

Implantação em Amazon Web Services (AWS) utilizando recurso EC2 de planta virtual utilizando node-red e supervisor utilizando Scada-LTS.

1. Pré-requisitos

1.1. Tem conta no Amazon Web Services (AWS): <https://aws.amazon.com>

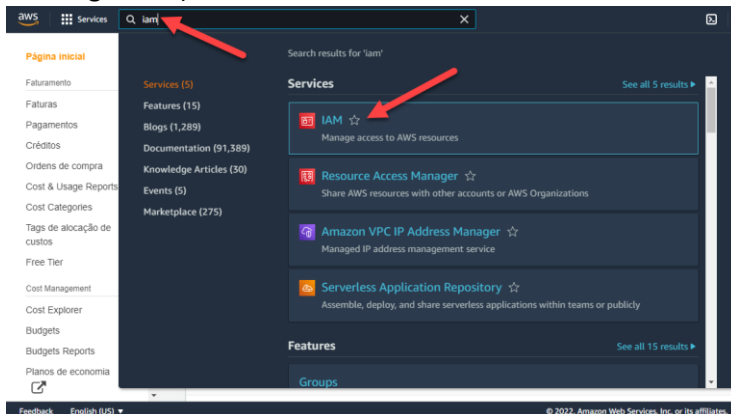
1.2. Conhecimento básico em informática, git e execução de comandos shell.

2. Clonar o repositório no GitHub: <https://github.com/rImariz/virtual-lab-deploy.git>

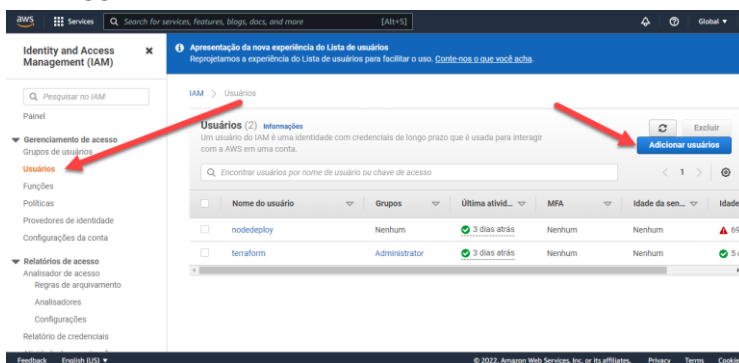
3. Após clonar o repositório a pasta de trabalho será "virtual-lab-deploy\aws", todos os arquivos salvos e comandos executados devem ser nessa pasta.

4. Criar conta de acesso para o terraform no AWS

4.1. Acesse o console da AWS e faça o login com sua conta e pesquise pelo produto IAM (Identity and Access Management).



4.2. Iremos criar um usuário para que o terraform possa interagir com a AWS, clique em USERS e em seguida em ADD USER.



4.4. Definir detalhes do usuário.

Definir detalhes do usuário

Você pode adicionar vários usuários de uma só vez com o mesmo tipo de acesso e permissões. [Saiba mais](#)

Nome de usuário* 1

[Adicionar outro usuário](#)

Selecione o tipo de acesso à AWS

Selecione a principal forma de acesso desses usuários à AWS. Se optar somente pelo acesso programático, isso NÃO impedirá que os usuários usem o console com uma função assumida. As chaves de acesso e as senhas geradas automaticamente são fornecidas na última etapa. [Saiba mais](#)

Selecionar tipo de credencial da AWS*

☒ **Chave de acesso: acesso programático** 2

Habilita uma ID da chave de acesso: chave de acesso secreta para a API da AWS, CLI, SDK, e outras ferramentas de desenvolvimento.

☐ **Senha: acesso ao Console de Gerenciamento da AWS**

Habilita uma senha que permite que os usuários façam login no Console de Gerenciamento da AWS.

3

* Obrigatório

[Cancelar](#) [Próximo: Permissões](#)

4.6. Adicione a política AmazonEC2FullAccess ao usuário, o que dará permissão total ao usuário apenas a recursos da EC2, e clique em Next.

4.8.

Adicionar tags (opcional)

4.10.

Revisar

Detalhes do usuário

Resumo de permissões

As políticas a seguir serão anexadas ao usuário mostrado acima.

