Terraform code for Amazon EC2 Systems Manager

Here’s a basic **Terraform configuration** for setting up **Amazon EC2 Systems Manager (SSM)** to manage your EC2 instances. This includes:

1. **IAM Role & Policy** – To allow EC2 instances to use SSM.
2. **Instance Profile** – So EC2 instances can assume the IAM role.
3. **SSM Agent Installation** – Ensuring EC2 instances are ready for SSM.
4. **Parameter Store Example** – Storing configuration parameters.

provider "aws" {

region = "us-east-1"

}

# IAM Role for SSM

resource "aws\_iam\_role" "ssm\_role" {

name = "EC2SSMRole"

assume\_role\_policy = jsonencode({

Version = "2012-10-17"

Statement = [{

Action = "sts:AssumeRole"

Effect = "Allow"

Principal = {

Service = "ec2.amazonaws.com"

}

}]

})

}

# Attach AmazonSSMManagedInstanceCore Policy to Role

resource "aws\_iam\_role\_policy\_attachment" "ssm\_role\_attach" {

role = aws\_iam\_role.ssm\_role.name

policy\_arn = "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore"

}

# IAM Instance Profile for EC2

resource "aws\_iam\_instance\_profile" "ssm\_profile" {

name = "EC2SSMInstanceProfile"

role = aws\_iam\_role.ssm\_role.name

}

# Launch EC2 Instance with SSM Role

resource "aws\_instance" "ssm\_instance" {

ami = "ami-0c55b159cbfafe1f0" # Replace with a valid AMI ID

instance\_type = "t2.micro"

iam\_instance\_profile = aws\_iam\_instance\_profile.ssm\_profile.name

subnet\_id = "subnet-12345678" # Replace with your subnet ID

security\_groups = [aws\_security\_group.ssm\_sg.name]

tags = {

Name = "SSM-Managed-Instance"

}

}

# Security Group for EC2

resource "aws\_security\_group" "ssm\_sg" {

name = "ssm-sg"

description = "Allow SSM Agent communication"

ingress {

from\_port = 22

to\_port = 22

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"] # Restrict this in production

}

}

# AWS SSM Parameter Store Example

resource "aws\_ssm\_parameter" "example\_parameter" {

name = "/app/config/database\_url"

type = "String"

value = "jdbc:mysql://db.example.com:3306/app\_db"

}

**How It Works**

1. **IAM Role & Policy** – Allows EC2 instances to communicate with AWS Systems Manager.
2. **Instance Profile** – Attaches the role to an EC2 instance.
3. **EC2 Instance** – Launched with SSM capabilities.
4. **SSM Parameter Store** – Stores a sample database URL.

Here’s a basic \*\*Terraform configuration\*\* for setting up \*\*Amazon EC2 Systems Manager (SSM)\*\* to manage your EC2 instances. This includes:

1. \*\*IAM Role & Policy\*\* – To allow EC2 instances to use SSM.

2. \*\*Instance Profile\*\* – So EC2 instances can assume the IAM role.

3. \*\*SSM Agent Installation\*\* – Ensuring EC2 instances are ready for SSM.

4. \*\*Parameter Store Example\*\* – Storing configuration parameters.

---

### \*\*Terraform Configuration for AWS SSM\*\*

```hcl

provider "aws" {

region = "us-east-1"

}

# IAM Role for SSM

resource "aws\_iam\_role" "ssm\_role" {

name = "EC2SSMRole"

assume\_role\_policy = jsonencode({

Version = "2012-10-17"

Statement = [{

Action = "sts:AssumeRole"

Effect = "Allow"

Principal = {

Service = "ec2.amazonaws.com"

}

}]

})

}

# Attach AmazonSSMManagedInstanceCore Policy to Role

resource "aws\_iam\_role\_policy\_attachment" "ssm\_role\_attach" {

role = aws\_iam\_role.ssm\_role.name

policy\_arn = "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore"

}

# IAM Instance Profile for EC2

resource "aws\_iam\_instance\_profile" "ssm\_profile" {

name = "EC2SSMInstanceProfile"

role = aws\_iam\_role.ssm\_role.name

}

# Launch EC2 Instance with SSM Role

resource "aws\_instance" "ssm\_instance" {

ami = "ami-0c55b159cbfafe1f0" # Replace with a valid AMI ID

instance\_type = "t2.micro"

iam\_instance\_profile = aws\_iam\_instance\_profile.ssm\_profile.name

subnet\_id = "subnet-12345678" # Replace with your subnet ID

security\_groups = [aws\_security\_group.ssm\_sg.name]

tags = {

Name = "SSM-Managed-Instance"

}

}

# Security Group for EC2

resource "aws\_security\_group" "ssm\_sg" {

name = "ssm-sg"

description = "Allow SSM Agent communication"

ingress {

from\_port = 22

to\_port = 22

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"] # Restrict this in production

}

}

# AWS SSM Parameter Store Example

resource "aws\_ssm\_parameter" "example\_parameter" {

name = "/app/config/database\_url"

type = "String"

value = "jdbc:mysql://db.example.com:3306/app\_db"

}

```

---

### \*\*How It Works\*\*

1. \*\*IAM Role & Policy\*\* – Allows EC2 instances to communicate with AWS Systems Manager.

2. \*\*Instance Profile\*\* – Attaches the role to an EC2 instance.

3. \*\*EC2 Instance\*\* – Launched with SSM capabilities.

4. \*\*SSM Parameter Store\*\* – Stores a sample database URL.

---

### \*\*Deploy Steps\*\*

1. Save the file as `main.tf`.

2. Initialize Terraform:

```bash

terraform init

```

3. Preview changes:

```bash

terraform plan

```

4. Apply changes:

```bash

terraform apply -auto-approve

```

Your EC2 instance should now be \*\*SSM-managed\*\*! 🎉 You can connect to it using \*\*AWS Systems Manager Session Manager\*\* instead of SSH.

Let me know if you need any modifications! 🚀