

# Know Your Enemy: Defining Virtual Honeynets

*Different types of Virtual Honeynets.*

[Honeynet Project](#)

<http://www.honeynet.org>

Last Modified: 27 January, 2003

Over the past several years [Honeynets](#) have demonstrated their value as a security mechanism, primarily to learn about the tools, tactics, and motives of the blackhat community. This information is critical for organizations to better understand and protect against the threats they face. One of the problems with Honeynets is they are resource intensive, difficult to build, and complex to maintain. Honeynets require a variety of both physical systems and security mechanisms to effectively deploy. However, the Honeynet Project has been researching a new possibility, virtual Honeynets. These systems share many of the values of traditional Honeynets, but have the advantages of running all the systems on a single system. This makes virtual Honeynets cheaper to build, easier to deploy, and simpler to maintain.

## What is a Honeynet

Honeynets are one type of [honeypot](#). A honeypot is a *resource who's value is in being probed, attacked or compromised*. A Honeynet is a high-interaction honeypot, meaning it provides real operating systems for attackers to interact with. This high interaction can teach us a great deal about intruders, everything from how they break into systems to how they communicate and why they attack systems. Honeynets accomplish this by building a network of systems. This network is highly contained, where all inbound and outbound traffic is both controlled and captured. Each system within the network is really a honeypot, a system designed to be attacked. However, these honeypots are fully functional systems, the same found in most organizations today. When these systems are attacked, Honeynets capture all of the attacker's activity. This information then teaches a great deal about the threats we face to day. For the technical details on Honeynets, you are encouraged to review [Know Your Enemy: Honeynets](#). This paper describes different ways of building Virtual Honeynets. This is not meant to be a HOWTO on building Virtual Honeynets. Detailed HOWTO's will follow. From this point on, it is assumed you have a understanding of Honeynet technologies and their requirements, specifically Data Control and Data Capture.

## Virtual Honeynets

So, what is a Virtual Honeynet? Its a solution that allows you to run everything you need on a single computer. We use the term virtual because it all the different operating systems have the 'appearance' to be running on their own, independent computer. These solutions are possible because of virtualization software that allows running multiple operating systems at the same time, on the same hardware. Virtual Honeynets are not a radically new technology, they simply take the concept of

Honeynet technologies, and implement them into a single system. This implementation has its unique advantages and disadvantages over traditional Honeynets.

The advantages are reduced cost and easier management, as everything is combined on a single system. Instead of taking 8 computers to deploy a full Honeynet, you can do it with only one. However, this simplicity comes at a cost. First, you are limited to what types of operating system you can deploy by the hardware and virtualization software. For example, most Virtual Honeynets are based on the Intel X86 chip, so you are limited to operating systems based on that architecture. You most likely cannot deploy an Alteon switch, VAX, or Cray computer within a virtual Honeynet. Second, virtual Honeynets come with a risk. Specifically, an attacker may be able to compromise the virtualization software and take over the entire Honeynet, giving them control over all the systems. Last, there is the risk of fingerprinting. Once the badguys have hacked the systems within your virtual Honeynet, they may be able to determine the systems are running in a virtual environment.

We have broken Virtual Honeynets into two categories, Self-Contained and Hybrid. Of the two, Self-Contained are the more common. We will first define these two different types, and then cover the different ways virtual Honeynets can be deployed.

### **Self-Contained Virtual Honeynet**

A Self-Contained Virtual Honeynet is an entire Honeynet network condensed onto a single computer. The entire network is virtually contained on a single, physical, system. A Honeynet network typically consists of a firewall gateway for Data Control and Data Capture, and the honeypots within the Honeynet. You can see a [Diagram of such a deployment here](#). Some advantages of this type of virtual Honeynet(s) are:

- Portable, Virtual Honeynets can be placed on a laptop and taken anywhere. The Honeynet Project demonstrated this functionality at the Blackhat Briefings in August, 2002
- Plug and Catch, You can take the one box and just plug it in to any network and be ready to catch those blackhats. This makes deployment much easier, as you are physically deploying and connecting only one system.
- Cheap in money and space, You only need one computer, so it cuts down on your hardware expenses. It also has a small footprint and only takes one outlet and one port! For those of us with very limited space and power, this is a life saver.

There are some disadvantages:

- Single Point of Failure, If something goes wrong with the hardware, the entire honeynet could be out of commission.
- High Quality computer, Even though a Self-Contained Honeynets only require one computer, it will have to be a powerful system. Depending on your setup, you may need a great deal of memory and processing power.
- Security, Since everything might be sharing the same hardware, there is a danger of an attacker getting at other parts of the system. Much of this depends on the virtualization software, which will be discussed later.
- Limited Software, Since everything has to run on one box, you are limited to the software you

can use. For instance, its difficult to run Cisco IOS on an Intel chip.

