WIP/CFT: mkjail

BY TOM JONES

Jails are one of FreeBSDs most famous features. They enable a single server to offer many services while maintaining clear separation between the applications the server runs. While jails were one of the first uses of containerization, FreeBSD didn't end up with the majority of the mindshare and, instead, tools such as docker won the day.

Part of the reason for this can probably be attributed to the high-quality tools in a normal FreeBSD installation being more than enough to manage a fleet of services on a single host. Over time, tooling has gotten friendlier all round, while in FreeBSD, there is a lack of a single, straight-forward management and automation framework for jail management tasks. This can be seen through the sheer number of projects that aim to add an extra layer on top of the FreeBSD base tools to manage jails and their life cycles.

mkjail is not one of these jail management layers, it is, instead, a simple interface to the creation, updating and upgrading of jails. mkjail only does these three things. Management of the rest of the jail's life cycle is handled by the normal
FreeBSD jail infrastructure.

mkjail eschews a lot of the complexity that
come with other fuller-featured jail frameworks. Other frameworks allow complex hierarchies of configuration and overlay, using thin jails on top of base jails to reduce foot print a little. Instead, mkjail is a simple tool for creating jails — it creates FreeBSD systems in bottles. The first line of its source describes it well: "Lazy, dirty tool for creating fat jails."

mkjail can be simple by reducing the situations in which it works. It only offers the creation of what are called "fat" jails and every fat jail has an vidual jails.

mkjail uses a small configuration file that describes the location of the zfs datasets for the jails
mkjail will create, their location in the file system and finally the install sets which should be installed in the jails. Once a jail has been created with mkjail, it must be integrated into /etc/jails.conf in the same way as another jail managed with the base system tools.

mkjail eschews a lot of the complexity that come with other fuller-featured jail frameworks.

Jail creation can be straightforward with base system tools — bsdinstall has the jail command that can install a jail into any specified directory. mkjail expands on this functionality and handles integration with zfs tools and a clean interface to performing updates and upgrades.

mkjail shines when it comes to updating and upgrading jails, turning these operations into single commands. Because mkjail only runs with fat jails, none of the management issues with thin jails appear when doing updates and upgrades.

WIP/CFT: mkjail

mkjail[1] was initially developed by Mark Felder, was put on github by BSDCan's Dan Lagille and has had contributions from Andrew Fyfe. The project is developed on github (https://github.com/mkjail/mkjail) and is still very young — the github import of the source tree to github goes back to mid 2021.

mkjail is intended to be a small tool to enhance and ease jail creation, updates, and upgrades. It has been written as a couple of small shell scripts and is small enough that it could be integrated into the FreeBSD base system.

mkjail is happy to accept contributions through its github in the form of pull requests for code or documentation or with issues there to discuss bugs and new features. mkjail is entirely written in sh which makes contributing code relatively straight forward.

As mkjail is a young project, it can benefit a lot from testing, both of its sub commands as implemented, but also through exposure to your workflow. The best way to contribute to mkjail is to try it out and feedback any issues you had with the documentation, the code or any pain points where small additions would make sense to enable your workflow.

TOM JONES wants FreeBSD-based projects to get the attention they deserve. He lives in the North East of Scotland and offers FreeBSD consulting..