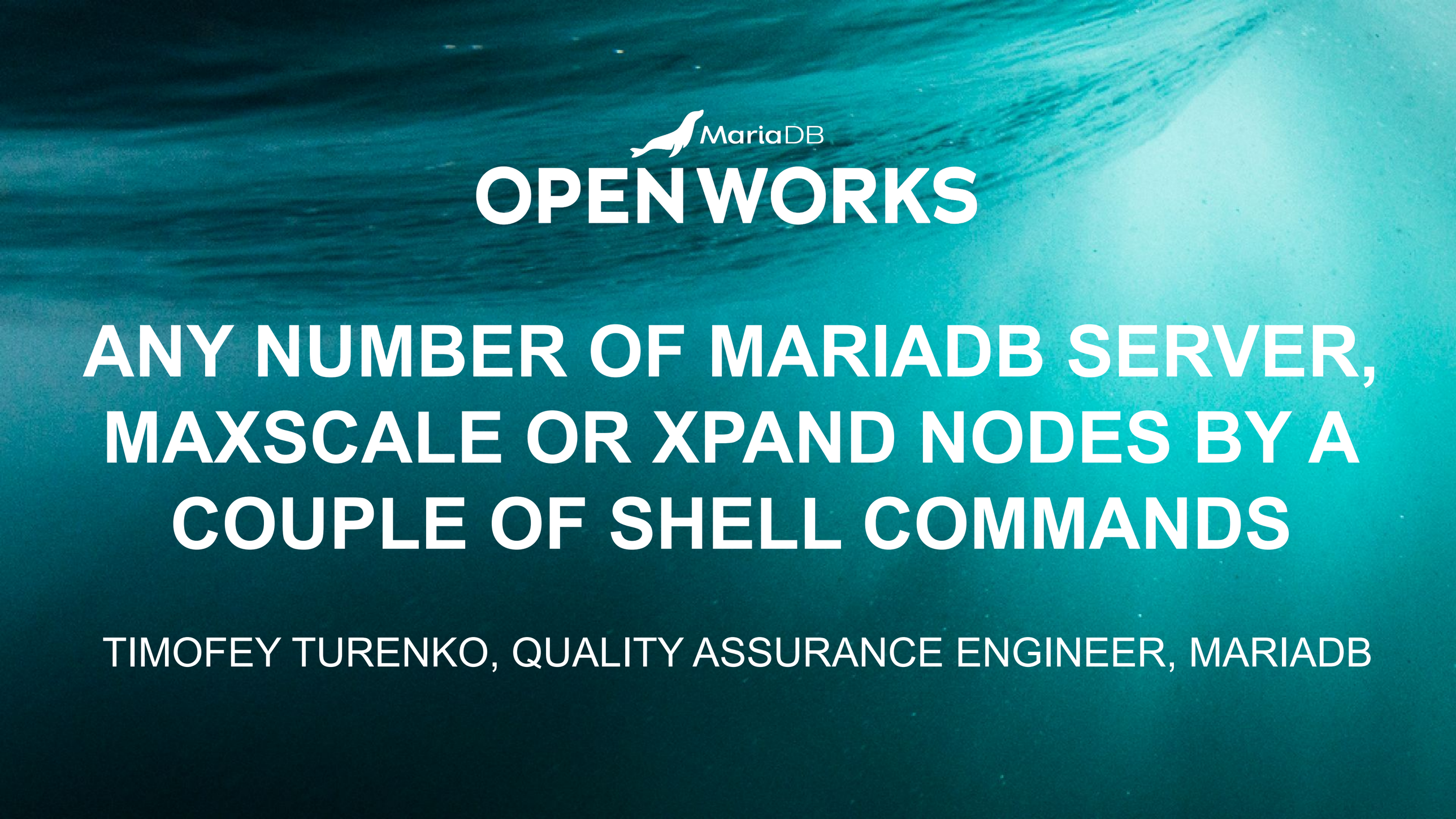

MariaDB
OPENWORKS
BE UNSTOPPABLE

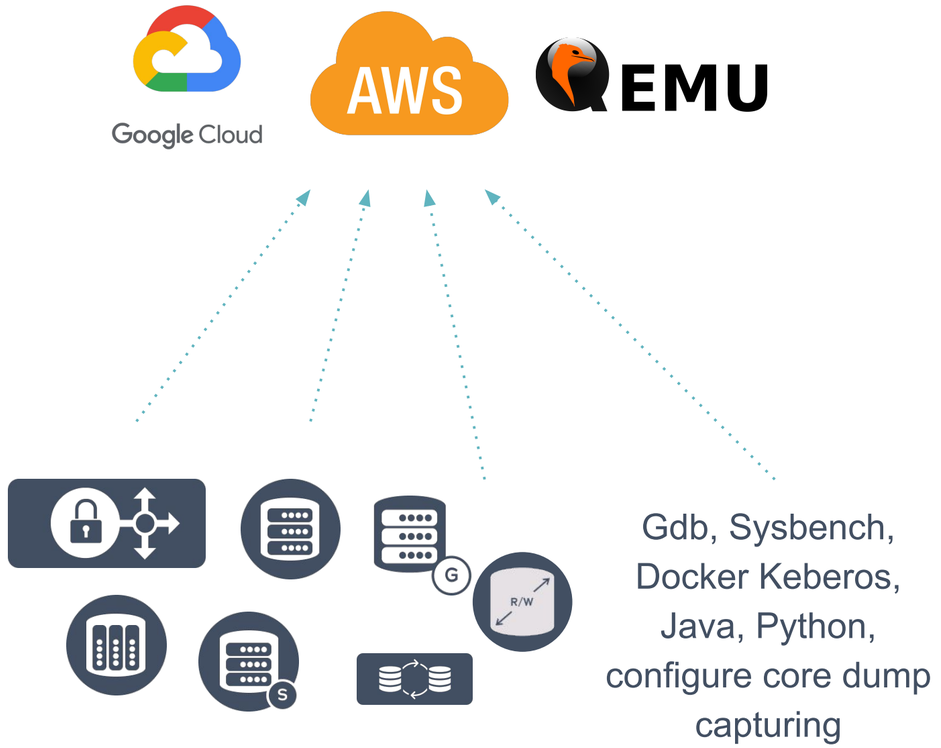


ANY NUMBER OF MARIADB SERVER, MAXSCALE OR XPAND NODES BY A COUPLE OF SHELL COMMANDS

TIMOFEY TURENKO, QUALITY ASSURANCE ENGINEER, MARIADB

MDBCI TOOL

- Manage Virtual Machines
  - Amazon Web Services
  - Google Cloud Platform
  - Qemu/Libvirt
- Install
  - MariaDB server (Community / Enterprise)
  - MariaDB plugins
  - Connectors
  - Xpand
  - Maxscale
  - Set of tools (e.g. Docker, debug tools, build environment, etc.)
- Monitor cloud resources (machines, disks, etc.)



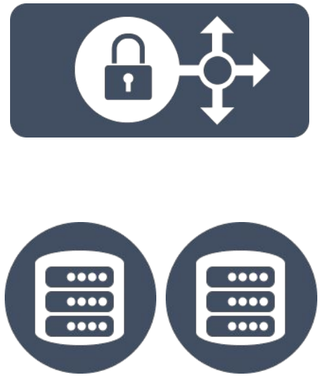
AGENDA

- Why do we need yet another VM/config management tool?
- How does it work?
- What are the system components?