## **Hybrid Virtual Honeynet**

A Hybrid Virtual Honeynet is a combination of the Classic Honeynet and Virtualization software. Data Capture, such as firewalls, and Data Control, such as IDS sensors and logging, are on a seperate, isolated system. This isolation reduces the risk of compromise. However, all the honeypots are virtually run on a single box. You can see a [diagram of such a deployment here](#). The advantages to this setup are:

- **Secure:** As we saw with the Self-Contained Virtual Honeynets, there is a danger of an attacker getting to the other parts of the honeynet (like the firewall). With Hybrid Virtual Honeynets, the only danger would be that the attacker accessing to the other honeypots.
- **Flexible:** You are able to use a myriad of software and hardware for the Data Control and Data Capture elements of the Hybrid network. An example would be that you can use the OpenSnort sensor on the network, or a Cisco pix appliance. You can also run any kind of honeypot you want because you can just drop another computer on the network (in addition to your Virtual Honeypot's box).

Some disadvantages are:

- **Not portable,** Since the honeynet network will consist of more then 1 box, it makes it more difficult to move.
- **Expensive in time and space,** You will have to spend more in terms of power, space and possibly money since there is more then one computer in the network.

## **Possible Solutions**

Now that we have defined the two general categories of virtual Honeynets, let's highlight some of the possible ways to implement a virtual Honeynet. Here we outline three different technologies will that allow you to deploy your own. Undoubtedly there are other options, such as [Bochs](#), however the Honeynet Project has used and tested all three methods. No solution is better then the other. Instead, they each have their own unique advantages and disadvantages, its up to you to decide which solution works best. The three options we will now cover are VMware Workstation, VMware GSX Server, and User Mode Linux.

### **VMware Workstation**

VMware Workstation is a long used and established Virtualization option. Its designed for the desktop user and is available for Linux and Windows platforms. Advantages to using VMware Workstation as a Virtual Honeynet are:

- **Wide range of Operating System support,** You are able to run a variety of operating systems within the virtual environment (called GuestOS's), including Linux , Solaris, Windows and FreeBSD Honeypots.
- **Networking options,** Workstation provides two ways to handle networking. The first is Bridged, which is useful for Hybrid Virtual Honeynet Networks because it lets a honeypot use the

computer's card and appear to be any other host on the honeynet network. The second option is Host-Only Networking, this is good for Self-Contained Virtual Honeynets because you are able to better control traffic with a firewall.

- VMware Workstation creates an image of each Guest Operating System. These images are simply a file, making them highly portable. This means you can transfer them to other computers. To restore a honeypot to its original condition, you can just copy a backup into its place.
- Ability to mount VMware virtual disk images. You are able to mount a VMware image just like you would a drive using `vmware-mount.pl`.
- Easy to use. VMware Workstation comes with a graphical interface (both Windows and Linux) that makes installing, configuring, and running the operating systems very simple.
- As a commercial product, VMware Workstation comes with support, upgrades, and patches.

Some disadvantages are:

- Cost, VMware workstation costs \$300 per license. This might be a bit much for the hobbyist, or the unemployed student.
- Resource requirements, VMware Workstation must run under an X environment, and each Virtual Machine will need its own window. So on top of the memory you allocate for the GuestOS's, you have the overhead of the X system.
- Limited amount of GuestOS's, With VMware you can only run a small number of Virtual Machines ~(1-4). This might make for a limited HoneyNet.
- Closed Source, since VMware is closed source, you can't really make any custom adjustments.
- Fingerprinting. It may be possible to fingerprint the VMware software on a honeypot, especially if the "VMware tools" are installed on the systems. This could give the honeypots away to the blackhat. However, VMware Workstation does have options that can make fingerprinting more difficult, such as the ability to set the MAC address for virtual interfaces..

VMware products also have some nice features, like the ability to suspend a Virtual Machine. You are able to pause the VM, and when you take it out of suspension, all the processes go on like nothing happened. Once a system was compromised and the intruder started an ICMP fragment attack. The intruder was also logged into IRC servers. We did not want to cut the connection because we would lose valuable information. So we suspended the VM, adjusted the firewall to block the attack, then brought the VM back up. An interesting use of VMware, and other virtualization software too, is the ease and speed of bringing up VM's. Once a honeynet is compromised, and we learned as much as we can from it, we want to start over. With a Virtual HoneyNet, all we have to do is copy files or use the undoable disk or nonpersistent disk feature in VMware Workstation to discard any changes made. Another feature of VMware Workstation is the ability to run several networks behind the HostOS. So if you only have 1 box, you can have your honeynet and personal computers all on the one box without worrying about data pollution on either side. If you would like to learn more about VMware and its capabilities for honeypot technology, check out Kurt Seifried's excellent paper [HoneyPotting with VMware - The Basics](#). Also, [Monitoring VMware HoneyPots](#) by Ryan Barnett.

