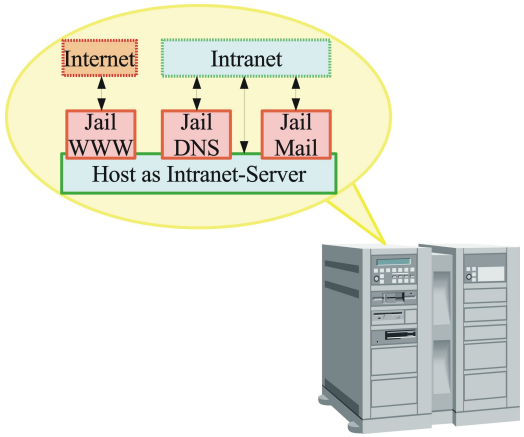
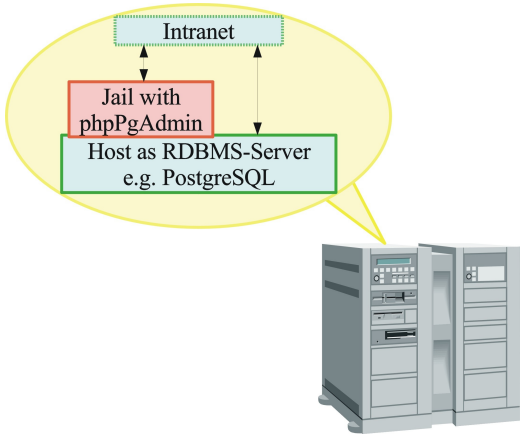
## Usage examples

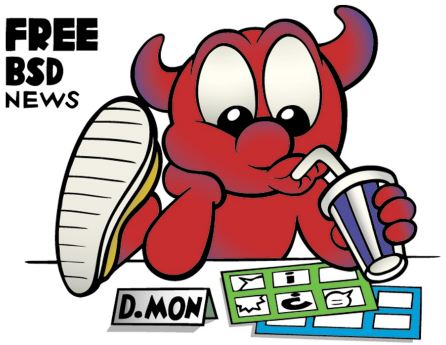
Servers for small and medium businesses need to be cost-effective and secure. Running services like HTTP, Mail and DNS on multiple machines is very expensive. Using Jails you can buy just one high-quality machine and run each service in its own Jail. Thereby saving costs in hardware, maintenance, power and cooling.




Running databases in a secure way is very easy using Jails. The administrator installs a web-based frontend in a Jail and the database in another Jail. This way both services are segregated and the compromise of one system does not affect the other.

Testing network services using Jails is also very easy and cheap. Install a service in Jail and delete or maintain the Jail when testing is complete.




## Further Information

With this short overview we hope we have been able to show you how FreeBSD Jails can work for you too.

The Wikipedia also has an article on Jails:

http://en.wikipedia.org/wiki/FreeBSD_jail

The manual page for Jails is very important:

man jail

If you're more interested in FreeBSD's security features, please see the following flyer too:

http://www.allbsd.de/src/Flyer/FreeBSD/PDF/flyer-en-fbsd-security.pdf

General information on FreeBSD is to be found here:

http://www.FreeBSD.org/doc/en_US.ISO8859-1/books/handbook/

If you need any help you may visit:

www.bsdforums.org

# FreeBSD
## Jails

## What are Jails?

Jails may be described as ‘FreeBSD within FreeBSD’. A Jail goes way beyond a simple chroot-environment. A Jail is a completely independent FreeBSD system within a FreeBSD system, whereby restrictions and on processes and child-processes are applied. Processes in a Jail can not access processes of the host system.

No hardware is emulated like with VMWare, nor is a kernel like with XEN. A Jail shares resources with the host system. The host system is not affected by changes within a Jail. The Jail system is far superior in performance to other virtualization techniques, especially when multiple virtual instances are running.

As soon as an administrator moves within a Jail he can see a complete FreeBSD system that can be administered just like any other FreeBSD system.

FreeBSD Jails are part of the Base System and can significantly improve system security. Besides locking in a whole operating system you can also just jail a process or service. Since a Jail behaves like a complete operating system, you can install all the software available in Ports.