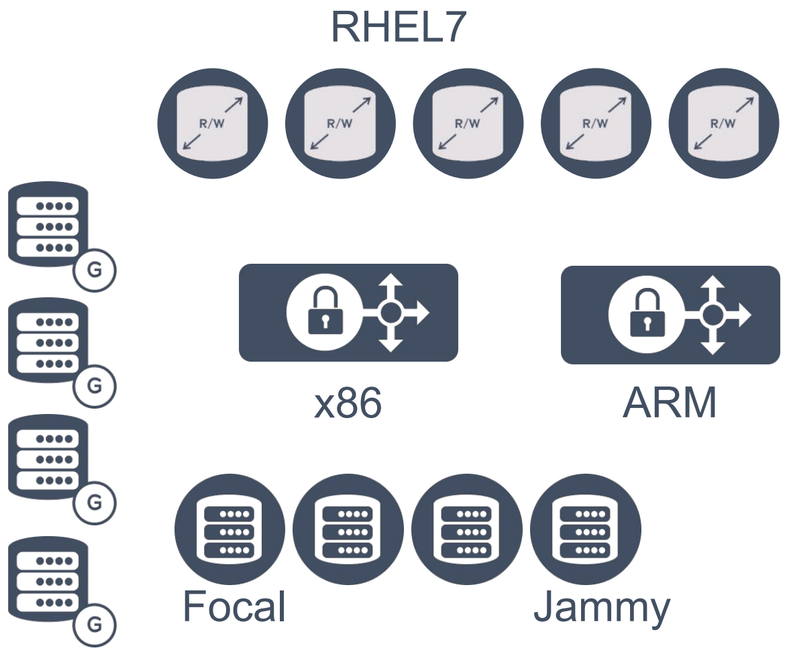
WHY DO WE NEED ANOTHER VM AND CONFIGURATION MANAGEMENT TOOL?

MariaDB

MAXSCALE TEST ENVIRONMENT STORY



MAXSCALE TEST ENVIRONMENT STORY



MAXSCALE TEST ENVIRONMENT STORY

It is simple:

It fails (very) often!

```
aws ec2 run-instances --image-id ami-xxxxxxxx --count 14
scp ...; ssh ...
cmake;make;make test
scp ...
aws ec2 terminate-instances --instance-ids ...
```

Here some stuff needs to be installed

Test have to know IP/keys of all other machines

List of instance IDs?

Or is it?

It also can fail!

- Cloud API error processing
- Instances types (CPUs, memory)
- VPC, security groups, key pairs, etc.
- Reliable way to destroy all cloud resources
- More than one cloud
- How to start only needed VMs?
- Instance provisioning (install only necessary stuff)

MariaDB

VAGRANT? TERRAFORM? ANSIBLE? CHEF?

```
terraform {
  required_providers {
    google = {
      source  = "hashicorp/google"
      version = ">= 3.65.0"
    }
  }
}
provider "google" {
  credentials =
   file("/home/timofey_turenko_mariadb_com/.config/mdbci/gcp
   -credentials.json")
  project     = "mariadb-maxscale-test"
  region      = "us-central1"
  zone        = "us-central1-c"
}
resource "google_compute_instance" "build" {
  name         = "mdbci-8rk4qdpi-1673269753-build"
  machine_type = "g1-small"
  tags         = ["iap"]
```

Cloud provider-dependent

Need to know machine types

```
  network_interface {
    network = "default"
  }
  metadata = {
    ssh-keys       = "****"
    enable-oslogin = "FALSE"
    username          = "timofey_turenko_mariadb_com"
    full-config-path = "/home/timofey_turenko_mariadb_com/vms/s"
  }
}
output "build_network" {
  value = {
    user       = "timofey_turenko_mariadb_com"
    private_ip =
   google_compute_instance.build.network_interface.0.network_ip
    public_ip =
   google_compute_instance.build.network_interface.0.network_ip
    key_file =
   "/home/timofey_turenko_mariadb_com/vms/s/maxscale.pem"
    hostname = "mdbci-8rk4qdpi-1673269753-build"
  }
}
```

Some unknown for normal tester parameter

How to get IP/user/pass/key from test code?

