



CloudX Associate: AWS for Testers

VPC

TASK 4 – VPC

1. DEPLOY THE IAM APPLICATION

Deploy the **cloudxinfo** CDK stack: [deployment instructions](#).

2. DEPLOYMENT VALIDATION

Create a manual/automated deployment validation test suite that covers the following requirements:

CXQA-VPC-01: VPC configuration:

- The application should be deployed in non-default VPC with has 2 subnets: public and private.
- VPC CIDR Block: 10.0.0.0/16
- VPC tags: cloudx: qa

CXQA-VPC-02: Subnets and routing configuration:

- The public instance should be accessible from the internet by Internet Gateway.
- The public instance should have access to the private instance.
- The private instance should have access to the internet via NAT Gateway.
- The private instance should not be accessible from the public internet.

2.1 Testing Tools:

- AWS Console
- AWS CLI
- AWS SDK (for automated tests).
- Application Swagger OpenAPI documentation
- Postman / CURL
- SSH client

2.2 Testing Tools:

- Application Swagger OpenAPI documentation
- Postman / CURL
- SSH client

Execute test cases and verify that requirements are met.

3. REGRESSION TESTING

- Deploy **version 3** of **cloudxinfo** application [deployment instructions](#).
- Execute deployment and functional validation test suites against new application version deployment.
- Create bug/root cause report for the found regression issues.

4. ENVIRONMENT CLEAN-UP

Delete the **cloudxiam** application stack and clean up the environment: [clean-up instructions](#).

5. SUBMIT RESULTS

Upload the home task artifacts (screenshots, test cases/links to automated tests code in the git repository) to Learn Portal and change the task status to „Needs Review“.

IMPORTANT THINGS TO KEEP IN MIND

- Once you create AWS Account, setup Multi-factor Authentication!
- Do NOT share your account!
- Do NOT commit your account Credentials to the Git repository!
- Terminate/Remove (destroy) all created resources/services once you finish the module (or the learning for the day)!
- Please Do NOT forget to delete NAT Gateway if you used it!
- Do NOT keep the instances running if you don't use it!
- Carefully keep track of billing and working instances so you don't exceed limits!

AWS VPC THEORY

What kinds of IP addresses does AWS VPC offer?

Private IP Addresses:

These are non-routable IP addresses used within your VPC. They are not accessible from the public internet. Each subnet within your VPC has a unique CIDR block, which defines the range of private IP addresses available for resources launched within that subnet. Common CIDR block sizes for subnets include /24, /20, and /16.

Private IP addresses are beneficial for:

- Isolating resources within your VPC from the public internet for enhanced security.
- Providing a dynamic allocation of IP addresses to your instances, managed by AWS.

Public IP Addresses (Optional):

These are routable IP addresses that allow your instances to communicate directly with the public internet. You can optionally associate a public IP address with an EC2 instance at launch time or later. However, it's generally recommended to avoid public IP addresses whenever possible and rely on NAT Gateways for controlled outbound internet access (explained below).

Public IP addresses are useful for:

- Instances that need to be directly accessible from the internet, such as web servers. Instances that require inbound connections from specific sources outside your VPC (not recommended for security reasons).

Additional Considerations:

- Elastic IP Addresses: These are public IP addresses that you can allocate and associate with your resources. They are more flexible than instance-attached public IP addresses as you can detach and attach them to different instances.
- NAT Gateways (Optional): If your instances in a private subnet need access to the internet without a public IP address, a NAT Gateway provides a one-way outbound connection. The NAT Gateway translates private IP addresses to its public IP address for outbound traffic, while inbound traffic from the internet cannot reach instances in the private subnet.

What happens to different IP address types when an EC2 instance is rebooted, stopped, started?

IP Address Type	Reboot	Stop	Start
Private IP Address	Remains the same	Remains the same	Remains the same
Public IP Address (Instance-Attached)	Remains the same	Released	New public IP address assigned (randomly)
Elastic IP Address	Remains the same (if associated with the instance)	Maintains association with the instance	Public IP address remains the same (associated Elastic IP)

How many elastic IPs is it possible to create per account/region?

By default, each AWS account is limited to creating a maximum of 5 Elastic IP addresses per Region. However, you can request an increase in this quota if you need more than 5 Elastic IPs in a specific region.

What is the difference between NAT gateways and NAT instances?

