Passing Terraform Attributes to User Data During EC2 Creation

Shumnan Sun Follow
Jan 24, 2020 · 3 min read

Working-around to send atribute to EC2

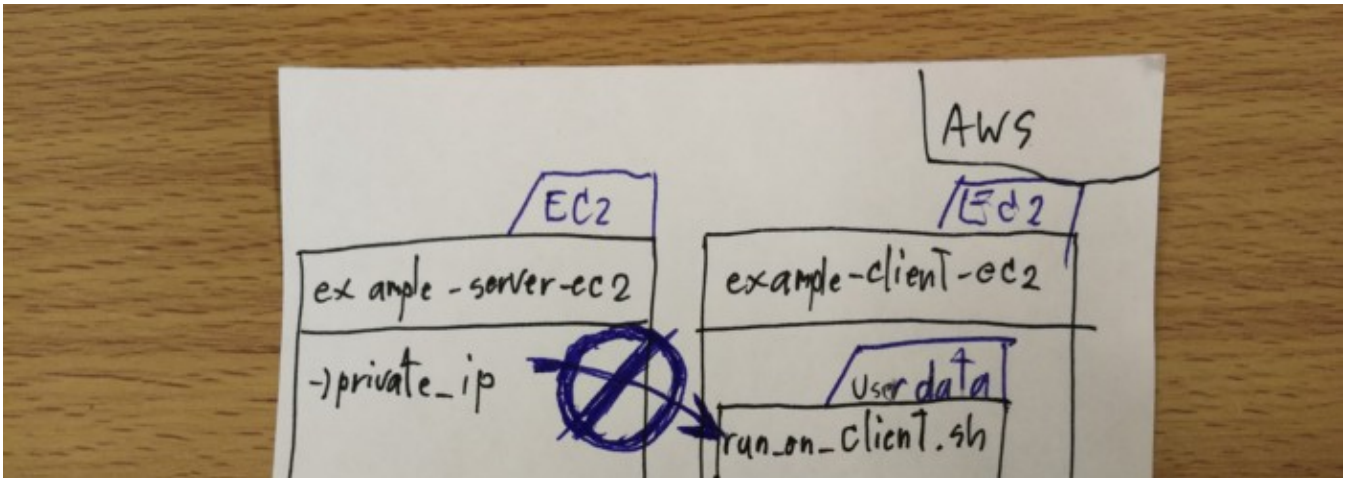
TL DR;

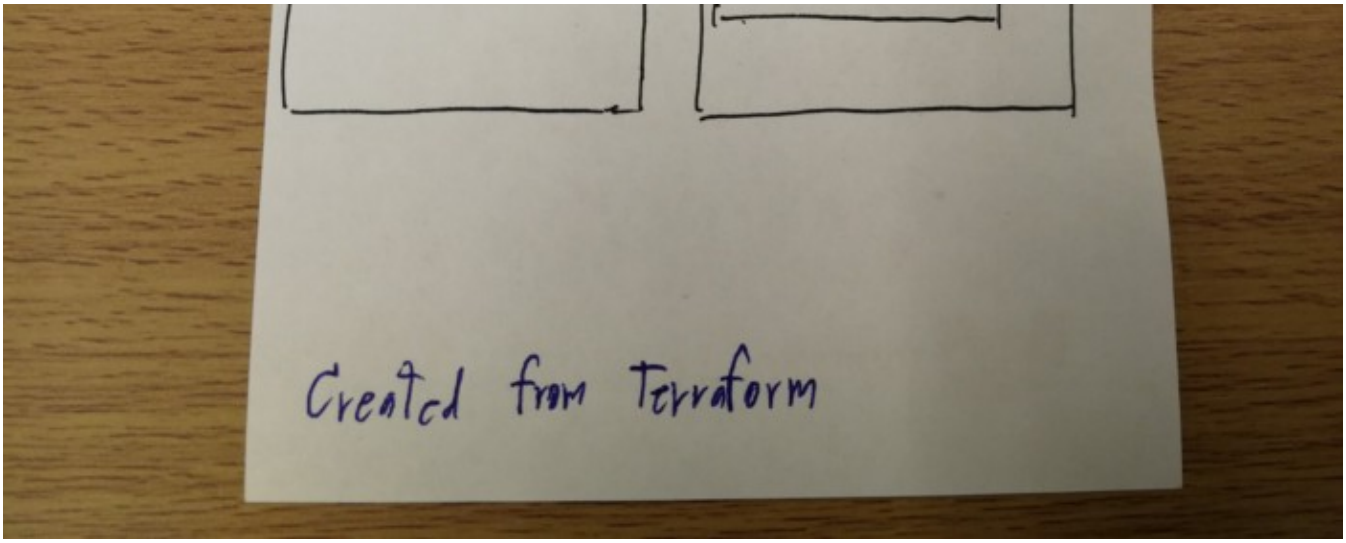
- The user data shell script is unable to get Terraform attributes
- Use ***template_cloudinit_config*** to pass Terraform attribute values to the shell script in ***template*** as **local config file**.

