

# Architectural overview of the NIFTI Network

Advanced Network Management Lab

<http://www.anml.iu.edu/>

[ebalas@iu.edu](mailto:ebalas@iu.edu)

4/22/2002

## Summary:

The Nodal Intrusion Forensic Technology Initiative(NIFTI) is developing a research network of computers designed to collect qualitative and quantitative data about network intrusions. The goals of this initiative are: to develop a scalable infrastructure that supports network intrusion and forensic research, conduct research pertaining to network intrusion, and to improve network intrusion detection and analysis. This document provides an architectural overview of the infrastructure used in this initiative.

## Introduction:

The NIFTI is a project conducted by the Advanced Network Management Lab (ANML) which is one of the Pervasive Technology Labs at Indiana University. Our focus is on collecting quantitative incident data, as such the scalability of our honeynet is of keen interest. As a result, our architecture is similar to what is known as a GENII honeynet<sup>1</sup> with emphasis on automation.

In GEN I, or traditional, honeynets there is a 1 to 1 relationship between the honeypot and the box its running on. This leads the the following problems:

1. Reconfiguring each honeypot is time consuming because switching from 1 operating system to another requires a hardware reboot.
2. Deploying a large scale honeynet becomes expensive due to running 1 honeypot per hardware device.

Within the NIFTI network, as with GENII honeynets, we are capable of serving multiple honeypots from 1 server utilizing virtual host technology. This gives us a number of desirable features:

1. We can execute multiple honeypots on one server, and are able to fully populate a /24 network with 7 servers.
2. We can change honeypot configurations for the virtual hosts we are serving by simply copying files and issuing commands to the VMware server. Further we can automate this process entirely such that the user only needs to modify a master configuration

---

<sup>1</sup> <http://project.honeynet.org/papers/honeynet/>

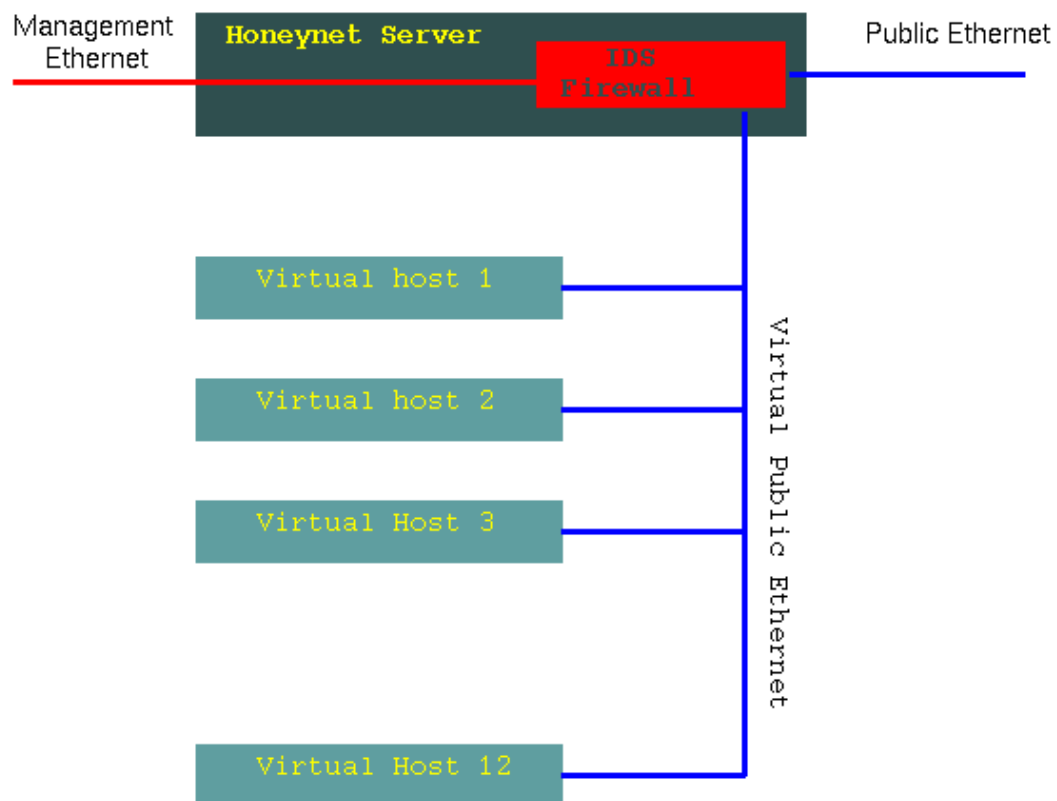
and the system takes care of the rest<sup>2</sup>

3. We are able to maintain common honeypot configurations and control their execution centrally.

### The Infrastructure:

The system has two types of physical device: Honeynet Servers, and Front End Processors. These 2 devices are not accessible from the Internet but they provide bridged ethernet access to the honeypots.

The Honeynet Server(HS) is a Linux based server running VMware GSX. The hosts it serves are honeypots in our case. Along with the virtual hosts that the server provides, the server also runs Snort<sup>3</sup>, as the IDS, to record all necessary data. The HS itself has no Internet access, but it bridges the public ethernet to the virtual ethernet of each honeypot.



