Honeynet Clusters as an early Warning System for Production Networks

Sushan Sudaharan, Srikrishna Dhammalapati, Sijan Rai and Duminda Wijesekera

Information and Software Engineering, George Maoson University, Fairfax, VA 22030-4444

Abstract. Due to the prevalence of distributed and coordinated Internet attacks, many researchers and network administrators study the nature and strategies of attackers. To analyze event logs, using intrusion detection systems and active network monitoring, Honeynets are being deployed to attract potential attackers in order to investigate their modus operandi. Our goal is to use Honeynet clusters as real-time warning systems in production networks. Towards satisfying this objective, we have built a Honeynet cluster and have run experiments to determine its effectiveness. Majority of the Honeynets function in isolation, not sharing information in real time. In order to rectify this deficiency, we built a federation of cooperating Honeynets (referred to as a Honeynet cluster) that shares knowledge of malicious traffic. This paper describes the methods in building a hardware assisted Honeynet cluster and testing its effectiveness.

1 Introduction

Currently, numerous intrusion detection and prevention mechanisms exist to dissuade and monitor malicious traffic. One mechanism to study intruder actions monitors network probes and attacks using a single system, commonly referred to as Honeypots [8]. Honeypots are stand-alone software/hardware tools that share knowledge off-line; the information obtained from such a collection of Honeypots must be correlated manually. Currently, the Honeynets do not retaliate in real time and share each others attack information.

In order to use Honeynet outputs for real-time counter actions (defensive or offensive), we built a hardware-assisted Honeynet cluster using a collection of routers, firewalls, and servers, running various operating systems. Honeynet clusters are a collection of distributed and cooperating Honeynets that function autonomously. An intelligence-gathering module that will issue online alerts that can be fed to security appliances of production networks to mitigate operational risks is also deployed. Risk mitigation will be dependent upon the certainty and the severity of alerts. It ranges from defensive actions, such as limiting access by dynamically switching to more restrictive filtering policies at the border gateways, to offensive actions as attacker tracing and/or counterattacking on identified targets. In order to measure the effectiveness and turnaround time of our countermeasures, we calculate the time it takes from attack detection, to notification to the subscribed Honeynets to

Sudaharan S., Dhammalapati S., Rai S. and Wijesekera D. (2005).
Honeynet Clusters as an early Warning System for Production Networks.
In *Proceedings of the 3rd International Workshop on Security in Information Systems*, pages 77-83
DOI: 10.5220/0002569700770083
Copyright © SciTePress

the retaliation, i.e. the system logs the total time from attack detection to notification to retaliation by the Honeynet.

The advantage of this approach is its ability to share knowledge about ongoing attacks as an early warning system for multiple production networks. Our research enhances existing work on distributed IDS systems, active networks, and sharing knowledge (obtained from streaming data and defensive/offensive measures against network attacks. Our goal is to study the degree of coordination necessary to identify and respond to undesirable behaviours in networks, while simultaneously altering/warning the production network about this behaviour.

The paper is organized as follows: Section 2 describes the design and functionality of our Honeynet; Section 3 describes the software modules developed for our Honeynet and preliminary experiments; and Section 4 summarizes the paper.

2 The honeynet

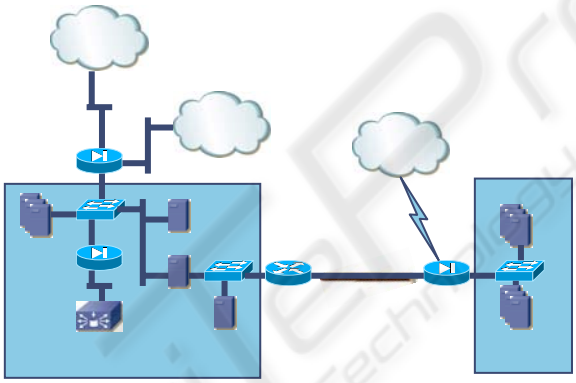

Fig.1. Honeynet

The physical design of our Honeynet is shown in Figure 1. The following are the basic hardware components:

1. Cisco products (one each): 7204 VXR router, 2950 Switch, PIX 515E – Firewall & VPN and PIX 501 – Firewall
2. Ten Gateway 935 series servers, four 1U Penguin Computing servers and 2 Sun ultra spark servers
3. One Arbornet network traffic generator and one terra byte storage server.

As shown in Figure 1, the Internet is directly connected to the Cisco PIX 515E. The DMZ (DMZ 1) on the Pix is connected to a Cisco 2950 switch. DMZ 1 hosts all applicable servers. Two ports on the Cisco 2950 are configured as a Span port. The server hosting Snort is connected to the Span port. This port is also shared by the Dell Terra byte storage server. The Arbornet traffic generator is located behind a second firewall (Cisco Pix 501) and will be generating simulated traffic on the DMZ. Services and transactions are all simulated, and multiple web servers will generate a high volume of transactions in order to make tempt an intruder even more. In

addition, several email servers with IMAP and other mail protocols are as a number of attacks are carried out through email and related services.