Problem:

- We want to create 2 instances of EC2, a server and a client.
- The client needs to know the server IP address during *user data* execution.
- Terraform doesn't provide access to its attributes for *user data*.

The user data always **gets empty values** when trying to access Terraform attributes.





Fail getting Terraform attribute

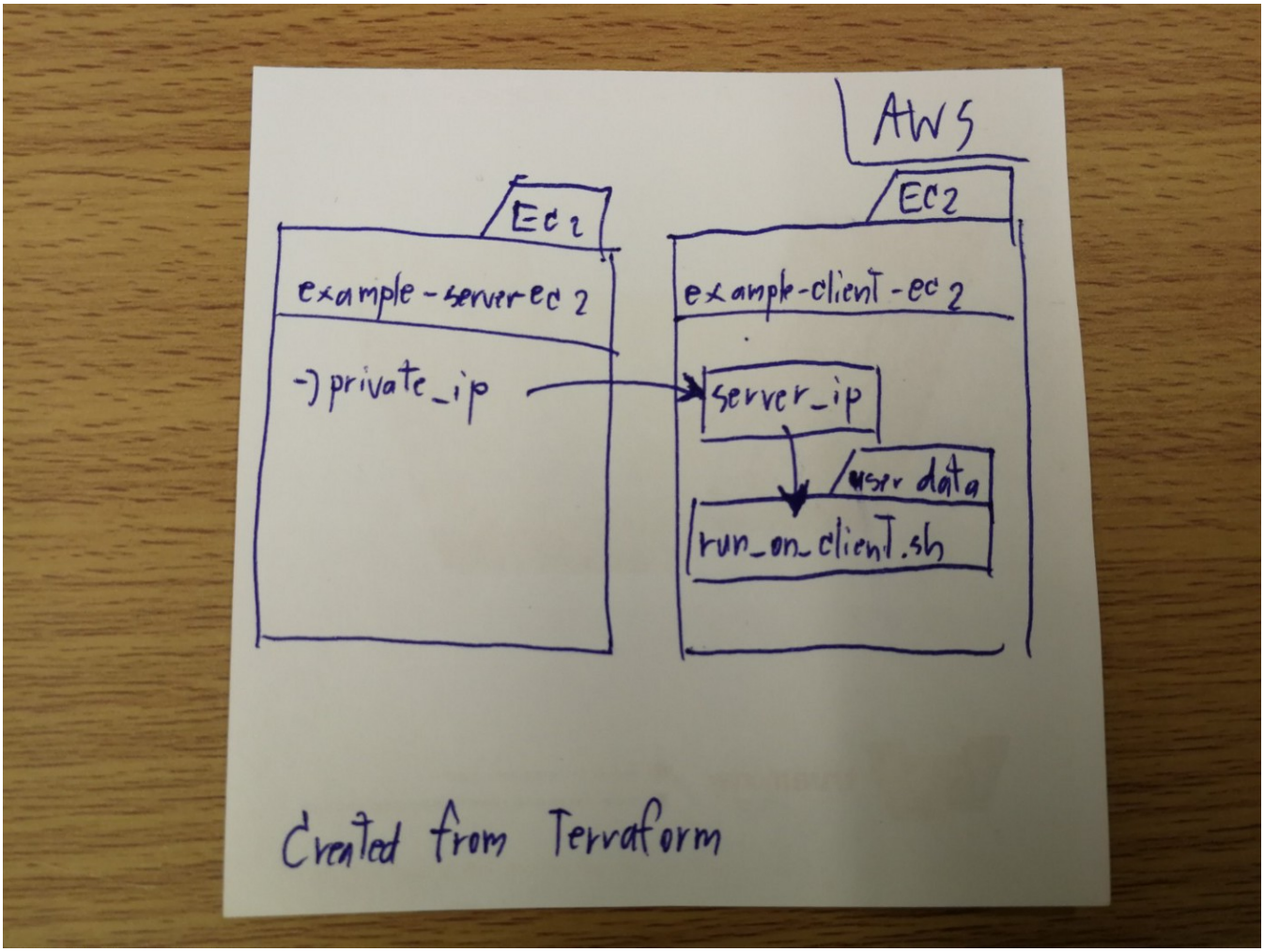
From above picture, Terraform would create instances as follows

