Jitsi on OpenBSD

Puffy presents video conferencing

Philipp Buehler
sysfive.com GmbH
Hamburg, Germany
pb-openbsd@sysfive.com

***Abstract*—This paper will cover all bits and bolts to fully understand the components at play, their intercommunications and how this knowledge can be used to create a Jitsi-on-OpenBSD setup that features a restricted (compartmentalized) setup using dedicated machines or -as shown- VMM based VMs, where each VM runs only one of the components.**

It'll be documented what's necessary to create a sensible pf.conf on each VM and how to add reverse proxy (relayd, haproxy) for distribution of workload.

Also covering pitfalls/hints along underlying components and what to lookout for on the client/browser side for interopability.

***Index Terms*—Jitsi, OpenBSD, VMM**

I. Introduction

Jitsi and OpenBSD are both not covered much as a documented setup. Installation documents for Jitsi are almost always about Linux OS (and there mostly Debian) and do not cover some internals. The reference documentation on the other hand can be very overwhelming.

There is some FreeBSD "all in one" port (package) with no explanation and it cannot be used to install core components (only) on different nodes.

This documentation is to show a distributed install on OpenBSD using pre-packages and need-to-function (minimum) firewall settings (‘pf.conf‘).

II. Riddles

Both major players show obstacles that have to be overcome to gain a functioning installation.

A. Jitsi

A "full blown" Jitsi installation can consist over over a dozen components and all the necessary networking/firewalling configuration can be exhaustive. Any possible discovery magic is not documented.

Some configuration snippets are undocumented and tend to make the understanding poorer not better. Worst example are necessary DNS settings and ‘nginx.conf.‘

The typical answer to be found on asking question is to use the official ‘all-in-one‘ Debian VM.

B. OpenBSD

This also leds to the question if it's possible to run a (core) Jitsi installation on OpenBSD only or if there's need be for Linux (VM).

Also it's a bit difficult to find example-based documentation on VMM, e.g. for using VMM as the core router, too. (Combination ‘vm.conf‘+‘pf.conf‘s).

Can we scale the installation horizontally and how to use Java based applications with ‘rcctl‘.

III. Components

A. OpenBSD

In this example setup I make heavy use of ‘VMM‘ eco system on OpenBSD which consists of:

- ‘vmm(4) - virtual machine monitor:‘ kernel driver isolating/providing the required resources for the VMs (hypervisor)
- ‘vmd(8):‘ userland daemon to interact with ‘vmm‘
- ‘vmctl(8)‘: administrative tool to create, start/stop, .. VMs
- ‘vm.conf(5)‘: persist VMs resource configuration

B. Jitsi

A ‘core‘ (basic video conferencing) setup comprised by:

- ‘nginx(8)‘ web: serving web assets and reverse proxy BOSH or websockets
- ‘prosody(8)‘ xmpp: conference chat + internal components communication (esp.PubSub for health/discovery)
- ‘jicofo‘ JItsi COnference FOcus: room+session handling in conferences (whos talking to whom and where)
- ‘jvb‘ videobridge: mediastream (WebRTC) handlings between participants (SFU)
- ‘jibri‘ JItsi BRoadcasting Infrastructure (optional): recording + streaming conferences

IV. Architecture

XXX: insert image img/arch-openbsd.png

V. Communications

XXX: insert image img/arch-tcp.png XXX: insert image img/arch-pubsub.png

This paper, presentation slides and other directly related resources can be found on github: https://github.com/double-p/presentations/AsiaBSDCon/2022/

References

[1] OpenBSD project https://www.openbsd.org/
[2] Jitsi https://github.com/jitsi/