## [VMware GSX Server](#)

The VMware GSX Server is a heavy-duty version of VMware Workstation. It is meant for running many

higher end servers. As we will see, this is perfect for use as a Honeynet. GSX Server currently runs on Linux and Windows as a Host OS. If you would like to learn more about deploying Virtual Honeynets on GSX, check out the paper [Know Your Enemy: Learning with VMware](#). Advantages:

- Wide range of Operating System support, GSX Server support Windows (including 95, 98, NT, 2000, XP and .NET server), various Linux distributions, and potentially BSD and Solaris (Not officially supported).
- Networking, includes all of the options that Workstation has.
- No X means more GuestOS's, GSX Server does not need X running in order to have VMware running. This allows you to run many more GuestOS's at the same time. However, it does require that some of the X libraries be installed if the Host is running Linux.
- Web Interface, GSX Server can be managed through a web page interface. GuestOS's can be started, paused, stopped and created via the web page.
- Remote terminal, this is one of the best features of GSX Server. Through the web page and with some VMware software, you can remotely access the GuestOS's as if you were sitting at the console. You are able to do things like remote installs and checking out the system without generating traffic on the honeynet.
- Ability to mount VMware virtual disk images, just like in Workstation.
- VMware GSX Server supports more host memory (up to 8GB), more CPU's (up to 8), and more memory per virtual machine (2GB) than VMware Workstation.
- Includes a Perl API to manage GuestOS's.
- Similar to Workstation, GSX Server is a supported product, including patches and upgrades.

Some disadvantages are:

- Cost, a GSX Server license will run around \$3,500.
- Limited types of GuestOS's, Operating systems like Solaris X86 and FreeBSD are not officially supported (however you may be able to install them). This can limit the diversity of your Honeynet.
- Memory hog, GSX Server recommends > 256meg just to run the GSX Server software. GUI based Operating Systems, such as Windows XP, require another 256 MG for each instance.
- Closed Source, just like Workstation.
- Fingerprinting. It may be possible to fingerprint the VMware software on a honeypot, especially if the "VMware tools" are installed on the systems. This could give the honeypots away to the blackhat. However, like Workstation there are configuration options that can reduce that risk.

VMware also makes an [VMware ESX Server](#) server. Instead of being just a software solution, ESX Server runs in hardware of the interface. ESX Server provides its own virtual machine OS monitor that takes over the host hardware. This allows more granular control of resources allocated to virtual machines, such as CPU shares, network bandwidth shares and disk bandwidth shares and it allows those resources to be changed dynamically. This product is even higher end than GSX Server. Some of its features are: It can support multiple processors, more concurrent virtual machines (up to 64 VMs), more host memory (up to 64GB) and more memory per virtual machine (up to 3.6GB) than GSX Server.

## [User Mode Linux](#)

User Mode linux is a special kernel module that allows you to run many virtual versions of linux at the same time. Developed by Jeff Dike, UML gives you the ability to have multiple instances of Linux, running on the same system at the same time. It is a relatively new tool with great amounts of potential. You can learn in detail how to deploy your own UML Honeynet in the paper [Know Your Enemy: Learning with User-Mode Linux](#). Some advantages to using User Mode Linux are:

- Free and open source, you have access to the source code.
- Small footprint and fewer resource requirements. User Mode Linux does not need to use X. It can also run extensive amount of systems with little memory.
- Ability to create several virtual networks and even create virtual routers all inside the original virtual network.
- Supports both [bridging](#) and [networking](#), similar to VMware.
- UML has the ability to log keystrokes through the GuestOS kernel. The keystrokes are logged right on the HostOS, so there are no issues with how to get the keystrokes off the honeypot in a stealth way.
- UML comes with pre-configured and downloadable file systems, making it fast and easy to populate your Honeynet with honeypots. Like VMware, these file system images are mountable.
- You can access UML consoles in a wide variety of ways, including through pseudoterminals, xterms, and portals on the host which you can telnet to. And there's always screen. Run UML inside screen, detach it, and you can log in to the host from anywhere and attach it back.

Some disadvantages are:

- Currently only supports Linux Virtual Machines, however a port to Windows is under development.
- As a new tool, there are some bugs, documentation, security issues.
- There is no GUI interface, currently all configuration and implementation is done at the command line. Steeper learning curve.
- As an OpenSource tool, there is no official or commercial support.
- Similar to VMware, it may be possible to fingerprint a UML Honeynet due to the virtualization software. However, the maintainer Jeff Dike is taking measures to reduce that risk, such as modifications to the /proc on the Guest operating systems.

## Conclusion

The purpose of this paper was to define what a Virtual Honeynet is, the different types, and options for deploying them. Virtual Honeynet take the technology of a Honeynet and combine them on a single system. This makes them cheaper to build, easier to deploy, and simpler to maintain. However, they also share common disadvantages, including a single point of failure and limitation with both the physical hardware and virtual software. Its up to you to decide which solution is best for your environment. In the future, we intend to develop documentation detailing how to deploy these technologies.

## The Honeynet Project