- ***example-server-ec2***
- ***example-client-ec2*** — This instance **needs to obtain the server IP address**

After Terraform creates ***example-client-ec2*** EC2, it will execute the **User data (*run_on_client.sh)*** to get ***example-server-ec2*** IP address**.**

```
#!/bin/bash
echo "Server IP: $instance_target_host"
```

Solution:



Success getting Terraform attribute

- Create **ec2.tf**

  Create a Terraform file with 2 EC2 instances as a server and a client.

  The client's *user data* will be rendered from the following snippet.

```
resource "aws_instance" "example-server-ec2" {
 ami = ami-10000001
 instance_type     = "t3.small"
 availability_zone = "ap-southeast-1a"
}

resource "aws_instance" "example-client-ec2" {
 ami = ami-10000002
 instance_type     = "t3.small"
 availability_zone = "ap-southeast-1a"
 user_data         = data.template_cloudinit_config.config.rendered
}
```

- Create **init.tf** with **template_cloudinit_config**

  The **first part** will pass Terraform attributes to a local config file

The **second part** will pass the rendered **script from template** to **the same** local config file.

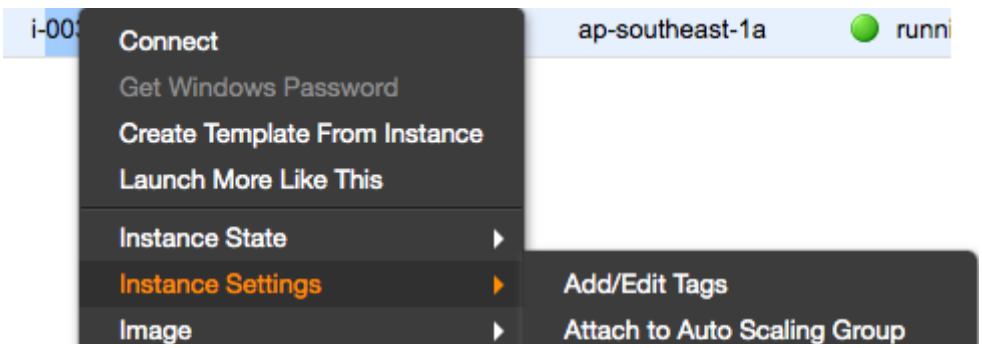
```
data "template_file" "client" {
  template = file("./user_data/run_on_client.sh")
}

data "template_cloudinit_config" "config" {
  gzip          = false
  base64_encode = false

  #first part of local config file
  part {
    content_type = "text/x-shellscript"
    content      = <<-EOF
    #!/bin/bash
    echo 'instance_target_host="${aws_instance.example-server-
ec2.private_ip}"' > /opt/server_ip
    EOF
  }

  #second part
  part {
    content_type = "text/x-shellscript"
    content      = data.template_file.client.rendered
  }
}
```

