NetBSD on Google Compute Engine

Benny Siegert <bsiegert@{google.com, netbsd.org}>
Google, The NetBSD Foundation
## Table of Contents

<table>
<thead>
<tr>
<th>Slide 3 - 6</th>
<th>Introduction</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>What is Google Compute Engine?</td>
</tr>
<tr>
<td>Slide 7 - 9</td>
<td>Instance Metadata</td>
</tr>
<tr>
<td></td>
<td>How to automate stuff</td>
</tr>
<tr>
<td>Slide 10 - 11</td>
<td>Notes on NetBSD</td>
</tr>
<tr>
<td></td>
<td>Environment and Networking</td>
</tr>
</tbody>
</table>
What is Google Compute Engine?

Run VMs on Google infrastructure
- Great performance and network
- Worldwide presence
- Competitive pricing (per minute!)
- Batch resources available
- Can scale to large VMs, lots of VMs, …

Part of Google Cloud Platform
- Cloud Storage
- Cloud Load Balancing
- Stackdriver Monitoring and Logging
- etc.
The NetBSD Foundation, Inc. All rights reserved.
Copyright (c) 1982, 1986, 1989, 1993
The Regents of the University of California. All rights reserved.
NetBSD 7.1_RC2 (GENERIC.2017022107392)
total memory = 58091 MB
avail memory = 57252 MB
kern.module.path=/stand/amd64/7.1/modules
mainbus0 (root)
ACPI: RSDP 0xf4940 000014 (v00 Google)
ACPI: CPU 0x0000000000000000 (v01 Google)
ACPI: FAC 0xbfffffff (v02 Google)
ACPI: SSDT 0xbfffd0d0 002022 (v01 Google)
ACPI: FACS 0xbfffffff (v01 Google)
ACPI: SSDT 0xbfffcf00 000266 (v01 Google)
ACPI: WAI 0xbfffffff (v01 Google)
ACPI: All ACPI Tables successfully acquired
iapic0 at mainbus0 apid 0
cpu0 at mainbus0 apid 8: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu1 at mainbus0 apid 1: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu2 at mainbus0 apid 2: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu3 at mainbus0 apid 3: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu4 at mainbus0 apid 4: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu5 at mainbus0 apid 5: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu6 at mainbus0 apid 6: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu7 at mainbus0 apid 7: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu8 at mainbus0 apid 8: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu9 at mainbus0 apid 9: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu10 at mainbus0 apid 10: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu11 at mainbus0 apid 11: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu12 at mainbus0 apid 12: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu13 at mainbus0 apid 13: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu14 at mainbus0 apid 14: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu15 at mainbus0 apid 15: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu16 at mainbus0 apid 16: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu17 at mainbus0 apid 17: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu18 at mainbus0 apid 18: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu19 at mainbus0 apid 19: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu20 at mainbus0 apid 20: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu21 at mainbus0 apid 21: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu22 at mainbus0 apid 22: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu23 at mainbus0 apid 23: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu24 at mainbus0 apid 24: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu25 at mainbus0 apid 25: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu26 at mainbus0 apid 26: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu27 at mainbus0 apid 27: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu28 at mainbus0 apid 28: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu29 at mainbus0 apid 29: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu30 at mainbus0 apid 30: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu31 at mainbus0 apid 31: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu32 at mainbus0 apid 32: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu33 at mainbus0 apid 33: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu34 at mainbus0 apid 34: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu35 at mainbus0 apid 35: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu36 at mainbus0 apid 36: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu37 at mainbus0 apid 37: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu38 at mainbus0 apid 38: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu39 at mainbus0 apid 39: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu40 at mainbus0 apid 40: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu41 at mainbus0 apid 41: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu42 at mainbus0 apid 42: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu43 at mainbus0 apid 43: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu44 at mainbus0 apid 44: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu45 at mainbus0 apid 45: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu46 at mainbus0 apid 46: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu47 at mainbus0 apid 47: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu48 at mainbus0 apid 48: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu49 at mainbus0 apid 49: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu50 at mainbus0 apid 50: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu51 at mainbus0 apid 51: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu52 at mainbus0 apid 52: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu53 at mainbus0 apid 53: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu54 at mainbus0 apid 54: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu55 at mainbus0 apid 55: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu56 at mainbus0 apid 56: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu57 at mainbus0 apid 57: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu58 at mainbus0 apid 58: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu59 at mainbus0 apid 59: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
cpu60 at mainbus0 apid 60: Intel(R) Xeon(R) CPU @ 2.30GHz, id 0x306f0
acpi0 at mainbus0: Intel ACPI 20131218
Creating NetBSD images for Google Compute Engine

