AWS CodeCommit FAQs | Amazon Web Services

aws.amazon.com (https://aws.amazon.com/codecommit/faqs/)

General

Q: What is AWS CodeCommit?

AWS CodeCommit is a secure, highly scalable, managed source control service that makes it easier for teams to collaborate on code. AWS CodeCommit eliminates the need for you to operate your own source control system or worry about scaling its infrastructure. You can use AWS CodeCommit to store anything from code to binaries, and it works seamlessly with your existing Git tools.

Q: What is Git?

Git is an open-source distributed version control system. To work with AWS CodeCommit repositories, you use the Git command line interface (CLI) or any of the available Git clients. To learn more about Git, see the Git documentation (http://git-scm.com/doc). To learn more about using AWS CodeCommit with Git, see Getting Started with AWS CodeCommit (http://docs.aws.amazon.com/codecommit/latest/userguide/getting-started.html).
Q: Who should use AWS CodeCommit?

AWS CodeCommit is designed for software developers who need a secure, reliable, and scalable source control system to store and version their code. In addition, AWS CodeCommit can be used by anyone looking for an easy to use, fully managed data store that is version controlled. For example, IT administrators can use AWS CodeCommit to store their scripts and configurations. Web designers can use AWS CodeCommit to store HTML pages and images.
Q: How is AWS CodeCommit different from other Git-based source control systems?
AWS CodeCommit offers a number of features not offered by other Git source control systems:

- Fully Managed –AWS CodeCommit eliminates the need to host, maintain, backup, and scale your own source control servers.
- Secure –AWS CodeCommit automatically encrypts your files in transit and at rest. AWS CodeCommit is integrated with AWS Identity and Access Management (IAM), allowing you to assign user-specific permissions to your repositories.
- Highly Available – AWS CodeCommit is built on highly scalable, redundant, and durable AWS services such as Amazon S3 and Amazon DynamoDB.
- Scalable - AWS CodeCommit allows you store any number of files and there are no repository size limits.
- Faster Development Lifecycle - AWS CodeCommit keeps your repositories close to your build, staging, and production environments in the AWS cloud. This allows you to increase the speed and frequency of your

development lifecycle.

Q: How does AWS CodeCommit compare to a versioned S3 bucket?

AWS CodeCommit is designed for collaborative software development. It manages batches of changes across multiple files, offers parallel branching, and includes version differencing (“diffing”). In comparison, Amazon S3 versioning supports recovering past versions of individual files but doesn’t support tracking batched changes that span multiple files or other features needed for collaborative software development.

Using AWS CodeCommit

Q: How do I get started with AWS CodeCommit?
You can sign in to the AWS Management Console (https://console.aws.amazon.com/codecommit), create a repository, and start working with the repository using Git. If you want an introduction to the service, see Getting Started (https://aws.amazon.com/codecommit/getting-started/), which includes a step-by-step tutorial.
Q: How do I create a repository?

You can create a repository from the AWS Management Console or by using the AWS Command Line Interface (AWS CLI) (http://docs.aws.amazon.com/cli/latest/reference/codecommit), the AWS SDKs (https://aws.amazon.com/tools/), or the AWS CodeCommit APIs (https://aws.amazon.com/documentation/).
Q: How do I update files in my repository?

You can edit your files directly from the CodeCommit console or you can use Git to work with the repository. For example Git commands, you can use the&nbsp;git clone&nbsp;command to make a local copy of the AWS CodeCommit repository. Make changes to the local files and use the git commit command when you’re ready to save the changes. Finally, use the *git push* command to upload the changes to the AWS CodeCommit repository. For step-by-step instructions, see Getting Started with AWS CodeCommit (http://docs.aws.amazon.com/codecommit/latest/userguide/getting-started.html).
Q: How do I import my existing repository to AWS CodeCommit?