Tags

Nenhuma tag foi adicionada.

4.12.

4.13. Clique em show e copie o Access key ID e Secret access key

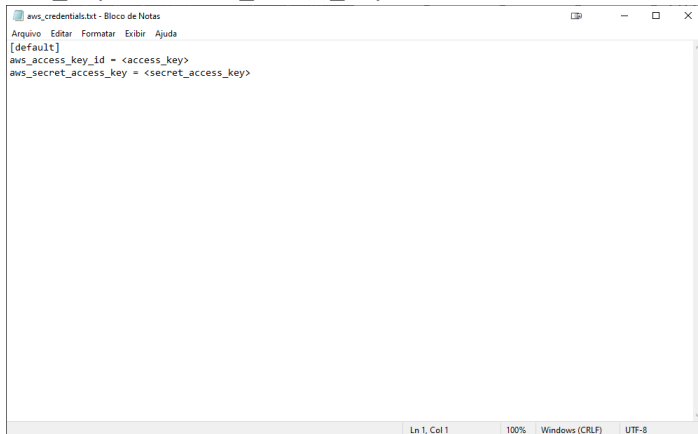
Adicionar usuário

4.14.

Fechar

4.15.O usuário criado e chave de acesso não devem ser compartilhados, uma vez que quem tiver acesso a estes dados terá controle sobre os recursos adicionados como política, por questões de segurança este usuário não existe mais em minha conta.

5. Editar arquivo **aws_credentials.txt** e adicionar a chave de acesso e a chave secreta substituindo os valores <access_key> e <secret_access_key>.



5.1.

6. Acessar o site e gerar par de chaves rsa que será utilizado para conexão ssh.

6.1.Acesso o website <https://www.wpoven.com/tools/create-ssh-key#>

6.2.Configure o type como rsa, length 2048, password deixe em branco e clique em create key.

Generate SSH Key Pair Online

Form fields for generating an SSH key pair online:

- Password:
- Type:
- Length:
- Create Key button

6.3. If password is needed

6.4.Fazer download do **Private Key** e salvar arquivo com nome **aws.key**.

6.5.Fazer download do **Public Key** e salvar arquivo com nome **aws.pub.key**.

7. Instalar o Terraform

7.1.Acesse o site <https://www.terraform.io/downloads>

7.2.Siga as instruções de acordo o sistema operacional que está utilizando

7.3.Ao fazer o download do executável de preferência coloque na pasta de trabalho para facilitar sua utilização

7.4.Pode ser feito o teste para verificar se está tudo ok executando no prompt de comandos: terraform --version

7.5.Vai exibir a versão instalada e a plataforma: Terraform v1.1.7 on windows_amd64

8. Caso seja necessário pode se editar o arquivo variables.tf e fazer os ajustes necessários

```
variable "region" {
  description = "Define what region the instance will be deployed"
  default     = "us-east-1"
}

variable "name" {
  description = "Name of the Application"
  default     = "virtuallab"
}

variable "env" {
  description = "Environment of the Application"
  default     = "prod"
}

variable "instance_type" {
  description = "AWS Instance type defines the hardware configuration of the machine"
  default     = "t2.micro"
}
```

8.1.

9. Executar o comando terraform init

```

PS C:\temp\virtual-lab-deploy\aws> .\terraform.exe init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v4.5.0...
- Installed hashicorp/aws v4.5.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
PS C:\temp\virtual-lab-deploy\aws>

```

9.1. PS C:\temp\virtual-lab-deploy\aws>

10. O próximo passo é criar infraestrutura e subir aplicação, vamos executar o comando: terraform apply

11. Vai ser exibido o plano de trabalho e estando tudo ok basta digitar “yes” e dar enter.

```

+ ipv6_cidr_blocks = []
+ prefix_list_ids  = []
+ protocol         = "tcp"
+ security_groups  = []
+ self             = false
+ to_port          = 22
},
]
+ name              = "allow_ssh"
+ name_prefix       = (known after apply)
+ owner_id          = (known after apply)
+ revoke_rules_on_delete = false
+ tags_all          = (known after apply)
+ vpc_id            = (known after apply)
}

```

Plan: 3 to add, 0 to change, 0 to destroy.

