Département de génie informatique

TP3 - Automation and Vulnerability checks

Auteur	*Email*	*Matricule*
Ilias Bettayeb	ilias.bettayeb@polymtl.ca	2092408
Benoit Dambrine	benoit.dambrine@polymtl.ca	2075984

Présenté à :
M. Armstrong Foundjem

10 novembre 2023

Table des matières

1 Questions

1.1 Vulnerability checks

1. *Go inside the folder of each configuration file in Lab2 and execute a vulnerability check. Then, classify and analyze each type of vulnerability. In particular, build a taxonomy of vulnerability and try to resolve those that are found in the config file.*

   For each file, checkov finds more misconfigurations / vulnerabilities than trivy. The vulnerabilities found in trivy were all found by checkov. Checkov found more vulnerabilities, which makes it a better solution. The best would be to use both to be secure.

   Checkov and Trivy vulnerability checks are attached in annexe.

2. How would you extract vulnerabilities automatically to avoid human error ?

   Automate the security check and stop the instance launch if a vulnerability is detected.

1.2 Automation

3. Replicate this CloudFormation stack to terraform.

```
# Define the AWS provider configuration (credentials and region)
provider "aws" {
    region = "us-east-1"
}

# Create a VPC
resource "aws_vpc" "my_vpc" {
    cidr_block = "10.0.0.0/16"
    enable_dns_support = true
    enable_dns_hostnames = true
}

# Create a public subnet
resource "aws_subnet" "public_subnet" {
    vpc_id = aws_vpc.my_vpc.id
    cidr_block = "10.0.0.0/24"
    availability_zone = "us-east-1a"
```

```
    map_public_ip_on_launch = true
}

# Create an Internet Gateway
resource "aws_internet_gateway" "internet_gateway" {}

# Attach the Internet Gateway to the VPC
resource "aws_vpc_attachment" "attach_gateway" {
    vpc_id = aws_vpc.my_vpc.id
    internet_gateway_id = aws_internet_gateway.internet_gateway.id
}

# Create a security group for SSH access
resource "aws_security_group" "security_group" {
    name = "Enable SSH access"
    description = "Enable SSH access"
    vpc_id = aws_vpc.my_vpc.id
    ingress {
      from_port = 22
      to_port = 22
      protocol = "tcp"
      cidr_blocks = ["0.0.0.0/0"]
    }
}

# Create EC2 instances
resource "aws_instance" "ec2_instance" {
    count = 4
    ami = "ami-0fc5d935ebf8bc3bc" # Specify the desired AMI ID here
    instance_type = "t2.micro"
    subnet_id = aws_subnet.public_subnet.id
    security_groups = [aws_security_group.security_group.id]
}

# Create an S3 bucket
resource "aws_s3_bucket" "my_s3_bucket" {}

# Create a DynamoDB table
resource "aws_dynamodb_table" "my_dynamo_db_table" {
    name = "MyTable"
    billing_mode = "PROVISIONED"
    read_capacity = 5
```

```
    write_capacity = 5
    attribute {
      name = "ID"
      type = "N"
    }
    key_schema {
      name = "ID"
      attribute_type = "N"
    }
}

# Define output values
output "ec2_instance_ids" {
    description = "IDs of the EC2 Instances"
    value = [for instance in aws_instance.ec2_instance : instance.id]
}
output "s3_bucket_name" {
    description = "S3 Bucket Name"
    value = aws_s3_bucket.my_s3_bucket.id
}
output "dynamo_db_table_name" {
    description = "DynamoDB Table Name"
    value = aws_dynamodb_table.my_dynamo_db_table.name
}
```

4. What are the key differences between security groups and subnets ?

   Security groups are responsible for the control of the incoming and outgoing traffic within one or several instances of a vpc. They are at the instance level. Subnets represent the IP address range associated to a VPC. They are responsible for the network layout at VPC level.

5. During the CloudFormation process, observe the order of service creation and explain the order how services are created.

