

Ampere Computing

# Getting started on Azure Ampere VMs with Debian using Terraform

## Table of Contents

* [Introduction](#introduction)
* [Requirements](#requirements)
* [Using the azure-ampere-vm terraform module](#X46cda69ec15395681ebfefc4b7dd980b6d40b71)
  + [Configuration with terraform.tfvars](#configuration-with-terraformtfvars)
  + [Creating the main.tf](#creating-the-maintf)
  + [Creating a cloud-init template](#creating-a-cloud-init-template)
  + [Running Terraform](#running-terraform)
  + [Logging in](#logging-in)
  + [Destroying when done](#destroying-done)

## Introduction

[Debian](https://debian.org) is one of the oldest operating systems based on the Linux kernel originally founded over 29 years ago. Since its founding, [Debian](https://debian.org) has been developed openly and distributed freely according to the principles of the GNU Project. The [Debian](https://debian.org) project is coordinated over the Internet by a team of volunteers guided by the [Debian](https://debian.org) Project Leader and three foundational documents: the [Debian](https://debian.org) Social Contract, the [Debian](https://debian.org) Constitution, and the [Debian](https://debian.org) Free Software Guidelines. New [Debian](https://debian.org) distributions are updated continually, and the next candidate is released after a time-based freeze.

For those unfamiliar with [Debian](https://debian.org), it has similarities with other Linux distrubutions, including similar package management tooling and methods, packages, and open source software stacks available for installation easily.

[Debian](https://debian.org) supports the industry standard metadata interfaces for Linux instance configurations in the cloud, [Cloud-Init](https://cloud-init.io). This allows you to automate your [Debian](https://debian.org) workloads. This also meams [Debian](https://debian.org) is perfectly suitable when using on a cloud platform.

Now personally speaking I have been working with the great team at the [Debian](https://debian.org) project for some time. For some time now The [Debian](https://debian.org) Project has used Ampere Platforms to develop, build and curate packages for each release. I’ve had the pleasure of watching thier craftmanship, while iterating, and helping achive the “it just works” experience for Aarch64 and Ampere platforms and customers who choose to build and run solutions on [Debian](https://debian.org). Recently [Microsoft announced the general availablity of Ampere Altra based processors in Azure](https://azure.microsoft.com/en-us/blog/azure-virtual-machines-with-ampere-altra-arm-based-processors-generally-available/). Additionally [Debian](https://debian.org) is one of the available operating systems for use on Ampere VMs utilzing our cloud-native Ampere(R) Altra(TM) Aarch64 processors within [Azure](https://azure.microsoft.com/en-us/).

In this post, we will build upon prevous work to quickly automate using [Debian](https://debian.org) on Ampere(R) Altra(TM) Arm64 processors using Ampere VMs within [Azure](https://azure.microsoft.com/en-us/).

## Requirements

Obviously to begin you will need a couple things. Personally I’m a big fan of the the DevOPs tools that support lots of api, and different use cases. [Terraform](https://www.terraform.io/downloads.html) is one of those types of tools. If you have seen my prevous session from [Microsoft Build on Cloud Workload Automation](https://mybuild.microsoft.com/en-US/partners/64d2f9ef-c7dd-43f7-9ee7-85ba25934a06?wt.mc_id=FP_Ampere_blog_Corp#:~:text=cloud%20native%20workload%20automation), I build a terraform module to quickly get you started using Ampere plaforms on Azure. Today we are going to use that module to launch a [Debian](https://debian.org) virtual machine while passing in some metadata to configure it.

* [Terraform](https://www.terraform.io/downloads.html)
* [Microsoft Azure Account](https://azure.microsoft.com/en-us/)
* [Microsoft Azure CLI](https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/guides/azure_cli)

## Using the azure-ampere-vm terraform module

The [azure-ampere-vm](https://github.com/amperecomputing/terraform-azure-ampere-vm) terraform module code supplies the minimal ammount of information to quickly have working Ampere A1 instances on Azure [“Always Free”](https://www.oracle.com/cloud/free/#always-free). It has been updated to include the ability to easily select [Debian](https://debian.org) by passing the appropriate parameter during usage. Addtional tasks performed by the [azure-ampere-vm](https://github.com/amperecomputing/terraform-azure-ampere-vm) terraform module.