NAT gateways and NAT instances are both methods to provide outbound internet access for private instances in an AWS VPC, but they differ in terms of management, scalability, and cost.

NAT Gateways:

- **Managed Service:** NAT gateways are a fully managed service offered by AWS. You simply create a NAT gateway in a public subnet within your VPC, and AWS handles the underlying infrastructure and maintenance.

Legal Notice: This document contains privileged and/or confidential information and may not be disclosed, distributed or reproduced without the prior written permission of EPAM®.

- **Highly Scalable:** NAT gateways are highly scalable and can automatically adjust their capacity based on the amount of traffic flowing through them. This ensures they can handle sudden spikes in outbound traffic without performance degradation.
- **Cost-Effective for High Traffic:** While NAT gateways incur hourly charges and data processing fees, they become more cost-effective as your outbound traffic volume increases.
- **Security:** NAT gateways leverage Amazon VPC firewall rules to control inbound and outbound traffic, potentially providing an additional security layer compared to NAT instances.
- **Limitations:** NAT gateways cannot be used for inbound traffic and are limited to IPv4 traffic.

NAT Instances:

- **User-Managed:** NAT instances are essentially EC2 instances running a NAT software package. You're responsible for choosing the instance type, configuring security groups, and ensuring the instance is healthy and available.
- **Manual Scaling:** You need to manually scale NAT instances by launching additional instances if outbound traffic demands increase. This can be time-consuming and requires proactive management.
- **Cost-Effective for Low Traffic:** NAT instances might be cost-effective for scenarios with low outbound traffic volume, as you only pay for the running EC2 instance and data processing fees.
- **Security:** Security relies on the security groups configured for the NAT instance, which might require more careful setup compared to NAT gateways.
- **Flexibility:** NAT instances can potentially handle both inbound and outbound traffic, although this is not a common practice and requires additional configuration. They can also support IPv6 traffic with specific configuration.

For most scenarios, NAT gateways are the recommended option. They offer a more managed, scalable, and secure solution for providing outbound internet access to your private instances.

Consider using NAT instances if:

- You have a very low volume of outbound traffic, and cost is the primary concern.
- You require both inbound and outbound traffic capabilities.
- You need IPv6 support (with additional configuration)

What is the difference between security groups and network ACLs?

Security groups and network ACLs (Network Access Control Lists) are both security features within AWS VPC that control traffic flow to and from your resources. However, they have distinct functionalities and apply at different levels:

Security Groups:

- **Function:** Act as firewalls at the instance level.
- **Control:** Inbound and outbound traffic for individual EC2 instances and other resources like network interfaces.
- **Scope:** Specific to an instance or network interface. You can associate multiple security groups with a single instance.
- **Rules:** Allow or deny rules based on source IP address or CIDR block, protocol (TCP, UDP, ICMP, etc.), and port range.
- **Stateful:** Security groups maintain information about past connections, allowing return traffic for established connections.

Network ACLs:

- **Function:** Act as firewalls at the subnet level.
- **Control:** Inbound and outbound traffic for all resources within a subnet.
- **Scope:** Apply to an entire subnet. You can only associate one network ACL with a subnet.
- **Rules:** Allow or deny rules similar to security groups, but typically used for broader access control.
- **Stateless:** Network ACLs do not maintain connection state, so both inbound and outbound traffic must be

Legal Notice: This document contains privileged and/or confidential information and may not be disclosed, distributed or reproduced without the prior written permission of EPAM®.

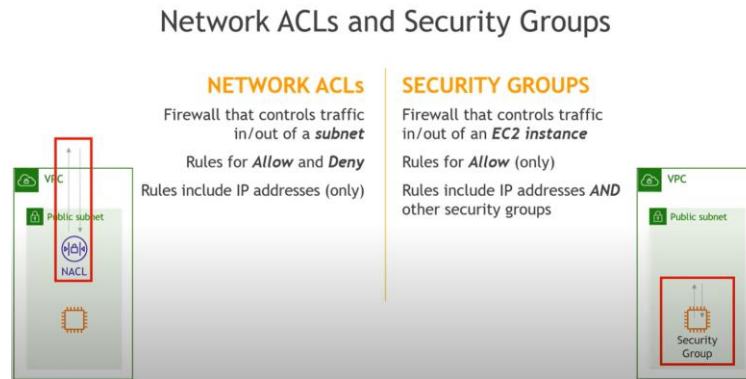
explicitly allowed for communication.