   1) Pick a template that specifies the resources that you want in your stack. The sample template creates a basic WordPress blog that uses a single Amazon EC2 instance with a local MySQL database for storage. The template also creates an Amazon EC2 security group to control firewall settings for the Amazon EC2 instance. A template is a JSON or YAML text file that contains the configuration information about the AWS resources

2) Make sure you have prepared any required items for the stack by making sure you have prepared any required items for the stack

3) Create the Stack based on the WordPress-1.0.0 file

4) Monitor the progress of stack creation. In the stack details pane, choose the Events tab to view each major step in the creation of the stack

5) Use your stack resources after a status of CREATE_COMPLETE. The sample WordPress stack creates a WordPress website. You can continue with the WordPress setup by running the WordPress installation script.

6) Clean up. To make sure you aren't charged for any unwanted services, you can clean up by deleting the stack and its resources.

Source :
https ://docs.aws.amazon.com/AWSCloudFormation/latest/UserGuide/GettingStarted
.Walkthrough.html#GettingStarted.Walkthrough.p

ANNEXE

checkov -d . lab2\01-s3-bucket\aws-backend\main.tf

logiciel	Vulnerabilité	Code	Solution

checkov	Ensure Dynamodb point in time recovery (backup) is enabled	44 | resource "aws_dynamodb_table" "terraform_locks" {
		45 | name = "terraform-state-locking"
		46 | billing_mode = "PAY_PER_REQUEST"
		47 | hash_key = "LockID"
		48 | attribute {
		49 | name = "LockID"
		50 | type = "S"
		51 | }
		52 | }
			add point-in-time recovery = enabled

checkov	Ensure DynamoDB Tables are encrypted using a KMS Customer Managed CMK	44 | resource "aws_dynamodb_table" "terraform_locks" {
		45 | name = "terraform-state-locking"
		46 | billing_mode = "PAY_PER_REQUEST"
		47 | hash_key = "LockID"
		48 | attribute {
		49 | name = "LockID"
		50 | type = "S"
		51 | }
		52 | }
			add in ressource server_side_encryption {
			enabled = true
			}

checkov	Ensure S3 buckets should have event notifications enabled	23 | resource "aws_s3_bucket" "terraform_state" {
		24 | bucket = "s3-bucket-lab-02"
		25 | force_destroy = true
		26 | }
			add resource "aws_s3_bucket_notification" "bucket_notification" {
			bucket = aws_s3_bucket.bucket.id
			
			topic {
			topic_arn = aws_sns_topic.topic.arn
			events = ["s3:ObjectCreated:*"]
			filter_suffix = ".log"
			}
			}

checkov	Ensure that S3 bucket has a Public Access block	23 | resource "aws_s3_bucket" "terraform_state" {
		24 | bucket = "s3-bucket-lab-02"
		25 | force_destroy = true
		26 | }
			add
			resource "aws_s3_bucket_public_access_block" "example" {
			bucket = aws_s3_bucket.example.id
			
			block_public_acls = true
			block_public_policy = true
			ignore_public_acls = true
			restrict_public_buckets = true
			}

checkov	Ensure that an S3 bucket has a lifecycle configuration	23 | resource "aws_s3_bucket" "terraform_state" {
		24 | bucket = "s3-bucket-lab-02"
		25 | force_destroy = true
		26 | }
			add
			resource "aws_s3_bucket_lifecycle_configuration" "example" {
			bucket = aws_s3_bucket.bucket.id
			
			rule {
			id = "rule-1"
			
			filter {}
			
			# ... other transition/expiration actions ...
			
			status = "Enabled"
			}
			}

checkov	Ensure the S3 bucket has access logging enabled	23 | resource "aws_s3_bucket" "terraform_state" {
		24 | bucket = "s3-bucket-lab-02"
		25 | force_destroy = true
		26 | }
			add
			resource "aws_s3_bucket_logging" "example" {
			bucket = aws_s3_bucket.example.id
			