You can use Git to import any existing Git repository to AWS CodeCommit. For other repositories, such as Subversion and Perforce, you can use a Git importer to first migrate it to a Git repository. For step by step instructions on importing Git repositories, see Migrate an Existing Repository to AWS CodeCommit. (http://docs.aws.amazon.com/codecommit/latest/userguide/how-to-migrate-repository.html) For step-by-step instructions on importing local or unversioned content, see the Git migration documentation (http://git-scm.com/book/en/v2/Git-and-Other-Systems-Migrating-to-Git).
Q: What Git operations are currently supported by AWS CodeCommit?
AWS CodeCommit currently supports clone, pull, push and fetch commands.
Q: Does AWS CodeCommit support Git submodules?

Yes. AWS CodeCommit can be used with Git repositories that include submodules.
Q: What are the service limits when using AWS CodeCommit?

For information on the service limits, see Limits (http://docs.aws.amazon.com/codecommit/latest/userguide/limits.html).
Q: What is the maximum size for a single file that I can store in CodeCommit?

A single file in a repository cannot be more than 2 GB in size.
Q: How do I backup my repository?

If you have a local copy of the repository from doing a full&nbsp;git clone, you can use that to restore data. If you want additional backups, there are multiple ways to do so. One way is to install Git on your backup server and run a scheduled job that uses the git clonecommand to take regular snapshots of your repository. You can use git pull instead of git clone if you want to copy only the incremental changes. Note that these operations may incur an additional user and/or request charges based on how you setup the backup server and the polling frequency.
Q: How do I restore a deleted AWS CodeCommit repository?

Deleting an AWS CodeCommit repository is a destructive one-way operation that cannot be undone. To restore a deleted repository, you will need to create the repository again and use either a backup or a local copy from a full clone to upload the data. We recommend using IAM policies along with MFA-protection to restrict users who can delete repositories. For more details, see the Can I use AWS Identity and Access Management (IAM) to manage access to AWS CodeCommit? question in the Security section of the FAQ.
Q: How do I manage code reviews with AWS CodeCommit?

CodeCommit supports code reviews and enables you to set permissions on branches of your code. Please see our documentation for help with code reviews or branch-level permissions.

Q: How do I integrate my continuous integration system with AWS CodeCommit?

Continuous Integration (CI) systems can be configured to use Git to pull code from AWS CodeCommit. For examples on using CI systems with AWS CodeCommit, see our blog post (http://blogs.aws.amazon.com/application-management/post/Tx1C8B98XN0AF2E/Integrating-AWS-CodeCommit-with-Jenkins) on integrating AWS CodeCommit with Jenkins.
Q: How do I create webhooks using AWS CodeCommit?

In the Amazon Simple Notification Service (SNS) console, you can create a SNS topic with an HTTP endpoint and the desired URL for the webhook. From the AWS CodeCommit console, you can then configure that SNS topic to a repository event using triggers.
Q: Can I get a history of AWS CodeCommit Git operations and API calls made in my account for security analysis and operational troubleshooting purposes?

Yes. You can review recent CodeCommit events, including Git operations and API calls, in the AWS CloudTrail console. For an ongoing record of events you can create a trail and log events in an Amazon S3 bucket. For more information, see Logging AWS CodeCommit API Calls with AWS CloudTrail.

Security

Q: Can I use AWS Identity and Access Management (IAM) to manage access to AWS CodeCommit?

Yes. AWS CodeCommit supports resource-level permissions (http://docs.aws.amazon.com/IAM/latest/UserGuide/PermissionsOverview.html). For each AWS CodeCommit repository, you can specify which users can perform which actions. You can also specify AWS multi-factor authentication (MFA) for a CodeCommit action. This allows you to add an extra level of protection for destructive actions such as deleting repositories. In addition to the AWS CodeCommit APIs, you can also specify git pull and git push as actions to control access from Git clients. For example, you can create a read-only user for a repository by allowing that user access to git pull but not git push on the repository. For more information on using IAM with AWS CodeCommit, see Authentication and Access Control for AWS CodeCommit (https://docs.aws.amazon.com/codecommit/latest/userguide/auth-and-access-control.html). For more information on authenticating API access using MFA, see Configuring MFA-Protected API Access (http://docs.aws.amazon.com/IAM/latest/UserGuide/MFAProtectedAPI.html).