To understand the difference:

Imagine your VPC as a gated community.

Security groups are like security guards at the doors of individual houses within the community. They control who (source IP) can enter a house (instance) and through which door (port).

Network ACLs are like the main gatekeeper at the entrance of the community (subnet). They decide whether to allow visitors (traffic) into the community in the first place.



Suppose you've assigned a CIDR block to a VPC. Will all the IPs in that block be available for the resources you create in the VPC?

No, not all the IPs in the CIDR block you assign to a VPC will be available for the resources you create within the VPC. Here's why:

Reserved IP Addresses:

A small number of IP addresses within your VPC's CIDR block are reserved by AWS and cannot be used for your resources. These typically include the network address and broadcast address of the CIDR block.

Subnet Allocation:

When you create subnets within your VPC, you specify a CIDR block for each subnet. This CIDR block is a smaller range of IP addresses carved out from the VPC's overall CIDR block. Only the IP addresses within the subnet's CIDR block are available for resources launched within that subnet.

Example:

Let's say you create a VPC with a CIDR block of 10.0.0.0/16. This provides a total of 65,536 (2^{16}) IP addresses.

However, the network address (10.0.0.0) and broadcast address (10.0.255.255) are reserved.

Now, you create a subnet within the VPC with a CIDR block of 10.0.1.0/24. This subnet has 256 (2^8) usable IP addresses (excluding reserved addresses within the /24 block).

What does "local" target mean in terms of an AWS routing table?

In an AWS route table, the term "local" used as a target refers to traffic destined for resources within the same VPC.

Understanding Route Tables:

A route table in an AWS VPC defines how traffic flows within your VPC and to the internet (if applicable).

Each route table entry specifies a destination CIDR block (range of IP addresses) and the target where the traffic should be directed.

Local Target:

When the target for a route is set to "local," it indicates that the traffic for the specified destination CIDR block should be routed to the local instance or resource within the VPC itself.

AWS doesn't need to send the traffic out to an internet gateway or another VPC for processing. This keeps traffic flow

Legal Notice: This document contains privileged and/or confidential information and may not be disclosed, distributed or reproduced without the prior written permission of EPAM®.

Error! Unknown document property name.

optimized within the VPC.

Example:

Imagine you have a VPC with two subnets: Subnet A (10.0.1.0/24) and Subnet B (10.0.2.0/24).

You create a route table associated with Subnet B.

One route entry in the table has a destination CIDR block of 10.0.1.0/24 and a target of "local."

If an instance in Subnet B tries to access a resource with an IP address in the 10.0.1.0/24 range (i.e., a resource in Subnet A), the route table in Subnet B will direct the traffic locally within the VPC.

The traffic will be routed to the appropriate subnet (Subnet A) without needing to leave the VPC or traverse the internet gateway.

Key Points:

"Local" target ensures efficient traffic routing for resources within the same VPC.

It's typically used for communication between instances or resources in different subnets within the same VPC.

There's usually a separate route in the route table that defines how to reach the internet (using an internet gateway as the target) for outbound traffic.

What is bastion in terms of networking?

In the context of networking, a bastion host (also sometimes called a jump server or jump box) acts as a secure gateway or access point to a private network, typically located behind a firewall or within a Demilitarized Zone (DMZ).

Purpose:

The primary purpose of a bastion host is to provide a controlled and secure entry point for authorized users to access resources within a private network. This enhances security by reducing the attack surface of critical internal systems.

Security Measures:

- Bastion hosts are typically hardened with robust security measures, including:
- Limited functionalities: They run minimal services, reducing the potential attack surface for vulnerabilities.
- Strong authentication: Multi-factor authentication (MFA) is often used to ensure only authorized users can access the bastion host.
- Restricted access: Access to the bastion host itself might be limited to specific IP addresses or user groups.
- Logging and monitoring: All access attempts and activities on the bastion host are closely monitored and logged for security analysis.

Access Flow:

Authorized users first connect to the bastion host from the public internet (or another trusted network).

Once authenticated, they can leverage the bastion host to access resources within the private network using tools like SSH or remote desktop software.

The bastion host itself might not have direct access to these internal resources, but it can act as a stepping stone for authorized users to establish secure connections.