* Operating system choice abstraction.
* Dynamically creating sshkeys to use when logging into the instance.
* Creating necessary core networking configurations for the tenancy
* Rendering metadata to pass into the Azure Ampere VMs.
* Launch N number of Azure Ampere VMs with metadata and ssh keys.
* Output IP information to connect to the instance.

### Configuration with terraform.tfvars

For the purpose of this we will quickly configure Terraform using a terraform.tfvars in the project directory.  
The following is an example of what terraform.tfvars should look like:

subscription\_id = "12345678-abcd-1234-abcd-1234567890ab"  
tenant\_id = "87654321-dcba-4321-dcba-ba0987654321"

For more information regarding how to get your Azure credentials working with terraform please refer to the following reading material:

* [Terraform Azure Provider](https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs)
* [Install the Azure CLI on Linux](https://docs.microsoft.com/en-us/cli/azure/install-azure-cli-linux?pivots=apt)
* [Install Azure CLI on macOS](https://docs.microsoft.com/en-us/cli/azure/install-azure-cli-macos)
* [Install Azure CLI on Windows](https://docs.microsoft.com/en-us/cli/azure/install-azure-cli-windows)
* [Azure CLI authentication in Terraform](https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/guides/azure_cli)

### Creating the main.tf

To use the terraform module you must open your favorite text editor and create a file called main.tf. Copy the following code which allows you to supply a custom cloud-init template at launch:

variable "subscription\_id" {}  
variable "tenant\_id" {}  
  
locals {  
 cloud\_init\_template\_path = "${path.cwd}/cloud-init.yaml.tpl"  
}  
  
module "azure-ampere-vm" {  
 source = "github.com/amperecomputing/terraform-azure-ampere-vm"   
 subscription\_id = var.subscription\_id  
 tenant\_id = var.tenant\_id  
 # Optional  
 # resource\_group = var.resource\_group  
 # rg\_prefix = var.rg\_prefix  
 # virtual\_network\_name = var.virtual\_network\_name  
 # address\_space = var.address\_space  
 # subnet\_prefix = var.subnet\_prefix  
 # vm\_size = var.vm\_size  
 location = "westus2"  
 azure\_vm\_count = 1  
 azure\_os\_image = "ubuntu2004"  
 instance\_prefix = "azure-ampere-vm-ubuntu-2004"  
 cloud\_init\_template\_file = local.cloud\_init\_template\_path  
}  
output "azure\_ampere\_vm\_private\_ips" {  
 value = module.azure-ampere-vm.azure\_ampere\_vm\_private\_ipaddresses  
}  
output "azure\_ampere\_vm\_public\_ips" {  
 value = module.azure-ampere-vm.azure\_ampere\_vm\_public\_ipaddresses  
}

### Creating a cloud init template.

Using your favorite text editor create a file named cloud-init.yaml.tpl in the same directory as the main.tf you previously created. Copy the following content into the text file and save it. With this metadata file we will add an external ‘apt’ repository for which will allow us to install the upstream Docker packages, then we will run a simple container registry on the Debian host.