MariaDB

HOW DOES IT WORK (LET'S HIDE ALL DIRTY WORK)

MariaDB

MDBCI CAPABILITIES

- Create VM instances in the cloud or locally
  - using Terraform/Vagrant
  - but also with direct cloud API calls (e. g. to get list of available machine types, quotas, list of regions, etc)
  - Knows relation between cloud machine type and actual machine size (CPU, memory)
  - restart only failed VMs
  - Mount addition disk, reconfigure VM to use this disk
- Automatically scan product repositories, creates versions list
- Register and unregister RHEL and SLES VMs
- Supported platforms: actual Ubuntu, Debian, RHEL, SLES, CentOS, Rocky Linux
  - Limited Windows support (prepared image is needed)
- Keep info about all created resources to destroy them in the reliable way
- Check to lost resources (VMs, disks, security groups, key pairs)

MariaDB

HOW DOES IT WORK

User creates JSON VMs description

MDBCI “generates” Terraform or Vagrant configuration

- Determine cloud machine type based on requested CPU/memory and available machine types for this particular platform
- Check available quotas and select less loaded cloud region (via Cloud API)
- Find requested product versions (e. g. find latest 10.6.x if requested version is “10.6”)
- Store all info into internal files

MDBCI bring up and provision VMs

- “Execute” terraform configuration
- Check the status of all started resources (via Cloud API)
- Check VM status by executing simple test inside VM
- Provision VMs (execute Chef recipes)
- Restart VM/repeat provisioning for failed VM
- Write ssh_config file
- Write all “how to destroy” info into internal files

MariaDB

SIMPLE VM

```
{
  "build" :
  {
        "hostname" : "build",
        "box" : "sles_15_gcp"
  }
}
```

```
Mdbci generate my_sles_vm --template sles.json
Mdbci up my_sles_vm
Ssh build -F my_sles_vm_ssh_config
...
Mdbci destroy my_sles_vm
```

MariaDB

TWO VM EXAMPLES

```
{
  "maxscale" :
  {
    "hostname" : "maxscale",
    "box" : "sles_15_gcp",
    "products" : [
       {"name" : "maxscale",
        "version" : "22.08.2"}
    ]
  },
```

```
  "mariadb" :
  {
    "hostname" : "mariadb",
    "box" : "rhel_8_gcp",
    "products" : [
       {"name" : "mariadb",
        "version" : "10.4.26"},

       {"name" : "plugin_backup",
        "version": "10.4.26"},

       {"name" : "core_dump"}
    ]
  }
}
```

MariaDB

SYSTEM COMPONENTS

MariaDB

SYSTEM COMPONENTS

.config/mdbci/config.yaml

- Cloud credentials
- RHEL and SLES credentials
- SLES RMT proxy
- MariaDB Enterprise credentials

.config/mdbci/clustrix_license

```
set global license='{"expiration":"2023-04-10
00:00:00","maxnodes":"8","company":"MariaDB","maxcores":"32"
,"email":"****","person":"*** ***","signature":"***"}'
```

```
rhel:
  username: ****
  password: ****
mdbe:
  key: ****
aws:
  access_key_id: *****
  secret_access_key: ****
  region: eu-west-1
  availability_zone: eu-west-1a
  use_existing_vpc: true
  vpc_id: vpc-***
  subnet_id: subnet-****
gcp:
  credentials_file:
"/home/tt_mariadb/.config/mdbci/gcp-cr.json"
  project: mariadb-maxscale-test
  default_region: us-central1
  regions:
  - us-central1
  - europe-west4
suse:
  email: ***@mariadb.com
  key: '****'
  registration_proxy_url:  https://my-rmt.com
mdbci:
  image_address:
https://mdbe-ci-repo.mariadb.net/MDBCI/mdbci
  mdbci_directory:
"/home/timofey_turenko_mariadb_com/mdbci"
```