Benefits:

- Reduced Attack Surface: By centralizing access through a bastion host, the attack surface of internal systems is reduced as they are not directly exposed to the public internet.
- Enhanced Security: Bastion hosts can enforce strong authentication and access control policies, adding another layer of security for your private network.
- Auditing and Compliance: Centralized access through a bastion host simplifies logging and monitoring user activity, aiding in security audits and compliance efforts.

What is elastic network interface?

An Elastic Network Interface (ENI) in AWS acts as a virtual network card for your EC2 instances. It allows your instances to connect to a Virtual Private Cloud (VPC) and communicate with other resources within the VPC or the internet (if configured).

Functionality:

Legal Notice: This document contains privileged and/or confidential information and may not be disclosed, distributed or reproduced without the prior written permission of EPAM®.

Each ENI is associated with a specific VPC and subnet. It provides a unique MAC address (Media Access Control address) for network identification and an IP address (private or public) for communication.

You can attach an ENI to an EC2 instance at launch time or dynamically attach it to a running instance. An instance can have multiple ENIs attached, enabling it to have multiple IP addresses and network connections.

Benefits:

- **Flexibility:** ENIs offer flexibility in managing network configurations. You can easily attach or detach ENIs from instances, allowing you to modify network connectivity on the fly.
- **Scalability:** You can scale your instances independently of their network configuration. By attaching additional ENIs with different IP addresses, you can accommodate changing network requirements for your instances.
- **Security:** ENIs can be associated with security groups, which act as firewalls controlling inbound and outbound traffic for the attached instance. This allows you to define granular security policies for each network interface.

Key Points:

- ENIs are a fundamental building block for networking within an AWS VPC.
- They provide virtual network cards for your EC2 instances, enabling them to connect to the VPC and communicate with other resources.
- ENIs offer flexibility, scalability, and enhanced security for network management in your AWS environment..

Do you pay for VPC when using EC2 instances?

No, you are not directly charged for creating and using the VPC itself when launching EC2 instances in AWS. VPC acts as a virtual network within your account that isolates your resources, and it's a free service to create and use.

What are the tools for monitoring VPC?

There are several tools you can leverage to monitor your VPC in AWS, categorized into two main approaches:

Built-in AWS Monitoring Tools:

- **Amazon CloudWatch:** This is the central monitoring service for AWS resources. CloudWatch provides various metrics, logs, and alarms for monitoring your VPC health and performance. Here are some key features:
- **VPC Flow Logs:** Capture detailed information about traffic flowing to and from network interfaces within your VPCs. You can analyze these logs to identify traffic patterns, troubleshoot connectivity issues, and ensure security compliance.
- **CloudWatch Metrics:** Provide insights into various VPC components, including:
- **NAT Gateway metrics:** monitor the health and utilization of your NAT Gateways.
- **VPC endpoint metrics:** track traffic and latency for VPC endpoints.
- **Route table metrics:** monitor the health and changes within your route tables.
- **CloudWatch Alarms:** You can set up alarms based on specific CloudWatch metrics to receive notifications when there are anomalies or potential issues in your VPC, such as high CPU utilization on a NAT Gateway or unexpected changes in route tables.