## Server Processes In Jails

Especially daemons like DNS, HTTP, SMTP, POP3, IMAP, FTP and many other ones had vulnerabilities which a hacker could use to access gain access to a system. Even when diligently patching a system and running it behind a firewall, a risk remains. To minimize this risk it's advisable to lock a daemon in a Jail.

Whether you put all daemons in one Jail or each daemon in its own is up to you. Even after a Jail has been broken into, your host system remains intact.

## Security through Jails

When (and if) an intruder is detected you can never be sure what has been changed or which backdoor has been installed. It's very hard to detect whether only the service is compromised or also the operating system. A compromised service and system also means reduced availability, because the whole system will probably have to be rebuilt. If a Jail's system has been compromised you just rebuild the Jail and keep the rest as it is.

Stop the Jail, overwrite the directory of the compromised Jail with your backup copy of the Jail (no backup – no mercy) and restart the Jail. This shouldn't take much longer than a minute and your service is up and running again.

Then you can do an offline forensic examination of the compromised Jail.

As you can see, besides using a good operating system, using a firewall, an Intrusion Detection System (IDS) and other security techniques, using a Jail can really help maintain a high availability of a service.

## Your Server as Fort Knox

Firewalls are the first barrier, the operating system the second, Jails the third and FreeBSD's Security Flags offer yet another one.

It's bad enough when an intruder compromises a system and renders all that's in it untrustworthy.

But what if an intruder gains root privileges within the Jail?

Even then the host system remains secure. But you can further harden your system using Security Flags and Securelevels. Using Securelevels you can prevent access to firewall rule modification, kernel module loading, writing to disk etc.

For more information regarding Security Flags and Securelevels, please refer to the FreeBSD security flyer.

## Added value using Jails

1. Multiple virtual servers can be run on one physical machine. You can actually create a whole demilitarized Zone (DMZ) using Jails on one physical machine.

2. Server services are often complex and require high maintenance. Therefore multiple administrators require access. They are either given root privileges or extensive sudo configurations need to be setup. Both methods have their problems. With Jails however, you can give each administrator full privileges for his service in his own Jail. Thus avoiding multiple root accounts that are not monitored and logged.

3. If you require a test-bed for developers or certain services you don't need to build a whole new server. Simply create a Jail and let the developer do what she wants without running the risk of compromising the host system.

4. In training courses attendees don't need to share a server, each attendee can use his own Jail and train in there. When the course is over no new server needs to be installed, you only need to overwrite the used Jails with new ones.

5. If you want to offer your customers root-shells you don't need to buy hardware, simply create a Jail for each customer.

6. An interesting application is to segregate a database from its user interface by putting the DB and the X-Windows application each in its own Jail.

Jails are obviously a cheap and easy way to more security, more control, lower costs and more effective use of available hardware.

## Some Technical Details

The implementation of Jails is not restricted to userland, it's an integral part of the kernel.

A Jail requires approximately 140 MB disk space, but can of course be more depending on what you put in it. On the other hand you can also strip it down and delete anything you don't need to run the actual jailed service.

From the system administrator's point of view a Jail is handled like any other system. You can log into it via SSH. From the point of view of the host system a Jail is only directory with a complete minimal FreeBSD installation on which you have access from the host system.

To work with the Jail you don't have to log into Jail. With 'jexec' you can start and stop daemons and programs, just look at running processes or which users are logged in.

In the host system's process list every Jail-process is marked with a 'J'. That way you can immediately see which processes are running in Jails on the host.

A Jail is subject to certain restrictions which you can influence via the host system's rc.conf(5) and sysctl(8). Normally raw sockets (eg. for ping and traceroute) are not allowed, that can be changed however, as can setting the Jail's host name from within the Jail.

One of the biggest restrictions with Jails though is that only one IP address per Jail is allowed, and the address has to be set with netmask '/32'.

A lot of improvements are in the making as of lately: ipv6-support, more than 1 IP per jail and resource limits.

The success has led to a project named sysjail for NetBSD; OpenBSD and MirOS, please see http://sysjail.bsd.lv/