The Cisco PIX 501 Firewall, shown in Figure 1, is configured to send traffic only outside the system, masking the traffic generator behind the firewall. The PIX 515 Firewall uses 3 interfaces as described below:

1. The first interface used is for DMZ 1. Logging and monitoring is performed through the span port connected to the Cisco 2950 switch. The information from this port is parsed and passed to our monitoring system. SNORT [1] Tcpdump [2], and various other tools are then used to analyze the network traffic. Analyzing this traffic, and the actions taken in response to it, is described in Section 3.
2. The second interface is connected to the existing lab that consists of two functional areas. The first functional area consists of personal computers connected to the internet. The second functional area is relegated to other experiments and is separated by a firewall which isolates it from the rest of the network.
3. The outside interface is connected to the internet.
4. Traffic flow policies are implemented using different filtering rules on the firewalls. For example, in the current design, the policy we implement is as follows:
   - Allow HTTP, SMTP, ICMP etc. to enter into DMZ 1 on the Pix 515E
   - Only allow established traffic into the Inside interface of the Pix 515E
   - Do not allow any traffic into the Pix 501 from outside

The experiment will measure the effects of multiple policies on different hardware devices and how they dynamically switch according to specific security requirements. In this manner, the Honeynet can be customized dynamically. Dynamic changes in policies can be used in response to the traffic that is already in the Honeynet or contained in data received from external resources.

A Linux server connects the Internet to the Honeynet. The decision maker box detects the attack and sends notification to remote system. A smart algorithm integrated into the decision maker box accomplishes the notification task. A communication system required to collect data has also been developed to communicate between distributed entities (either between Honeynets or from a Honeynet to a production network) using Java.

The test first calculates the timing delays in the data sharing mechanism that alerts a client system, instating a new policy to safeguard itself. This is accomplished by sending a flag through a VPN connection. PIX 515E firewalls can sustain traffic associated with a small office environment. A flushing mechanism implemented at the firewall base, addresses the undetected flooding attack causing the Denial of Service (DOS or clog) on the system. In order to solve the DoS problem, as suggested by the test and review material from Cisco and other router vendors, we use the clear arp command to flush the ARP cache [3] in the PIX 515 firewall. Our initial tests with one DoS Attack tool (using our Arbornet Traffic generator) support this proposal. We have tested this architecture and quantify its ability to withstand DoS attacks by devising mechanisms that flush them during high traffic periods.

3 Modules

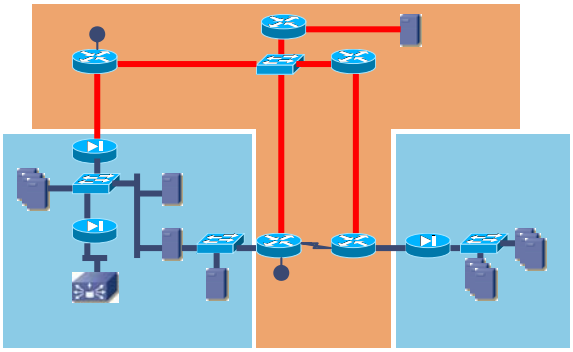

Fig. 2. Honeynet cluster with notification facilities

Since the primary objective is to use the data, obtained from the Honeynet, to secure production networks, we developed a number of modules to support the Honeynet infrastructure to collect, understand and react to ongoing traffic in the Honeynet. Figure 2 shows how they are connected to the Honeynet through the span port on the Cisco 2950 switch. This allowed us to capture all traffic on the Honeynet segment, with two data collection points.

3.1 Data Collection and Analysis Module

Independent of the physical technique and the physical location, network traffic comes in the Pcap [4] format. The libpcap [5] library is able to read data in this format. TCPDUMP [2] is popular software used to read Pcap data streams; TCPDUMP can be redirected to another application or stored for forensic analysis. Alternatively, many analyzers have their own libpcap-based packet capture capability, for real-time analysis. We used TCPDUMP data for flow-based analysis and real-time packet capture using the Snort [1] intrusion detection engine for signature and anomaly detection.

Signature Based Analysis: Snort [1] is a popular open-source, easily extendable network traffic analysis engine. The distribution includes a fairly broad set of rules (i.e. signatures), and provides flexible language for custom rule generation. Snort includes its own packet capture interface that can take the Ethernet feed off of the switch's span port, or can be configured to read a TCPDUMP data file. It also suits the physical architecture. The rule set and configuration of the Snort engine is managed from a remote console, and alert data is used in our reactionary module

3.2 Reaction Module

The purpose of this reaction module is to react to signals received from the Honeynet. The setup is described in steps below:

1. A remote network was setup with a Cisco PIX 515E as the front-end firewall.

2. VPN sessions were established from our Honeynet to the remote site.
3. Attacks and probes were initiated against the Honeynet and different response times were monitored. Once the attack was in session, it was monitored through the span port in the Cisco 2950 switch.
4. The controlling software runs in the decision maker box sending out the signal through the VPN tunnel to the remote listening agents.
5. The decision maker box located on another production network, analyzed the code, made a decision, and initiated appropriate actions.
6. This experiment was repeated numerous times with multiple network parameters, and host based vulnerabilities.
7. The latency of the whole transaction was measured under different conditions, and was optimized.