SYSTEM COMPONENTS

.config/mdbci/boxes

- MDBCI knows a number of Linux distributions
- New can be added by user – just define a new “box”

```
{
  "fedora_rawhide_libvirt": {
    "provider": "libvirt",
    "architecture": "amd64",
    "box": "fedora-rawhide",
    "platform": "fedora",
    "platform_version": "rawhide"
  },
  "fedora_rawhide_aws": {
    "provider": "aws",
    "architecture": "x86_64",
    "ami": "ami-07fecccdb8d802c9b",
    "user": "fedora",
    "vpc": "true",
    "default_machine_type": "c4.4xlarge",
    "platform": "fedora",
    "platform_version": "rawhide",

"supported_instance_types":["t2.nano","t2.micr
o","t2.small","t2.medium","t2.large","t2.xlarg
e","c4.4xlarge"]
  }
}
```

MariaDB

SYSTEM COMPONENTS

.config/mdbci/repo.d

- Contains results of repositories scans

```
mdbci generate-product-repositories

mdbci generate-product-repositories –product
maxscale
```

- Created automatically!!


```
 {
    "version": "11.0.1",
    "platform": "ubuntu",
    "platform_version": "kinetic",
    "architecture": "s390x",
    "repo":
"https://dlm.mariadb.com/repo/mariadb-serv
er/11.0.1/repo/ubuntu",
    "components": [
      "main",
      "main/debug"
    ],
    "repo_key": [
      "0x8167EE24",
      "0xE3C94F49",
      "0xcbcb082a1bb943db",
      "0xf1656f24c74cd1d8",
      "0x135659e928c12247"
    ],
    "product": "mariadb"
  }
```

NEXT STEPS

Check out these resources for more information

- SOURCE
- LATEST BINARY
- DOCUMENTATION


MariaDB
OPENWORKS
THANK YOU


MariaDB
OPENWORKS
BE UNSTOPPABLE

CONCLUSION

- An easy way to create any number of VM with MariaDB, plugins, Maxscale, Xpand – convenient way to create/test/destroy test configuration in the automatic mode
- The tool makes its best to start and provision VM despite all cloud and network failures
- No need to dig Terraform/Chef/Ansible/AWS API/GCP API documentation

```
mdbci up my_config
```

