Game Production in the Cloud – Continuous integration/continuous deployment (CI/CD)
Offload game builds to the AWS Cloud

An engine-agnostic, high-level architecture for offloading game builds from remote or on-premises game development environments to the AWS Cloud. This architecture aids developers with migrating or building net new build farms on AWS.

**AWS Direct Connect** provides a low latency, private, dedicated connection to AWS for in-office developers. Remote developers use **AWS Client VPN**.

**AWS Transit Gateway** simplifies network management for connectivity between VPCs and from on-premises networks. Perforce manages source and version control (CI) backed by Amazon EBS storage for quickly-accessed, persistent data. Perforce Helix Core (P4D) is available on AWS Marketplace.

Commits start a build (CD) in Jenkins when developers push changes to Perforce tied to a branch. Perforce initiates POST a JSON payload to Jenkins. The Jenkins controller calls engine “headless” CLI commands to run and parallelize the build process across ephemeral, Docker nodes such as Amazon EC2 Spot Instances (one hour or less build time), or Amazon EC2 On-Demand instances. Developers can increase availability with two Jenkins controllers, one in each Availability Zone, behind a load balancer. For some engines, developers may need additional licensing infrastructure configured in additional subnets to vend licenses for the build context each time a concurrent build is run.

The Xcode portion of iOS builds is offloaded to Amazon EC2 Mac instance to sign, build, and export the .IPA file, splitting the process and reducing build times. **AWS Secrets Manager** holds provisioning profiles, private keys, and certificates.

Build artifacts delivered to Amazon S3 trigger third-parties notice flows of success or failures. **AWS Device Farm** enables automated testing.