#cloud-config  
  
apt:  
 sources:  
 docker.list:  
 source: deb [arch=arm64] https://download.docker.com/linux/debian $RELEASE stable  
 keyserver: pgp.mit.edu  
 key: |  
 -----BEGIN PGP PUBLIC KEY BLOCK-----  
   
 mQINBFit2ioBEADhWpZ8/wvZ6hUTiXOwQHXMAlaFHcPH9hAtr4F1y2+OYdbtMuth  
 lqqwp028AqyY+PRfVMtSYMbjuQuu5byyKR01BbqYhuS3jtqQmljZ/bJvXqnmiVXh  
 38UuLa+z077PxyxQhu5BbqntTPQMfiyqEiU+BKbq2WmANUKQf+1AmZY/IruOXbnq  
 L4C1+gJ8vfmXQt99npCaxEjaNRVYfOS8QcixNzHUYnb6emjlANyEVlZzeqo7XKl7  
 UrwV5inawTSzWNvtjEjj4nJL8NsLwscpLPQUhTQ+7BbQXAwAmeHCUTQIvvWXqw0N  
 cmhh4HgeQscQHYgOJjjDVfoY5MucvglbIgCqfzAHW9jxmRL4qbMZj+b1XoePEtht  
 ku4bIQN1X5P07fNWzlgaRL5Z4POXDDZTlIQ/El58j9kp4bnWRCJW0lya+f8ocodo  
 vZZ+Doi+fy4D5ZGrL4XEcIQP/Lv5uFyf+kQtl/94VFYVJOleAv8W92KdgDkhTcTD  
 G7c0tIkVEKNUq48b3aQ64NOZQW7fVjfoKwEZdOqPE72Pa45jrZzvUFxSpdiNk2tZ  
 XYukHjlxxEgBdC/J3cMMNRE1F4NCA3ApfV1Y7/hTeOnmDuDYwr9/obA8t016Yljj  
 q5rdkywPf4JF8mXUW5eCN1vAFHxeg9ZWemhBtQmGxXnw9M+z6hWwc6ahmwARAQAB  
 tCtEb2NrZXIgUmVsZWFzZSAoQ0UgZGViKSA8ZG9ja2VyQGRvY2tlci5jb20+iQI3  
 BBMBCgAhBQJYrefAAhsvBQsJCAcDBRUKCQgLBRYCAwEAAh4BAheAAAoJEI2BgDwO  
 v82IsskP/iQZo68flDQmNvn8X5XTd6RRaUH33kXYXquT6NkHJciS7E2gTJmqvMqd  
 tI4mNYHCSEYxI5qrcYV5YqX9P6+Ko+vozo4nseUQLPH/ATQ4qL0Zok+1jkag3Lgk  
 jonyUf9bwtWxFp05HC3GMHPhhcUSexCxQLQvnFWXD2sWLKivHp2fT8QbRGeZ+d3m  
 6fqcd5Fu7pxsqm0EUDK5NL+nPIgYhN+auTrhgzhK1CShfGccM/wfRlei9Utz6p9P  
 XRKIlWnXtT4qNGZNTN0tR+NLG/6Bqd8OYBaFAUcue/w1VW6JQ2VGYZHnZu9S8LMc  
 FYBa5Ig9PxwGQOgq6RDKDbV+PqTQT5EFMeR1mrjckk4DQJjbxeMZbiNMG5kGECA8  
 g383P3elhn03WGbEEa4MNc3Z4+7c236QI3xWJfNPdUbXRaAwhy/6rTSFbzwKB0Jm  
 ebwzQfwjQY6f55MiI/RqDCyuPj3r3jyVRkK86pQKBAJwFHyqj9KaKXMZjfVnowLh  
 9svIGfNbGHpucATqREvUHuQbNnqkCx8VVhtYkhDb9fEP2xBu5VvHbR+3nfVhMut5  
 G34Ct5RS7Jt6LIfFdtcn8CaSas/l1HbiGeRgc70X/9aYx/V/CEJv0lIe8gP6uDoW  
 FPIZ7d6vH+Vro6xuWEGiuMaiznap2KhZmpkgfupyFmplh0s6knymuQINBFit2ioB  
 EADneL9S9m4vhU3blaRjVUUyJ7b/qTjcSylvCH5XUE6R2k+ckEZjfAMZPLpO+/tF  
 M2JIJMD4SifKuS3xck9KtZGCufGmcwiLQRzeHF7vJUKrLD5RTkNi23ydvWZgPjtx  
 Q+DTT1Zcn7BrQFY6FgnRoUVIxwtdw1bMY/89rsFgS5wwuMESd3Q2RYgb7EOFOpnu  
 w6da7WakWf4IhnF5nsNYGDVaIHzpiqCl+uTbf1epCjrOlIzkZ3Z3Yk5CM/TiFzPk  
 z2lLz89cpD8U+NtCsfagWWfjd2U3jDapgH+7nQnCEWpROtzaKHG6lA3pXdix5zG8  
 eRc6/0IbUSWvfjKxLLPfNeCS2pCL3IeEI5nothEEYdQH6szpLog79xB9dVnJyKJb  
 VfxXnseoYqVrRz2VVbUI5Blwm6B40E3eGVfUQWiux54DspyVMMk41Mx7QJ3iynIa  
