets sent.
Step 4: If number of requests less than limit specified by server then
allow packet transfer between server and client
Else
- if server displays message "Blacklist IP"??" then
  - Add IP address to Blacklisted Log
- Else goto step 3.

Step 6: Disconnect client from server.

Figure 4: Attack Finder Algorithm

3.1.3. System File Watcher: In this module, a constant watch on the files having the dummy data stored in it is made. To the outside world, the data will be seeming and correct but actually it is not. The system file watcher will continuously monitor the files and as soon as there are changes in the content of the file it will be alerting the administrator with a pop up message as to which file and when the file has been modified.

The Proposed Algorithm is:

Step 1: Specify the path of the file which needs to be watched by clicking on.
Step 2: Add the file to the watch list by clicking on "Add to Watcher".
Step 3: Click on "Start" to start monitoring any modifications being made.
Step 4: If any modification is made to the file, it is reported on the UI specifying the date and time along with the file being modified.
Step 5: Click on "Stop" to stop monitoring the file.

Figure 5: System File Watcher Algorithm

3.1.4. Attack Client: This is a testing module so as to try and make an attack on the system. The input this module will be the IP address of the server to be attacked and the port. The attacker will be giving the port to attack in a random fashion and if the port is closed or occupied there will be no effect on the system but if incidentally the port is open then the application will ask for the number of requests to be sent. At the same time, as there is our module sent and working on the open ports we will be receiving messages in the form of log records as to who is sending for how many requests and if the number exceeds and particular threshold (say 60) the module will ask to whether we wish to blacklist the IP or not.

3.2 Benefits of Proposed System:

- Since there is a complete division of the databases there is only need to keep a close watch on the data files that have dummy database stored in them.
- As the dummy layer is being attacked there is no harm on our true database. Any changes on the dummy database will not be reflected on the true database.
- As work is divided in to various modules the work becomes easy and even the maintenance is easy.
- Work can be done in any order. That is you can first open some ports and then watch them or start watching them and then open them in the same way any activity can be done in any way but it is always advisable to do the tasks in the order the is specified by the software and that being: Put all the important file on the watch, put the ports on watch and then finally, open the ports. However the order change will affect the working of the system in the least.

4. Results

Figure 6 shows the result of ports that are open on the server. User or attacker will give random port range to test. The port scanner module will take the IP address of the server and range of the ports as input. This displays the ports that are open. The result attacker finder module is as shown in figure 7. The result shows for the open ports, the number of request messages from the attacker will log into the log file. If it crosses the threshold value for number of request from IP address, the IDS will decide to blacklist the IP address. Figure 8 shows the result of file attack watch window. This shows the modified file by an attacker with modified date and time. By looking into the log of files, the system administrator will come to know the attack on the file system.