3.3 Even Handling Module

In order to manage the online alerts and reaction modules, an agent system using Java is implemented. This system uses Java Message Server (JMS) to send messages between systems. All the Detecting Agents send notifications to the Java Decision Maker (JDM). This JDM is highly configurable so that it is possible to setup the JDM to respond to various alerts differently. The primary function of the JDM is to send JMS Messages to the JMS Server. Currently we are using OpenJMS, which is an open source implementation of JMS specification. It will be easily replaceable with other JMS implementations in the future.

Finally, Java Listening Agents (JLA) completes the response process. The JLAlistens on the JMS server for anomaly events. These events are classified based on different queues and topics to which they are sent by different JDMs. The JLAs communicate with the JMS through the VPN connection and are guaranteed by the JMS to receive the messages they are interested in. The JLAs process these messages differently depending on their setup parameters. For example, a JLA that is intended to change firewall settings in response to a particular alert will change the configuration on the system it is running on.

3.4 Java Module

Event handling module process data streams from the Honeynet, will also process data from more than one Honeynet. We developed a collection of Honeynets as a source of warning systems and data collection points to expand the capabilities of our decision-making unit to support information gathered from a cluster of Honeynets.

The following set of instructions then measures the time (represented by To….Tn) taken from detecting an attack to notifying the subscribed Honeynet group:

1. ICMP packets of varying sizes transmitted from the loopback interface (1.1.1.1) on the 2621 router in the “internet” to the “target box” (10.1.2.2). Initially, all ICMP packets delivered without any problem.
2. The “attack box” (192.168.1.252) started the DoS Attacks (targa and cyclone) against the “target box” at T0. This disrupts services to the target box, and the ICMP from 1.1.1.1 could no longer reach 10.1.2.2.

3. Once the attack begins, the "snort box" on the inside network detects attack in real-time and responds to it by informing the java decision maker at T1. The JDM looks at the attack signature and informs the clients (i.e. JLA) through the java message servers.
4. JLAs sitting at the remote location receive this message. It processes the message and takes appropriate actions at T2. In this attack case, it starts a counter attack (cyclone) on the "attack box", thus, disrupting it. This allows the "legitimate" traffic to proceed without hindrance.

For example, a time measurement was initiated at T0 and terminated at T1. Some crude recording indicate a time of 0.95sec. Another time measure was initiated at T1 to T2; this recording is about ~ 1 sec. The whole turnaround time from attack to notification to subscribed clients was 3.5sec. The sample timings were recorded in the lab while performing real time logging.

4 Conclusions

In addition to network based and host based intrusion detection, Honeynets have been used to collect information about malicious behaviour. We create a novel design of Honeynet clusters, which we demonstrate, through its robust architecture and sample experiment, as being viable as an early warning system for networks that require a high degree of assurance. Our initial results show that under low loads we can react in approximately 3.5 seconds.

To further understand the full functionality and capabilities of our unique system, we are also experimenting with other add-on features to enhance the performance of the Honeynet system. This includes modules that use the collected data to perform data mining to detect the presence of new trends, as well as an intelligent agent that allows the system to learn and be adaptive to recognize such events. One application could be for the detection of new bots that crawl throughout the internet collecting sensitive data.

An additional study may investigate the legal ramifications of un-consented monitoring of transactions and implementations of possible hack-back rules. Our cluster architecture automatically monitors traffic by parsing header information. However, the reaction modules may need to take this activity beyond the headers, perhaps involving tracing back to the offending traffic's origin or hacking back. Currently, all hacking back (or any activity against the intruder) has been performed within the confines of a closed system. As outlined in [7], crossing legal boundaries for the purposes of investigating or reacting depend upon inter-state and/or international agreements. We are querying appropriately populated databases to keep track of the legality of crossing network and political boundaries. As a secondary objective, we parameterize invasive procedures so that the algorithms that enforce them would succeed only if the calling instances result in legal combinations.

References

1. http://www.snort.org
2. http://www.tcpdump.org/
3. http://www.cisco.com/en/US/products/sw/secursw/ps2120/products_configuration_guide_chapter09186a0080089948.html#41850
4. http://www.tcpdump.org/pcap.htm
5. http://www.tcpdump.org/changes/2003-11-13.02:21:40.html
6. http://richie.idc.ul.ie/eoin/SILICON%20DEFENSE%20-%20Flash%20Worm%20Analysis.htm
7. Wingfield, T. C. The Law of Information Conflict: National Security Law in Cyberspace. Falls Church, Va. Aegis Research Corp., 2000.
8. http://honeypots.sourceforge.net/

SciTePress Science and Technology Publications