Q: What communication protocols are supported by AWS CodeCommit?

You can use either the HTTPS or SSH protocols or both to communicate with AWS CodeCommit. To use HTTPS, first install the AWS CLI. The AWS CLI installs a Git credential helper that can be configured with AWS credentials. It automatically signs all HTTPS requests to AWS CodeCommit using the Signature Version 4 (http://docs.aws.amazon.com/general/latest/gr/signature-version-4.html) signing specification. To use SSH, users create their own public-private key pairs and add their public keys to their IAM users. The private key encrypts the communication with AWS CodeCommit. For step-by-step instructions on setting up HTTPS and SSH access, see the Setting up AWS CodeCommit (http://docs.aws.amazon.com/codecommit/latest/userguide/setting-up.html) page.

Q: What ports should I open in my firewall for access to AWS CodeCommit?

You will have to open outbound access to an AWS CodeCommit service endpoint on port 22 (SSH) or port 443 (HTTPS).

Q: How do I encrypt my repository in AWS CodeCommit?

Repositories are automatically encrypted at rest. No customer action is required. AWS CodeCommit uses AWS Key Management Service (KMS) to encrypt repositories. When you create your first repository, an AWS-managed CodeCommit key is created under your AWS account. For details, see Encryption for AWS CodeCommit Repositories (http://docs.aws.amazon.com/codecommit/latest/userguide/encryption.html).

Q: Can I enable cross-account access to my repository?

Yes. You can create an IAM role in your AWS account to delegate access to a repository to IAM users in other AWS accounts. The IAM users can then configure their AWS CLI to use AWS Security Token Service (STS) and assume the role when running commands. For details see Assuming a Role (http://docs.aws.amazon.com/cli/latest/userguide/cli-roles.html) in the AWS CLI documentation.

Regions

Q: Which regions does AWS CodeCommit support?

Please refer to Regional Products and Services (https://aws.amazon.com/about-aws/global-infrastructure/regional-product-services/) for details of CodeCommit availability by region.

Billing

Q: How much does AWS CodeCommit cost?

AWS CodeCommit costs $1 per active user per month. For every active user, your account receives an additional allowance of 10 GB-month of storage and 2,000 Git requests for that month. Unused allowance for storage and Git requests does not carry over to later months. If you need more storage or Git requests for your users, additional usage will be charged at $0.06 per GB-month and $0.001 per Git request. Users may store as many Git repositories as they would like. Your usage is calculated each month across all regions and automatically applied to your bill. Please see the pricing page for more details.

Q: What is the definition of an active user in AWS CodeCommit?

An active user is any unique AWS identity (IAM user/role, federated user, or root account) that accesses AWS CodeCommit repositories during the month, either through Git requests or by using the AWS Management Console. A server accessing CodeCommit using a unique AWS identity counts as an active user.

Q: Which Git requests are considered towards the monthly allowance?

A Git request includes any push or pull that transmits repository objects. The request does not count towards your Git request allowance if there is no object transfer due to local and remote branches being up-to-date.

Learn more about AWS CodeCommit pricing

Visit the pricing page

Ready to build?

Get started with AWS CodeCommit (https://console.aws.amazon.com/console/home)

Have more questions?

Contact us

Register for re:Invent Bootcamps

Learn from AWS experts at exam prep bootcamps - space is limited!

(https://reinvent.awsevents.com/learn/bootcamps/?sc_icampaign=aware_reinvent_bootcamps2019_aws&sc_ichannel=ha&sc_icontent=awssm-2448&sc_iplace=2up&trk=ha_awssm-2448)

Get AWS Certified

AWS Certifications can help you advance your career and boost your earning power

Page Content

General Using AWS CodeCommit Security Regions

Billing

aws.amazon.com (https://aws.amazon.com/codecommit/faqs/)