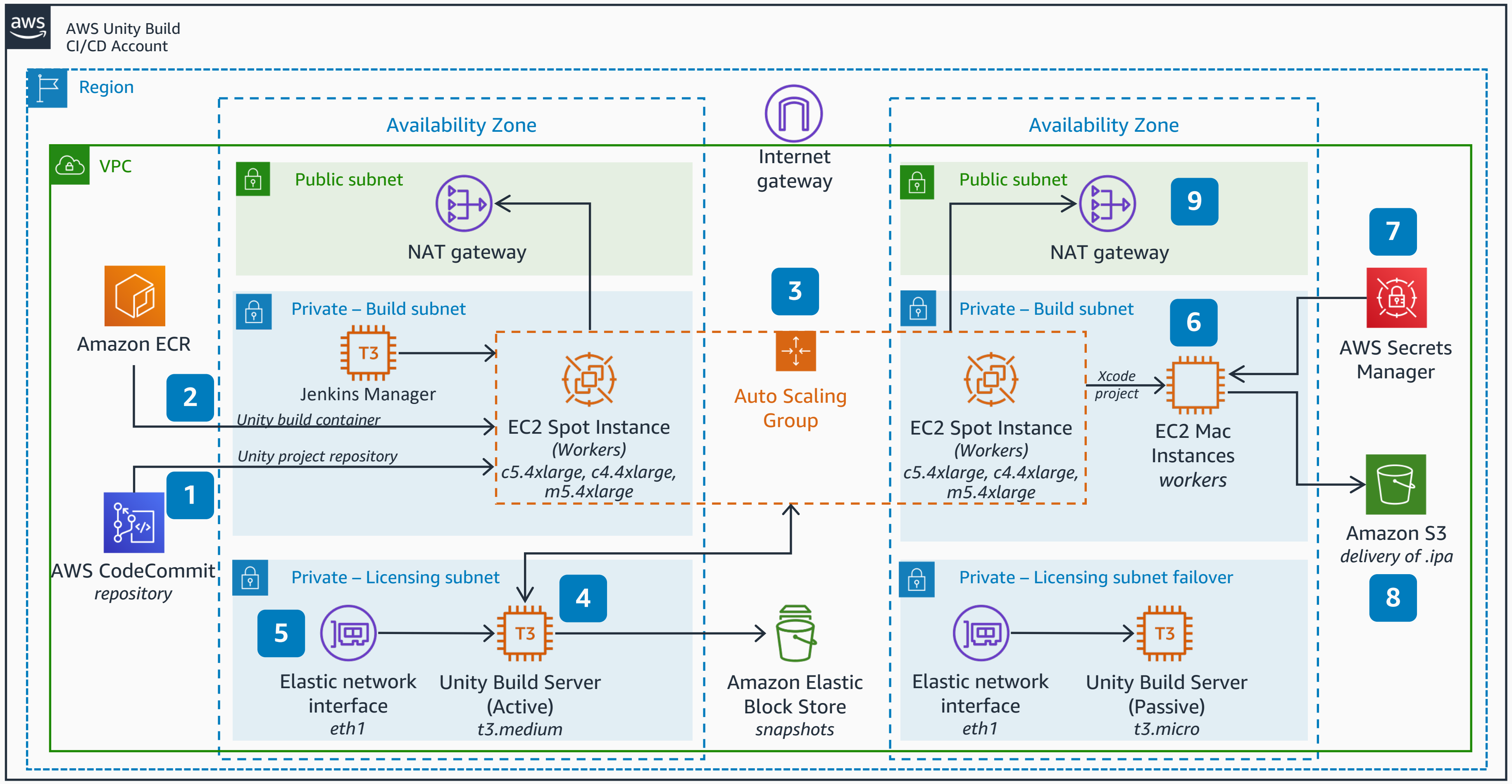
# Unity Build Pipeline

## Build Unity games for iOS in the AWS Cloud

This architecture builds Unity-based games for iOS in the AWS Cloud with Jenkins. The build process runs in two stages to optimize cost and performance. First, an Xcode project is generated on an auto-scaled fleet of Amazon EC2 Spot Instances. Second, the build is finalized and signed on a set of Amazon EC2 Mac Instances. Unity Build Server is used to manage floating Unity licenses.



AWS Unity Build CI/CD Account
Region
Availability Zone
Internet gateway
Availability Zone
VPC
Public subnet
NAT gateway
Public subnet
NAT gateway
9
Amazon ECR
Private – Build subnet
3
Private – Build subnet
6
7
T3
Jenkins Manager
2
Unity build container
Unity project repository
EC2 Spot Instance (Workers) c5.4xlarge, c4.4xlarge, m5.4xlarge
Auto Scaling Group
EC2 Spot Instance (Workers) c5.4xlarge, c4.4xlarge, m5.4xlarge
Xcode project
EC2 Mac Instances workers
AWS Secrets Manager
1
AWS CodeCommit repository
Amazon S3 delivery of .ipa
8
Private – Licensing subnet
4
Private – Licensing subnet failover
5
Elastic network interface eth1
T3
Unity Build Server (Active) t3.medium
Amazon Elastic Block Store snapshots
Elastic network interface eth1
T3
Unity Build Server (Passive) t3.micro

1. Source code is stored in an **AWS CodeCommit** repository and is pulled by Jenkins on a build start.
2. A Unity container image is pulled from **Amazon Elastic Container Registry (Amazon ECR)** and is deployed by the Jenkins Docker agent on a worker node.
3. The first build stage (generating Xcode project from Unity source code), is run on **Amazon Elastic Compute Cloud (Amazon EC2)** Spot Instances. These are placed into an Auto Scaling group for scalability and redundancy.
4. A Unity Build Server vends floating licenses to workers in the Auto Scaling group.
5. Unity license bounds to the **EC2** instance's Ethernet interface MAC address. That's why an elastic network interface is used. It can be reattached to a new instance preserving the MAC address if the instance gets recreated.
6. The resulting XCode project is transferred to a Jenkins worker on one of **Amazon EC2 Mac** Instances to finalize and sign the build and export and ipa file.
7. Certificates, private keys, and provisioning profiles are stored in **AWS Secrets Manager** and dynamically pulled onto the Mac during a build.
8. An .ipa archive file is exported as a Jenkins artifact and stored in an **Amazon Simple Storage Service (Amazon S3)** bucket.
9. Jenkins workers and manager can download plugins and packages by a NAT gateway.

*Reviewed for technical accuracy November 24, 2021*

**AWS Reference Architecture**