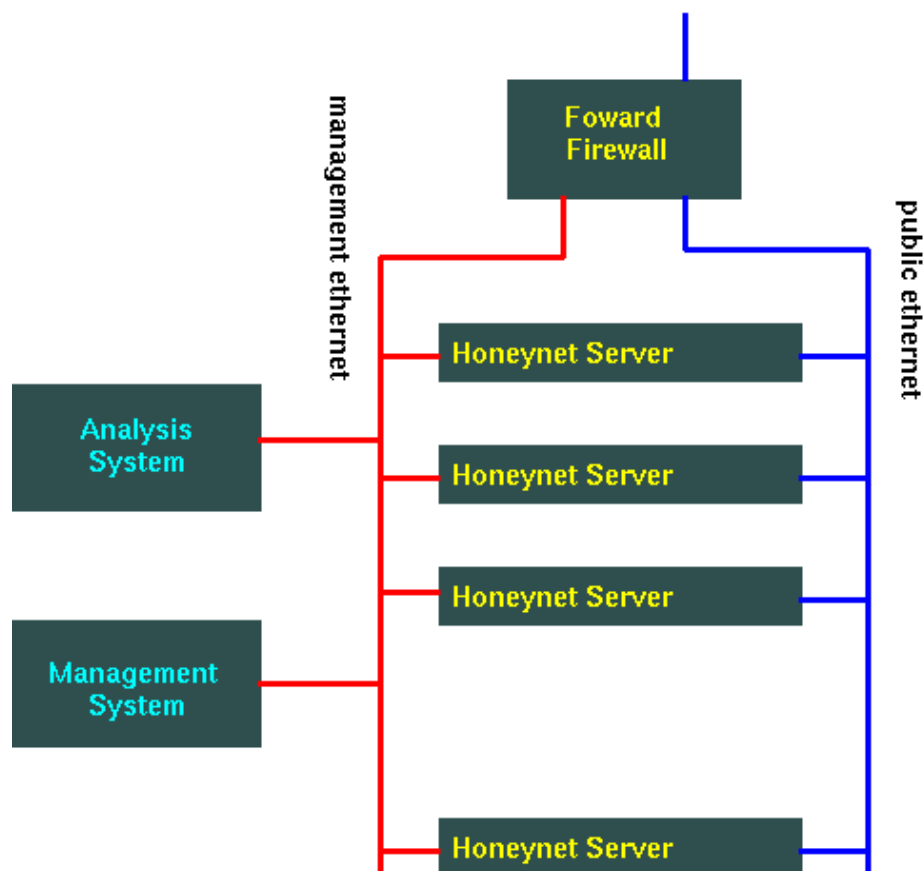
<sup>2</sup> The Distributed Virtual Host Management System: <http://www.anml.iu.edu/docs/blah>

<sup>3</sup> <http://www.snort.org>

The Front End Processor(FEP) is the brains of the system, it provides bridged ethernet access to the internet for the honeypots, which is firewalled and rate-limited. It acts as a data collector and configuration manager as well. As a data collector, it runs a relational database which the Snort instances log to and it records every packet that enters the honeynet. As the configuration manager this box maintains a master configuration that controls every aspect virtual host execution and address assignment.

### Basic Network Structure:

From the network perspective, there are two networks: the public network, and the management network. The public network is bridged through the FEP to HS where it is further bridged to the honeypots. A remote entity is unable to detect the presence of the FEP or HS in the path to the honeypot. The management network connects the private side of the FEP and HS. This is a restricted access high capacity network used for internal honeynet communications: Data Collection, honeypot execution control, etc.



**Data Control:**

Data control is provided at multiple levels. At the network level we rate limit traffic in and out of the honeynet: at the switch we limit by configuring 10M ports, at the FEP we use Dummynet to provide each honeypot with 128Kbps capacity. The FEP also acts as the Honeynet's firewall, thus we can do any type of filtering we need.

**Data Capture:**

The Data we gather is accomplished through use of libpcap based tools, mostly Snort. Currently the system records all traffic on the honeynet, all IDS signature matches, and sessions broken out by the IDS. All signature matches are recorded to a postgresql database on the FEP and made available for realtime viewing. Currently session breakout and packet capture are done on the FEP and signature matching is distributed across each HS.

**Data Collection:**

Data is collected on the FEP and the data from the HS is provided to the FEP through a Database connection across the management network. To keep all of the servers in sync, the FEP acts as an NTP server to all HS. Currently we are not in full compliance with the standard requirements as defined by "Honeynet Definitions, Requirements, and Standards" version 1.4.5<sup>4</sup> but as the network matures this will occur.

**Virtual Host Management:**

With large scale virtual honeynets it is critical to have access to mechanisms that facilitate the configuration and control of virtual honeypots across the servers (the scope of this configuration is which honeypots should run on the HS and what IPs they should have). As a result we have developed a Distributed Virtual Host Management System (VHMS) for Vmware. This system is fully described in a document titled "A Distributed Virtual Host Management System for Vmware GSX". This system provides the following features:

1. Centralized configuration management based on XML and rdist.
2. Centralized execution control and monitoring based on the GSX management API
3. High level of automation to facilitate the reconfiguration of the honeynet's distribution of honeypot types and the IP address each instance uses.
4. Ability to automatically suspend all honeypots on a HS at shutdown and not need to reboot them when HS comes back on line.
5. Ability to shut all honeypots down with one command.

**Future Developments:**

---

<sup>4</sup> <http://project.honeynet.org/alliance/requirements.html>

This design as it stands will allow us to do a great amount of research and development into the use of honeynets to collect quantitative data. However this current design is predicated on having the FEP and HS co-located. In the next generation of this network we are going to work on being able to not only physically separate the HSs from the FEP but also develop ways to virtually locate the honeypots into a large number of networks across the internet, without having to physically move the HS. With this next generation network, we will be able to use better sampling techniques to get data which is more representative the the Internet as a whole.