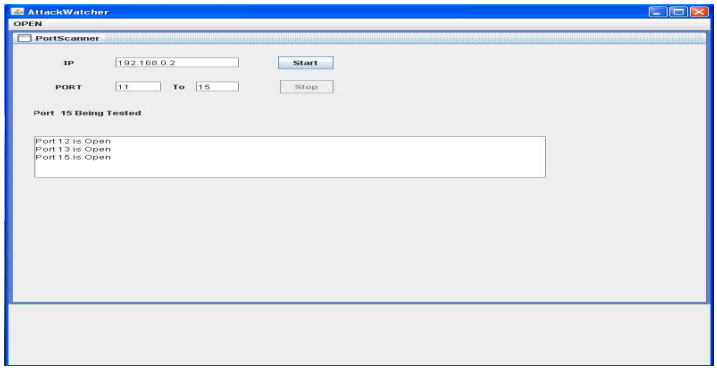

Figure 6: Port Scanner Module

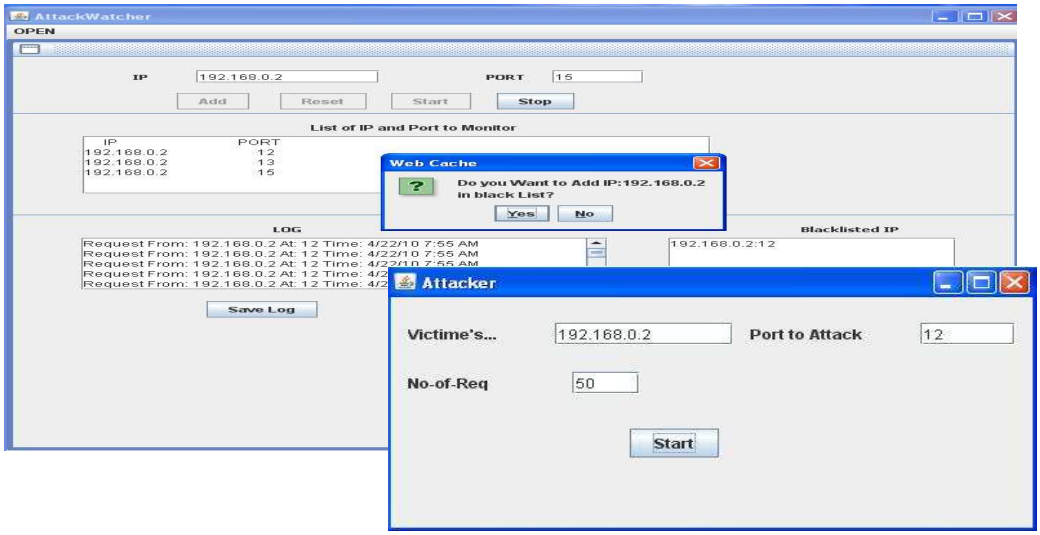

Figure 7: Attack Finder Module

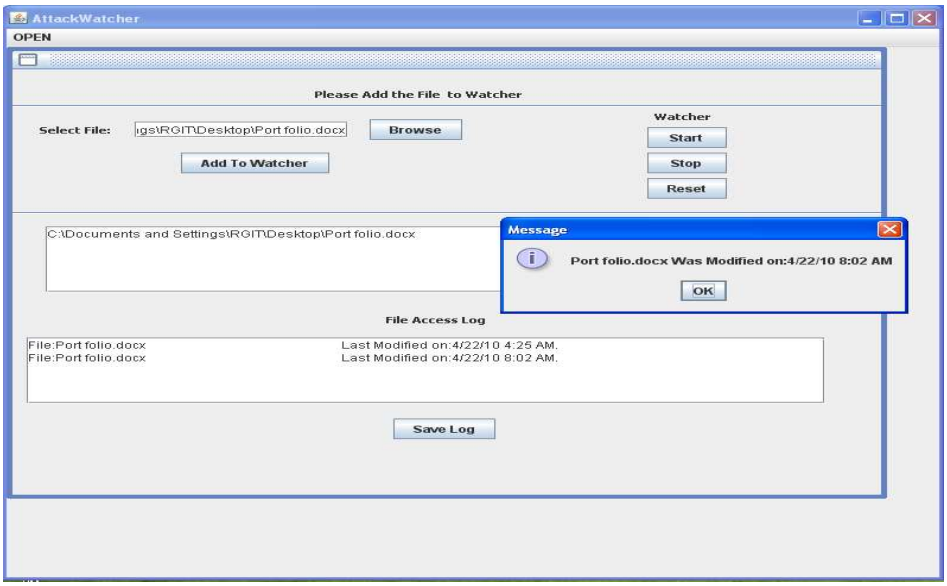

Figure 8: File Watcher Module

5. Conclusion and Future Scope

In the future, honeypots are going to become a critical weapon in a security professional's arsenal. Honeypots have the ability to catch new hacker toolkits and scripts, and are able to reduce the effectiveness of these tools in the wild by allowing security practitioners the capability to analyze these new tools. As we know in the IT security field we have a viable arms race between the discovery and exploitation of vulnerabilities and patching those discovered vulnerabilities (in addition to developing strategic security

mechanisms). Security professionals can use honeypots to delay the time between when vulnerability is found (or a new tool is created) and when that vulnerability is exploited by a malicious intruder. Also, security professionals can use honeypots to develop better methods and skills by gaining invaluable knowledge from watching an actual attack in progress.

There have been a lot of intrusion going on and the only way out of this or atleast to minimize it would be to counter the attack. Like a lot of solution's that have been ,success initially and failed the text of time it wont be any different for the honey pot solution as well as an alternative path will be found out, till that time it can be deployed. Utilizing the best of the year and the time allocated to us we came to as good solution for the problem as much we could have. a lot of research have been done in the field but no banking solution have been implemented with the honey pot solutions. So the project can be further developed and raise the bars for banking security as there is no enough money that any other field can supply as banking can.

Future Scope:

- In our implementation, we have not included the latest rules for virus and worm detection. This system can be exclusively designed for worm detection, so that the prevention of worms can be made easier.
- The alerting message is sent to the server only periodically. This can be done based on the severity and the type of attack encountered and automatically.
- We have not emulated all the protocols and this can be extended to the operating system also.
- Upgradation in hardware and software will result in a faster response time to detect unauthorized access.

6. References

[1] Lanz Spitzner, Know Your Enemy: Learning with User-Mode Linux Building Virutal Honeynets using UML, http://www.honeynet.org , 20 December, 2002.
[2] Lanz Spitzner, Know Your Enemy: Honeywall CDROM Eeyore Bootable GenII Honeynet Gateway , http://www.honeynet.org , 07 May, 2004.
[3] Lanz Spitzner, Know Your Enemy: GenII Honeynets, http://www.honeynet.org, 12 May, 2005.
[4] Lanz Spitzner, Know Your Enemy: Honeywall CDROM Roo 3rd Generation Technology, http://www.honeynet.org , 17 August, 2005.
[5] Lanz Spitzner, Know Your Enemy: Honeynets, http://www.honeynet.org, 31 May,2006, 978-0-7695-3347-6/ 978-0-7695-3347-6/08, Jungsuk SONG-Kyoto University, Hiroki TAKAKURA-Kyoto University, Yasuo OKABE-Kyoto University, Cooperation of Intelligent Honeypots to Detect Unknown Malicious Codes, IEEE, 2008 .

Author

Mr. Sandeep Chaware, Asst. Prof. in Information Technology department at D.J.Sanghvi College of Engineering, Mumbai INDIA. He is pursuing PhD (Engineering) from NMIMS Deemed University, Mumbai. His areas of interest are mobile computing, information and network security, mobile agent-based technology.