			target_bucket = aws_s3_bucket.log_bucket.id
			target_prefix = "log/"
			}

checkov	Ensure that S3 bucket has cross-region replication enabled	23 | resource "aws_s3_bucket" "terraform_state" {
		24 | bucket = "s3-bucket-lab-02"
		25 | force_destroy = true
		26 | }
			add
			resource "aws_s3_bucket_replication_configuration" "east_to_west" {
			depends_on = [aws_s3_bucket_versioning.east]
			role = aws_iam_role.east_replication.arn
			bucket = aws_s3_bucket.east.id
			
			rule {
			status = "Enabled"
			
			destination {
			bucket = aws_s3_bucket.west.arn
			storage_class = "STANDARD"
			}
			}
			}

checkov	Ensure that S3 buckets are encrypted with KMS by default	23 | resource "aws_s3_bucket" "terraform_state" {
		24 | bucket = "s3-bucket-lab-02"
		25 | force_destroy = true
		26 | }
			add
			resource "aws_s3_bucket_server_side_encryption_configuration" "good_sse_1" {
			bucket = aws_s3_bucket.bucket_name.bucket
			
			rule {
			
			apply_server_side_encryption_by_default {
			kms_master_key_id = aws_kms_key.mykey.arn
			sse_algorithm = "aws:kms"
			}
			}
			}

trivy	Table encryption is not enabled.	44 ┌ resource "aws_dynamodb_table" "terraform_locks" {
		45 │ name = "terraform-state-locking"
		46 │ billing_mode = "PAY_PER_REQUEST"
		47 │ hash_key = "LockID"
		48 │ attribute {
		49 │ name = "LockID"
		50 │ type = "S"
		51 │ }
		52 └ }
			add in resource server_side_encryption {
			enabled = true
			}

trivy	Point-in-time recovery is not enabled.	44 ┌ resource "aws_dynamodb_table" "terraform_locks" {
		45 │ name = "terraform-state-locking"
		46 │ billing_mode = "PAY_PER_REQUEST"
		47 │ hash_key = "LockID"
		48 │ attribute {
		49 │ name = "LockID"
		50 │ type = "S"
		51 │ }
		52 └ }
			add in resource point_in_time_recovery {
			enabled = true
			}

trivy	Table encryption does not use a customer-managed KMS key	44 ┌ resource "aws_dynamodb_table" "terraform_locks" {
		45 │ name = "terraform-state-locking"
		46 │ billing_mode = "PAY_PER_REQUEST"
		47 │ hash_key = "LockID"
		48 │ attribute {
		49 │ name = "LockID"
		50 │ type = "S"
		51 │ }
		52 └ }
			add in resource
			server_side_encryption {
			enabled = true
			kms_key_arn = aws_kms_key.dynamo_db_kms.key_id
			}
			
			add resource
			resource "aws_kms_key" "dynamo_db_kms" {
			enable_key_rotation = true
			}

trivy	No public access block so not blocking public acls	23 ┌ resource "aws_s3_bucket" "terraform_state" {
		24 │ bucket = "s3-bucket-lab-02" # REPLACE WITH YOUR BUCKET NAME
		25 │ force_destroy = true
		26 └ }
			add resource
			resource "aws_s3_bucket_public_access_block" "terraform_state" {
			bucket = aws_s3_bucket.terraform_state.id
			block_public_acls = true
			}

trivy	No public access block so not blocking public policies	23 ┌ resource "aws_s3_bucket" "terraform_state" {
		24 │ bucket = "s3-bucket-lab-02" # REPLACE WITH YOUR BUCKET NAME
		25 │ force_destroy = true
		26 └ }
			add resource
			resource "aws_s3_bucket_public_access_block" "terraform_state" {
			bucket = aws_s3_bucket.terraform_state.id
			block_public_policy = true
			}