Changes to Outputs:

```

+ nodered          = (known after apply)
+ public_dns       = (known after apply)
+ public_ip        = (known after apply)
+ scada-lts        = (known after apply)
+ ssh_connection   = (known after apply)

```

Do you want to perform these actions?

Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

11.1.

12. O processo leva certa de 6 minutos para ser implementado e ao final será exibido os endpoints para acesso ao supervisor, node-red e caso necessário o comando para acesso via ssh.

```

aws_instance.server: Still creating... [5m0s elapsed]
aws_instance.server: Still creating... [5m10s elapsed]
aws_instance.server (remote-exec): + sudo mv all-databases.sql ./scadalts-data
aws_instance.server: Still creating... [5m20s elapsed]
aws_instance.server (remote-exec): + docker exec -i mysql sh -c 'exec mysql -uroot -proot -f < /var/lib/mysql/all-databases.sql'
aws_instance.server (remote-exec): mysql: [Warning] Using a password on the command line interface can be insecure.
aws_instance.server: Still creating... [5m30s elapsed]
aws_instance.server (remote-exec): ERROR 1146 (42502) at line 2542: Table 'scadalts.plcalarms' doesn't exist
aws_instance.server (remote-exec): ERROR 1146 (42502) at line 2560: Table 'scadalts.plcalarms' doesn't exist
aws_instance.server (remote-exec): + docker stop scadalts
aws_instance.server: Still creating... [5m40s elapsed]
aws_instance.server (remote-exec): scadalts
aws_instance.server (remote-exec): + docker start scadalts
aws_instance.server: Still creating... [5m50s elapsed]
aws_instance.server (remote-exec): scadalts
aws_instance.server (remote-exec): + docker cp ./uploads/5.png scadalts:/usr/local/tomcat/webapps/Scada-LTS/uploads/5.png
aws_instance.server: Creation complete after 5m53s [id=i-09ff591aaa4536904]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Outputs:
nodered = "ec2-34-238-126-127.compute-1.amazonaws.com:1880"
public_dns = "ec2-34-238-126-127.compute-1.amazonaws.com"
public_ip = "34.238.126.127"
scada-lts = "ec2-34-238-126-127.compute-1.amazonaws.com:8080/Scada-LTS"
ssh_connection = "ssh -i aws_key ec2-user@ec2-34-238-126-127.compute-1.amazonaws.com"
PS C:\temp\virtual-lab-deploy\aws>

```

12.1.

13. O usuário e senha para acesso ao supervisor é admin/admin.

14. É muito importante desalocar os recursos após finalizar sua utilização para não ter custos extras, para fazer isso basta executar o comando terraform destroy e confirmar com yes, ao final os recursos serão desalocado.

```

}

Plan: 0 to add, 0 to change, 3 to destroy.

Changes to Outputs:
- nodered = "ec2-34-238-126-127.compute-1.amazonaws.com:1880" -> null
- public_dns = "ec2-34-238-126-127.compute-1.amazonaws.com" -> null
- public_ip = "34.238.126.127" -> null
- scada-lts = "ec2-34-238-126-127.compute-1.amazonaws.com:8080/Scada-LTS" -> null
- ssh_connection = "ssh -i aws_key ec2-user@ec2-34-238-126-127.compute-1.amazonaws.com" -> null

Do you really want to destroy all resources?
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: yes

aws_key_pair.deployer: Destroying... [id=aws_virtualab_key]
aws_instance.server: Destroying... [id=i-09ff591aaa4536904]
aws_key_pair.deployer: Destruction complete after 0s
aws_instance.server: Still destroying... [id=i-09ff591aaa4536904, 10s elapsed]
aws_instance.server: Still destroying... [id=i-09ff591aaa4536904, 20s elapsed]
aws_instance.server: Still destroying... [id=i-09ff591aaa4536904, 30s elapsed]
aws_instance.server: Still destroying... [id=i-09ff591aaa4536904, 40s elapsed]
aws_instance.server: Destruction complete after 41s
aws_security_group.allow_ssh: Destroying... [id=sg-016bbc0deea88dc39]
aws_security_group.allow_ssh: Destruction complete after 1s

Destroy complete! Resources: 3 destroyed.
PS C:\temp\virtual-lab-deploy\aws>

```

14.1.