 1N4ZAqVMAEruyXTRTxc9XW0tYhDMA/1GYvz0EmFpm8LzTHA6sFVtPm/ZlNCX6P1X  
 zJwrv7DSQKD6GGlBQUX+OeEJ8tTkkf8QTJSPUdh8P8YxDFS5EOGAvhhpMBYD42kQ  
 pqXjEC+XcycTvGI7impgv9PDY1RCC1zkBjKPa120rNhv/hkVk/YhuGoajoHyy4h7  
 ZQopdcMtpN2dgmhEegny9JCSwxfQmQ0zK0g7m6SHiKMwjwARAQABiQQ+BBgBCAAJ  
 BQJYrdoqAhsCAikJEI2BgDwOv82IwV0gBBkBCAAGBQJYrdoqAAoJEH6gqcPyc/zY  
 1WAP/2wJ+R0gE6qsce3rjaIz58PJmc8goKrir5hnElWhPgbq7cYIsW5qiFyLhkdp  
 YcMmhD9mRiPpQn6Ya2w3e3B8zfIVKipbMBnke/ytZ9M7qHmDCcjoiSmwEXN3wKYI  
 mD9VHONsl/CG1rU9Isw1jtB5g1YxuBA7M/m36XN6x2u+NtNMDB9P56yc4gfsZVES  
 KA9v+yY2/l45L8d/WUkUi0YXomn6hyBGI7JrBLq0CX37GEYP6O9rrKipfz73XfO7  
 JIGzOKZlljb/D9RX/g7nRbCn+3EtH7xnk+TK/50euEKw8SMUg147sJTcpQmv6UzZ  
 cM4JgL0HbHVCojV4C/plELwMddALOFeYQzTif6sMRPf+3DSj8frbInjChC3yOLy0  
 6br92KFom17EIj2CAcoeq7UPhi2oouYBwPxh5ytdehJkoo+sN7RIWua6P2WSmon5  
 U888cSylXC0+ADFdgLX9K2zrDVYUG1vo8CX0vzxFBaHwN6Px26fhIT1/hYUHQR1z  
 VfNDcyQmXqkOnZvvoMfz/Q0s9BhFJ/zU6AgQbIZE/hm1spsfgvtsD1frZfygXJ9f  
 irP+MSAI80xHSf91qSRZOj4Pl3ZJNbq4yYxv0b1pkMqeGdjdCYhLU+LZ4wbQmpCk  
 SVe2prlLureigXtmZfkqevRz7FrIZiu9ky8wnCAPwC7/zmS18rgP/17bOtL4/iIz  
 QhxAAoAMWVrGyJivSkjhSGx1uCojsWfsTAm11P7jsruIL61ZzMUVE2aM3Pmj5G+W  
 9AcZ58Em+1WsVnAXdUR//bMmhyr8wL/G1YO1V3JEJTRdxsSxdYa4deGBBY/Adpsw  
 24jxhOJR+lsJpqIUeb999+R8euDhRHG9eFO7DRu6weatUJ6suupoDTRWtr/4yGqe  
 dKxV3qQhNLSnaAzqW/1nA3iUB4k7kCaKZxhdhDbClf9P37qaRW467BLCVO/coL3y  
 Vm50dwdrNtKpMBh3ZpbB1uJvgi9mXtyBOMJ3v8RZeDzFiG8HdCtg9RvIt/AIFoHR  
 H3S+U79NT6i0KPzLImDfs8T7RlpyuMc4Ufs8ggyg9v3Ae6cN3eQyxcK3w0cbBwsh  
 /nQNfsA6uu+9H7NhbehBMhYnpNZyrHzCmzyXkauwRAqoCbGCNykTRwsur9gS41TQ  
 M8ssD1jFheOJf3hODnkKU+HKjvMROl1DK7zdmLdNzA1cvtZH/nCC9KPj1z8QC47S  
 xx+dTZSx4ONAhwbS/LN3PoKtn8LPjY9NP9uDWI+TWYquS2U+KHDrBDlsgozDbs/O  
 jCxcpDzNmXpWQHEtHU7649OXHP7UeNST1mCUCH5qdank0V1iejF6/CfTFU4MfcrG  
 YT90qFF93M3v01BbxP+EIY2/9tiIPbrd  
 =0YYh  
 -----END PGP PUBLIC KEY BLOCK-----  
bootcmd:  
 - curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -  
package\_update: true  
package\_upgrade: true  
packages:  
 - screen  
 - rsync  
 - git  
 - curl  
 - python3-pip  
 - python3-dev  
 - python3-selinux  
 - python3-setuptools  
 - python3-venv  
 - libffi-dev  
 - gcc  
 - libssl-dev  
 - apt-transport-https  
 - ca-certificates  
 - curl  
 - gnupg-agent  
 - software-properties-common  
  