trivy	Bucket has logging disabled	23 ┌ resource "aws_s3_bucket" "terraform_state" {
		24 │ bucket = "s3-bucket-lab-02" # REPLACE WITH YOUR BUCKET NAME
		25 │ force_destroy = true
		26 └ }
			add in resource
			logging {
			target_bucket = "target-bucket"
			}

trivy	No public access block so not ignoring public acls	23 ┌ resource "aws_s3_bucket" "terraform_state" {
		24 │ bucket = "s3-bucket-lab-02" # REPLACE WITH YOUR BUCKET NAME
		25 │ force_destroy = true
		26 └ }
			add resource
			resource "aws_s3_bucket_public_access_block" "terraform_state" {
			bucket = aws_s3_bucket.terraform_state.id
			
			ignore_public_acls = true
			}

trivy	No public access block so not restricting public buckets	23 ┌ resource "aws_s3_bucket" "terraform_state" {
		24 │ bucket = "s3-bucket-lab-02" # REPLACE WITH YOUR BUCKET NAME
		25 │ force_destroy = true
		26 └ }
			add resource
			resource "aws_s3_bucket_public_access_block" "terraform_state" {
			bucket = aws_s3_bucket.terraform_state.id
			
			restrict_public_buckets = true
			}

trivy	Bucket does not have a corresponding public access block	23 ┌ resource "aws_s3_bucket" "terraform_state" {
		24 │ bucket = "s3-bucket-lab-02" # REPLACE WITH YOUR BUCKET NAME
		25 │ force_destroy = true
		26 └ }
			add in resource
			acl = "private-read"
			
			add resource
			
			resource "aws_s3_bucket_public_access_block" "terraform_state" {
			bucket = aws_s3_bucket.terraform_state.id
			block_public_acls = true
			block_public_policy = true
			}

trivy	Bucket does not encrypt data with a customer managed key	35 resource "aws_s3_bucket_server_side_encryption_configuration" "terraform_state_crypto_conf" {
		36 bucket = aws_s3_bucket.terraform_state.bucket
		37 rule {
		38 ┌ apply_server_side_encryption_by_default {
		39 │ sse_algorithm = "AES256"
		40 └ }
		41 }
		42 }
			add resource
			resource "aws_kms_key" "good_example" {
			enable_key_rotation = true
			}
			
			change the apply_server_side_encryption_by_default to
			kms_master_key_id = aws_kms_key.good_example.arn
			sse_algorithm = "aws:kms"

checkov -d . lab2\vpc\main.tf

checkov	Vulnérabilité	Code	Solution

checkov	Ensure that detailed monitoring is enabled for EC2 instances	29 | resource "aws_instance" "vm" { 30 | count = 8 31 | ami = var.ami_id 32 | instance_type = "t2.micro" 33 | subnet_id = element(aws_subnet.subnet[*].id, count.index % length(aws_subnet.subnet[*].id)) 34 | }	add in resource monitoring = true
checkov	Ensure that EC2 is EBS optimized	29 | resource "aws_instance" "vm" { 30 | count = 8 31 | ami = var.ami_id 32 | instance_type = "t2.micro" 33 | subnet_id = element(aws_subnet.subnet[*].id, count.index % length(aws_subnet.subnet[*].id)) 34 | }	add in resource ebs_optimized = true

checkov	Ensure Instance Metadata Service Version 1 is not enabled	29 | resource "aws_instance" "vm" {
		30 | count = 8
		31 | ami = var.ami_id
		32 | instance_type = "t2.micro"
		33 | subnet_id = element(aws_subnet.subnet[*].id, count.index % length(aws_subnet.subnet[*].id))
		34 | }
			add in resource
			metadata_options {
			...
			http_endpoint = "enabled"
			http_tokens = "required"
			}

checkov	Ensure all data stored in the Launch configuration or instance Elastic Blocks Store is securely encrypted	29 | resource "aws_instance" "vm" {
		30 | count = 8
		31 | ami = var.ami_id
		32 | instance_type = "t2.micro"
		33 | subnet_id = element(aws_subnet.subnet[*].id, count.index % length(aws_subnet.subnet[*].id))
		34 | }
			add in resource
			root_block_device {
			encrypted = true
			}