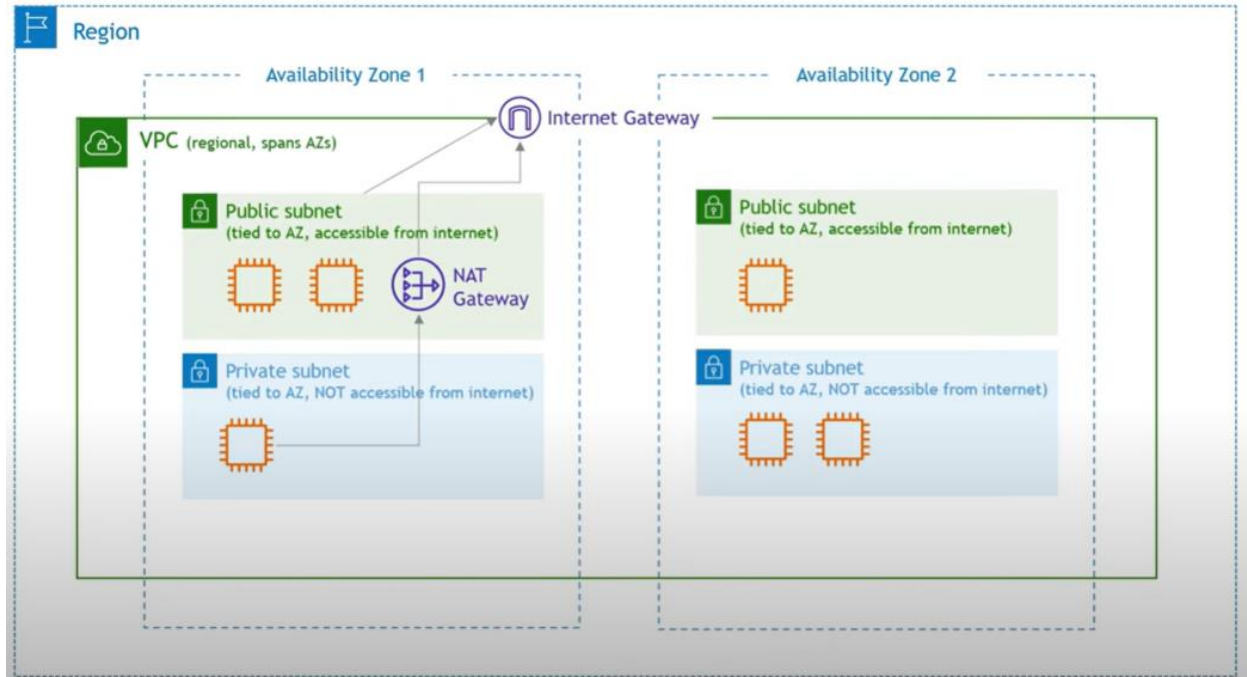
Third-Party Monitoring Tools:

In addition to built-in AWS tools, several third-party monitoring solutions integrate with AWS and offer comprehensive VPC monitoring capabilities. These tools can provide additional functionalities and visualizations compared to native AWS tools. Here are some popular options:

- **Datadog:** Provides real-time monitoring and alerting for VPC resources, including network traffic analysis, flow visualization, and security posture assessments.
- **SolarWinds Network Performance Monitor (NPM):** Offers detailed monitoring of network performance within your VPC, including bandwidth utilization, latency, and application performance.

How do VPC subnets map onto AZs?

Legal Notice: This document contains privileged and/or confidential information and may not be disclosed, distributed or reproduced without the prior written permission of EPAM®.



Legal Notice:

This document contains privileged and/or confidential information and may not be disclosed, distributed or reproduced without the prior written permission of EPAM®.

TASK IMPLEMENTATION

1. DEPLOYMENT VALIDATION

1.1 TAF

Link to repo with task implemented: <https://github.com/Chubaka2612/CloudX/pull/2/files>

The TAF solution was extended:

- CloudX.Auto.Tests – The following test suite was added:
 - VPC – covers CXQA-VPC-01, CXQA-VPC-02 scenarios

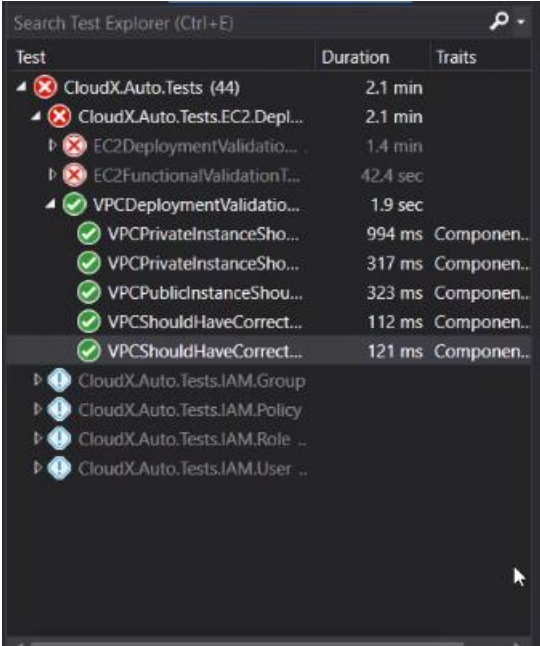
In total includes 5 new tests.