- Create user data shell script

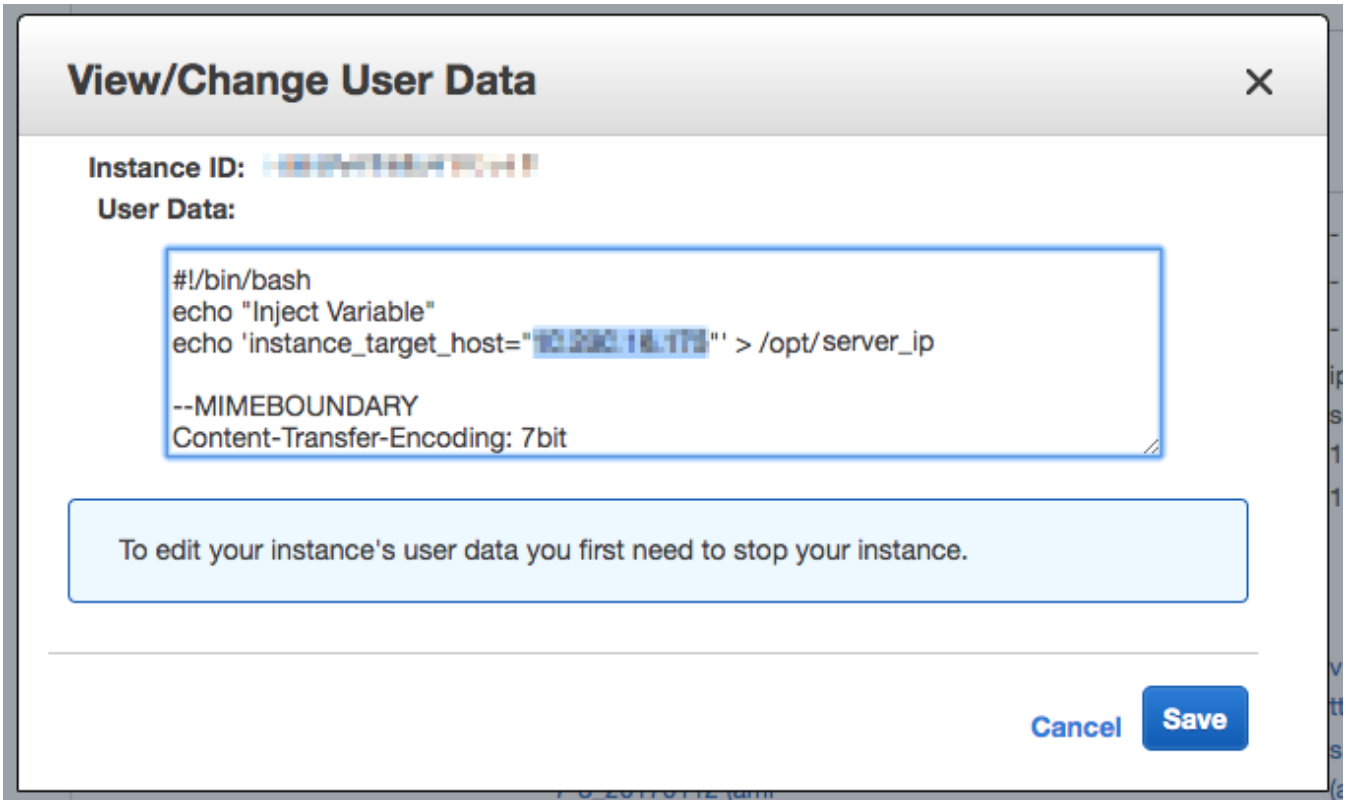
  The *run_on_client.sh* will be imported as **template_file** in the above snippet.

```
#!/bin/bash
source /opt/server_ip
echo "Inject Variable"
echo "Server IP: $instance_target_host"
```

- Then the value can be called from local config file as a user data of EC2.



Accessing User data on AWS console



Check User data via AWS console

Terraform is able to provide its attributes as variables for user data as a config file

And now it also becomes a better practice!

> **Another Use Cases:**
>
> 1. Provide **NFS server IP** to client
> 2. Provide **another sibling instance id**

> 3. Provide any **other Terraform attributes** after apply from the host instance to client instances.

References and Read more:

- Cloudinit PDF
- Cloudinit Readme
- Instancemetadata
- Merging Cloudinit

Thanks to Yuu HaMeaw and Jame James.

AWS Terraform Ec2