groups:  
 - docker  
system\_info:  
 default\_user:  
 groups: [docker]  
  
runcmd:  
 - curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -  
 - sudo apt-get update -y && apt-get install -y docker-ce docker-ce-cli  
 - docker run -d --name registry --restart=always -p 4000:5000 -v registry:/var/lib/registry registry:2  
 - pip3 install -U pip  
 - pip3 install -U wheel  
 - echo 'Azure Ampere VM Debian 11 Example' >> /etc/motd

### Running Terraform

Executing terraform is broken into three commands. The first you must initialize the terraform project with the modules and necessary plugins to support proper execution. The following command will do that:

terraform init

Below is output from a ‘terraform init’ execution within the project directory.

After ‘terraform init’ is executed it is necessary to run ‘plan’ to see the tasks, steps and objects. that will be created by interacting with the cloud APIs. Executing the following from a command line will do so:

terraform plan

The ouput from a ‘terraform plan’ execution in the project directy will look similar to the following:

Finally you will execute the ‘apply’ phase of the terraform exuction sequence. This will create all the objects, execute all the tasks and display any output that is defined. Executing the following command from the project directory will automatically execute without requiring any additional interaction:

terraform apply -auto-approve

The following is an example of output from a ‘apply’ run of terraform from within the project directory:

### Logging in

Next you’ll need to login with the dynamically generated sshkey that will be sitting in your project directory. To log in take the ip address from the output above and run the following ssh command:

ssh -i ./azure-id\_rsa debian@155.248.228.151

You should be automatically logged in after running the the command. The following is output from sshing into an instance and then running ‘sudo cat /var/log/messages’ to verify cloud-init execution and package installation:

### Destroying when done

You now should have a fully running and configured Debian instance. When finished you will need to execute the ‘destroy’ command to remove all created objects in a ‘leave no trace’ manner. Execute the following from a command to remove all created objects when finished:

terraform destroy -auto-approve

The following is example output of the ‘terraform destroy’ when used on this project.

Modifing the cloud-init file and then performing the same workflow will allow you to get interating quickly. At this point you should definately know how to quickly get automating using Debian with Ampere on the Cloud!