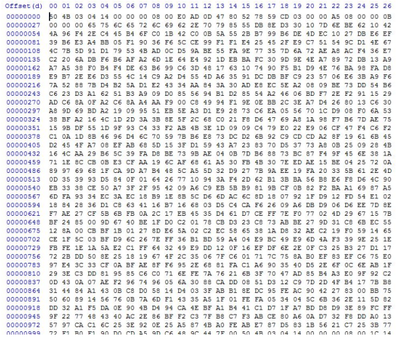
```
mdbci destroy my_config
```

https://mdbe-ci-repo.mariadb.net/MDBCI/doc/index.html

MariaDB

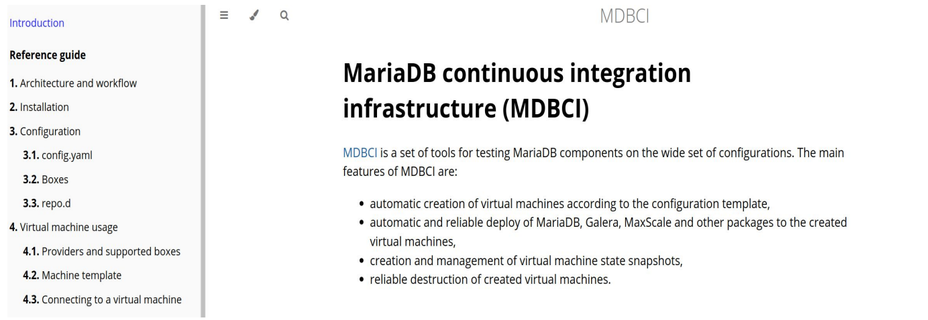
RESOURCES

- Source https://github.com/mariadb-corporation/mdbci
- Latest Binary https://mdbe-ci-repo.mariadb.net/MDBCI/mdbci
- Documentation https://mdbe-ci-repo.mariadb.net/MDBCI/doc/index.html



```
Offset(d)  00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
00000000  50 4B 03 04 14 00 00 00 08 00 E0 AD 5D 47 80 52 78 59 CD 03 00 00 A5 08 00 00 0B
00000027  00 00 00 65 75 6C 65 72 4C 69 42 2E 70 79 85 55 DB 8E D3 30 10 7D 6E BE 62 10 42
00000054  4A 96 F4 2E C4 45 B4 6F C0 1B 42 C0 0B 5A 55 2B B7 99 B6 DE 4D EC 10 27 DB E6 EF
00000081  39 B6 E3 A4 B8 05 F1 90 36 F4 5C CE 99 F1 F1 E4 25 45 2F E9 C7 51 54 9C D1 4E 67
00000108  4C 7B 5D 91 D1 79 53 4B AD 0C D5 9A BE 55 FA 9E 77 35 7D 6A 72 AE A5 AC F4 36 E7
00000135  C2 20 6A DB F6 B6 AF A2 6D 1E 64 E4 92 1D EB BA FC 30 9D 9E 4E A7 89 72 DB 13 A9
00000162  A7 A5 38 F0 B4 F6 DE 63 B6 99 C6 3D 48 17 63 10 74 90 F5 B1 D9 4E 76 BA 98 FA D8
00000189  E9 B7 2E E6 D3 55 4C 14 C9 A2 D4 55 4D A6 35 91 DC DB BF C9 23 57 06 E6 3B A9 F6
00000216  7A 52 88 7B D4 B2 5A D1 E2 43 34 AA E4 3A 30 AD E8 EC 5E A2 08 09 BE 73 DD 54 B6
00000243  C6 23 D3 A1 62 51 B3 A9 09 D0 85 56 94 B1 D2 85 54 A2 46 06 BD F7 2E F2 91 15 29
00000270  AD C6 8A 0F A2 C6 8A A4 AA F9 00 C8 49 94 F1 9E 0E BB 2C 3E A7 D4 26 80 13 C6 30
00000297  A8 8D 69 BD A2 19 09 95 51 EB 5E A3 D1 E9 28 73 CE EA 05 56 70 1C D9 08 F0 6A 53
00000324  38 BF A2 16 4C 1D 2D 3A 3B 8E 5F 2C 68 C0 21 F8 D4 47 69 A8 1A 98 F7 B6 7D AE 75
00000351  15 9B DF 55 1D 9F 93 C4 33 F2 AB 4B 3E 1D 09 09 C4 79 E0 22 E9 06 CF 47 F4 C6 F2
00000378  C1 0A 1D 8B 46 96 D4 6C 70 59 7B B6 E8 73 DC D2 6B 92 C9 CD CD A2 8F 19 61 6B 65
00000405  D2 45 4F A7 08 EF AB 68 5D 15 3F D1 59 43 A7 23 83 70 D5 37 73 A8 0B 25 09 28 4B
00000432  16 4C AA 29 B4 9C 5C 39 FA D8 BE 73 9B AE 04 0B 7D B4 88 73 BC 87 F4 9F 45 6E 38 1A
00000459  71 1E 8C C8 0B E3 CF AA 19 6C AF 68 61 A5 30 FB 4B 30 7E ED AE 15 BE 04 25 72 0A
00000486  89 97 69 68 1F CA 9D A7 B4 48 5C A5 5D 32 D9 27 7B 9A EE 19 FA 20 33 5B 61 2E 4D
00000513  DD 35 39 93 D5 84 0F 01 64 26 77 10 94 3A F4 2D 62 B1 3B BA 56 B8 E6 F8 D6 4C 90
00000540  EB 33 38 CE 50 A7 3F 2F 95 42 09 A6 C9 EB 5B B9 81 9B CF 0B 82 F2 BA A1 69 87 A5
00000567  6D FA 93 34 EC 3A EC 18 B9 1E 8B 5C D6 4D AC 6C 80 18 07 92 1F D9 12 FD 54 E1 02
00000594  18 84 28 34 D1 C6 63 41 16 B7 16 68 03 D5 C4 CA F6 26 09 A6 DB D9 06 D6 EE 7D 8E
00000621  F7 AE 27 CF 5B 6B FB 0A 2C 17 EB 45 35 D4 61 D7 CE FF 7E F0 77 02 4D 29 67 15 7B