checkov	Ensure an IAM role is attached to EC2 instance	29 | resource "aws_instance" "vm" {
		30 | count = 8
		31 | ami = var.ami_id
		32 | instance_type = "t2.micro"
		33 | subnet_id = element(aws_subnet.subnet[*].id, count.index % length(aws_subnet.subnet[*].id))
		34 | }
			add in resource
			iam_instance_profile = "test"
			
			add resource
			resource "aws_iam_instance_profile" "test" {
			name = "test"
			role = aws_iam_role.role.name
			}
			
			add resource
			resource "aws_iam_role" "role" {
			name = "test_role"
			path = "/"
			assume_role_policy = data.aws_iam_policy_document.assume_role.json
			}

checkov	Ensure KMS key Policy is defined	38 | resource "aws_kms_key" "encryption_key" { 39 | description = "Encryption key for sensitive data" 40 | enable_key_rotation = true 41 | deletion_window_in_days = 7 42 | }	add in resource policy = {valid json policy document}

checkov	Ensure VPC flow logging is enabled in all VPCs	10 | resource "aws_vpc" "main" {
		11 | cidr_block = "10.0.0.0/16"
		12 | enable_dns_support = true
		13 | enable_dns_hostnames = true
		14 |
		15 | tags = {
		16 | Name = "MainVPC"
		17 | }
		18 | }
			add resource
			resource "aws_flow_log" "example" {
			iam_role_arn = "arn"
			log_destination = "log"
			traffic_type = "ALL"
			vpc_id = aws_vpc.main.id
			}

checkov	Ensure the default security group of every VPC restricts all traffic	10 | resource "aws_vpc" "main" {
		11 | cidr_block = "10.0.0.0/16"
		12 | enable_dns_support = true
		13 | enable_dns_hostnames = true
		14 |
		15 | tags = {
		16 | Name = "MainVPC"
		17 | }
		18 | }
			add resource
			resource "aws_default_security_group" "default" {
			vpc_id = aws_vpc.main.id
			
			- ingress {
			- protocol = "-1"
			- self = true
			- from_port = 0
			- to_port = 0
			- }
			
			- egress {
			- from_port = 0
			- to_port = 0
			- protocol = "-1"
			- cidr_blocks = ["0.0.0.0/0"]
			- }
			}

trivy	Instance does not require IMDS access to require a token	29 ┌ resource "aws_instance" "vm" {
		30 | count = 8
		31 | ami = var.ami_id
		32 | instance_type = "t2.micro"
		33 | subnet_id = element(aws_subnet.subnet[*].id, count.index % length(aws_subnet.subnet[*].id))
		34 └ }
			add in resource
			metadata_options {
			http_tokens = "required"
			}

trivy	Root block device is not encrypted	29 ┌ resource "aws_instance" "vm" {
		30 │ count = 8
		31 │ ami = var.ami_id
		32 │ instance_type = "t2.micro"
		33 │ subnet_id = element(aws_subnet.subnet[*].id, count.index % length(aws_subnet.subnet[*].id))
		34 └ }
			add in resource
			root_block_device {
			encrypted = true
			}
			
			ebs_block_device {
			device_name = "/dev/sdg"
			volume_size = 5
			volume_type = "gp2"
			delete_on_termination = false
			encrypted = true
			}

trivy	VPC Flow Logs is not enabled for VPC	10 ┌ resource "aws_vpc" "main" {
		11 │ cidr_block = "10.0.0.0/16"
		12 │ enable_dns_support = true
		13 │ enable_dns_hostnames = true
		14 │
		15 │ tags = {
		16 │ Name = "MainVPC"
		17 │ }
		18 └ }
			add resource
			resource "aws_flow_log" "example" {
			iam_role_arn = "arn"
			log_destination = "log"
			traffic_type = "ALL"
			vpc_id = aws_vpc.main.id
			}