This repository holds tools to build a NetBSD image for use on Google Compute Engine (GCE). GCE is part of the Google Cloud Platform.

Running `make.bash`

`make.bash` can be run under a GNU/Linux, BSD or macOS operating system. To run the script, you need a few things to be installed:

- bash
- qemu
- cdttools
- GNU tar ([http://pkgsrc.se/archivers/tar](http://pkgsrc.se/archivers/tar))
- GNU coreutils ([http://pkgsrc.se/sysutils/coreutils](http://pkgsrc.se/sysutils/coreutils))
- Python 2.7
- python-support ([http://pkgsrc.se/sysutils/py-support](http://pkgsrc.se/sysutils/py-support))
Getting Started With NetBSD

There are no official NetBSD images yet, you have to create your own.

- uses Anita to stage an installation in qemu, packs up the image
- You upload the .tar.gz to a Cloud Storage bucket
- You create a GCE image from the .tar.gz file
- Launch VMs based on the image!

The Google Cloud SDK is not available for NetBSD :(  
- You can use the web interface ([console.cloud.google.com](http://console.cloud.google.com))  
  or Cloud Shell.
Instance Metadata

Network interfaces
- nic0:

Firewalls
- Allow HTTP traffic
- Allow HTTPS traffic

Network tags

Boot disk and local disks
- Name: netbsd-gce-services
  - Size (GB): 4
  - Type: Standard persistent disk
  - Mode: Boot, read/write

- Delete boot disk when instance is deleted

Additional disks (Optional)

Availability policies
- Preemptibility
  - Off (recommended)
- Automatic restart
  - Off

On host maintenance
- Migrate VM instance (recommended)

Custom metadata
- my-key: my-value

SSH Keys
- Block project-wide SSH keys
  - When checked, project-wide SSH keys cannot access this instance (Learn more)

You have one SSH key
- Show and edit
```bash
$ curl \n  http://metadata.google.internal/computeMetadata/v1/instance/attributes/ \
  -H "Metadata-Flavor: Google"
my-key

$ curl \n  http://metadata.google.internal/computeMetadata/v1/instance/attributes/my-key \
  -H "Metadata-Flavor: Google"
my-value
```
Instance Metadata (cont’d)

GCE services provided through metadata:

- startup script on instance creation
- ssh key exchange
- automatic user creation for SSH logins (!)
- etc.

Supported by a set of daemons on the host (pkgsrc.se/sysutils/py-google-compute-engine).

- Porting to NetBSD is ongoing, at the moment none of them run.
- Should they be on by default? (I think yes)
Networking

All VMs in a given project are in the same internal /16 (?) subnet. DHCP responses use Classless Static Routes.

**Example:** own IP 10.240.0.2/32
- Static route to 10.240.0.1/32 direct via the interface
- Default route via 10.240.0.1

External IPs are static or dynamic; SDN takes care of routing.

- dhclient only supports this on Linux.
- dhcpcd had a bug that created wrong routes.
  - fixed 2017-09-05 in -current
  - fixed 2017-09-10 in -8
The Environment

- Virtualization is based on KVM. No Xen PV, plain NetBSD/amd64.
- Network is vioif
- Storage (Persistent Disk) is vioscsi
  - NetBSD-7.0 shipped without a vioscsi driver:(
- NetBSD-7 is ... crashy when using disk at all
  - Use NetBSD-8_BETA or -current!
Thank you!

Now go and try it out!
The world’s first connected ketchup dispenser isn’t going to build itself.