Used tools:

1. NUnit – test runner
2. AWS SDK for .NET – API for AWS resource interaction
3. Log3Net – logging library

1.2 Deployment test execution

For cloudxiam v=1 the are no issues found. Results are below:



Test	Duration	Traits
CloudX.Auto.Tests (44)	2.1 min	
CloudX.Auto.Tests.EC2.Depl...	2.1 min	
EC2DeploymentValidatio...	1.4 min	
EC2FunctionalValidationT...	42.4 sec	
VPCDeploymentValidatio...	1.9 sec	
VPCPrivateInstanceSho...	994 ms	Componen...
VPCPrivateInstanceSho...	317 ms	Componen...
VPCPublicInstanceShou...	323 ms	Componen...
VPCShouldHaveCorrect...	112 ms	Componen...
VPCShouldHaveCorrect...	121 ms	Componen...
CloudX.Auto.Tests.IAM.Group		
CloudX.Auto.Tests.IAM.Policy		
CloudX.Auto.Tests.IAM.Role ..		
CloudX.Auto.Tests.IAM.User ..		

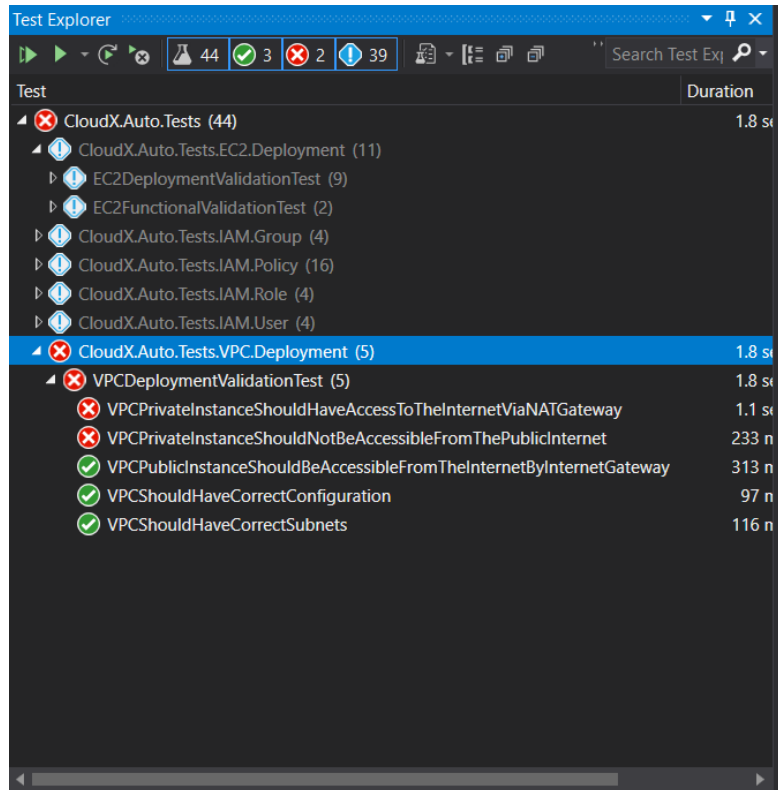
Pic 1 – test results execution for cloudxinfo = 1

Link to test execution: [demo](#)

1.3 Regression Deployment test execution

The cloudxinfo v=3 is deployed.

Tests were re-executed and 2 issues were found



Pic 2 – tests are executed on cloudinfo v=3 and 2 tests failed

Link to regression test execution: [demo](#)

1.4 Bug report

Summary: The private instance should not be accessible from the public internet

Type: Bug
Priority: Critical
Affects Version: 3
Component: VPC
Labels: Regression

Pre-conditions:

1. The private EC2 instance is established and running
2. cloudxinfo is deployed on the EC2 instance

Steps to reproduce:

Case#1:

1. Get instance's public IP (use any convenient way: CLI, Instance Dashboard)

Legal Notice: This document contains privileged and/or confidential information and may not be disclosed, distributed or reproduced without the prior written permission of EPAM®.

2. Observe result.

Actual results: The private instance has a public IP assigned.

Expected results: The private instance should not have a public IP assigned.

Case#2:

1. Get instance's subnet assigned (use any convenient way: CLI, Instance Dashboard)
2. Observe subnet MapPublicIpOnLaunch Value.

Actual results: The assigned subnet has MapPublicIpOnLaunch = true for private instance.

Expected results: The assigned subnet should have MapPublicIpOnLaunch = false for private instance.