00000648  BF 24 85 00 90 67 40 BE 1F D0 C2 01 78 CB D3 23 C8 73 AB BE 27 9D 31 C8 6B EC 55
00000675  12 8A 00 CB BF 1B 01 27 8D E6 5A 02 C2 EC 58 65 38 1A D8 32 AE C2 19 F0 59 14 65
00000702  CE 1F 5C 03 BF D9 6C 26 7E FF 36 B1 8D 59 A4 04 E9 BC 49 E9 6D 4A F3 39 9E 25 1E
00000729  FB FE 1E 1A 5A E2 C1 FF 64 32 49 E9 DD 12 CF 1E EF DF 6E 2E 0F C3 25 B3 27 D1 17
00000756  72 2B DD 50 8E 25 18 19 67 4F 2C 35 06 7F C6 01 71 7C 75 8A B0 EF 83 EF C6 75 E0
00000783  97 E4 3C 33 CF 0A BF AE 8F F6 95 2E 68 81 FA C1 A6 90 35 40 D5 2E 4F 0C 6E AB 1F
00000810  29 3E C3 DD B1 95 85 C6 C0 71 6E FE 7A 76 21 6B 3F 70 47 AD 85 B4 A3 E0 9F 92 C2
00000837  0D 43 0A 07 AE F2 96 74 96 05 4A 30 88 CA DD 08 51 D3 12 C9 7D 2D 4F B4 17 7B B8
00000864  31 44 84 A1 43 0B C8 D0 58 14 D4 03 3F AB B1 8E DC 95 FE AC 90 42 27 83 00 BB 75
00000891  50 4B 01 02 14 00 14 00 08 00 08 00 E0 AD 5D 47 80 52 78 59 CD 03 00 00 A5 08 00 00
00000918  0B 00 24 00 00 00 00 00 00 00 20 00 00 00 00 00 00 00 65 75 6C 65 72 4C 69 42 2E 70
00000945  0A 00 20 00 00 00 00 00 01 00 18 00 9F 22 77 48 43 40 AC 2E 86 BF F2 C3 7F B8 C7 F3
00000972  57 97 CA C1 6C 25 3E 92 0E 25 A5 87 4B A0 FE AB E7 87 D5 83 1B 56 21 C7 25 3B 77
```

```
71  $mdbci_exec_dir = File.expand_path(__dir__)
72
73  require_relative 'core/session'
74  require_relative 'core/out'
75  require_relative 'core/services/version'
76  require_relative 'core/models/result'
77
78  session = Session.new
79  $session = session
80  session.template_file = 'instance.json'
81  session.ipv6 = false
82
83  Dir.chdir ENV['OLD_CWD'] unless ENV['OLD_CWD'].nil?
84
85  Dir.chdir ENV['MDBCI_VM_PATH'] unless ENV['MDBCI_VM_PATH'].nil?
86
87  session.mdbci_dir = $mdbci_exec_dir
88
89  # Storing argument for the further processing
90  initial_arguments = ARGV.join(' ')
91
```



Introduction

Reference guide

1. Architecture and workflow
2. Installation
3. Configuration
  3.1. config.yaml
  3.2. Boxes
  3.3. repo.d
4. Virtual machine usage
  4.1. Providers and supported boxes
  4.2. Machine template
  4.3. Connecting to a virtual machine

MDBCI

MariaDB continuous integration infrastructure (MDBCI)

MDBCI is a set of tools for testing MariaDB components on the wide set of configurations. The main features of MDBCI are:

- automatic creation of virtual machines according to the configuration template,
- automatic and reliable deploy of MariaDB, Galera, MaxScale and other packages to the created virtual machines,
- creation and management of virtual machine state snapshots,
- reliable destruction of created virtual machines.