An In-Depth Look at How CodeGuard Works

The following is an in-depth guide through CodeGuard's backup and restore processes from start to finish for our FTP/SFTP and MySQL/MS-SQL backup tool.

Contents:

Connect Site

First, connect your site by giving CodeGuard your FTP/SFTP and MySQL/MS-SQL information. It's that easy.

Connections Options

The first step in activating CodeGuard is to connect your website or database with our servers. There are several options available, and the right one depends on your level of sophistication, goals, and server configuration. At a high level, you really have two choices: (1)FTP/SFTP or (2) MySQL/MS-SQL.

Pros and Cons

FTP

Pros	Cons
Supported by almost all hosting providers	Not secure
Access to all files, even those outside of your website content (logs, mail, user-created folders)	Have to set up database manually
Ability to select which folders should and should not be backed up	

SFTP

Pros	Cons
Secure Transfer	Not supported by all hosting providers
Access to all files, even those outside of your website content (logs, mail, user-created folders)	Have to set up database manually
Ability to select which folders should and should not be backed up	

MySQL

Pros	Cons
Can connect directly to the database or use an SSH tunnel to connect	You may have to whitelist CodeGuard's IP addresses for us to connect
Backs up all content accessible by database user	

MS-SQL

Pros	Cons
Backs up all content accessible by database user	There is no option to use an SSH tunnel, only direct connections
	You may have to whitelist CodeGuard's IP addresses for us to connect

Initial Backup

Now we take an initial backup of your site. During the process, you'll be able to view real-time updates on its progress.

Initial Backup Process

The initial website or database backup is a complete data retrieval of all files that CodeGuard has access to. Depending on the number of files and total size, the initial backup can take up to 4 hours. After this first backup, future backups are differential, both in the files that are transferred and the files that are stored on your behalf. CodeGuard relies upon a queueing system and our backup process is not unlike FedEx package pickup and delivery. Our process is so similar, in fact, that we modeled our user experience after their process.
The four main steps of the initial backup are (1) process initiation (2) file pick up (3) file transit, and (4) final delivery. Process initiation contains verifying the credentials and transmitting them to CodeGuard. File pick up begins with an analysis of the file structure and creation of a git repository within Amazon's Elastic Compute Cloud (EC2). The transit process begins after the file structure has been analyzed and the list of the files to be transmitted is finalized.
Files are then transferred to EC2. After all files are transferred to EC2, they are committed to the git repository on EC2, but do not remain there for long. As soon as the git commit is finished, the files are compressed and sent to Amazon Simple Storage Service (S3), where they are encrypted using industry-leading AES 256 bit techniques. The last step of process is the deletion of the files from their temporary storage on EC2.

Detailed Walkthrough

FTP/SFTP Static Content

We capture all file content and the process can take up to 4 hours depending on the size of your site, number of files on your site, and the wait in the queue. All subsequent backups will not take as long or be as resource intensive.

1. Test the connection to the site using the same protocol (FTP, SFTP) that the backup will use
2. Create a Git repository on the local server instance (i.e. EC2)
3. Build a list of files for the site
4. Add all of those files to the download queue
5. Download each file to the Git repository (on the local machine)
6. Commit all the downloaded files to the Git repository
7. Create a tarred and gzipped archive of the Git repository
8. Upload the archive to Amazon S3 which simultaneously encrypts the archive and all of its content
9. Build the file mix
10. Record the statistics from the backup

Dynamic Content

The database backups are stored as flat text files and the content of the text files is a list of executable SQL statements which, when run, will recreate the database in its entirety. The tables are stored this way so that when a user initiates a restore, no transformation of the data is needed. It can be run as-is to restore the database.
We connect and export the entire contents of the database using mysqldump. This provides us with the database schema (the list of all of the database tables and the columns they contain). The content of the database tables is then transformed into a format that allows us to use version control (Git) to track the changes made to your data. The process is sequential and a failure at any step will stop the process. The specific flow of the process is as follows:

Dynamic Content Steps

1. Connect to the database using the specified method (direct MySQL, tunnel over SSH, or direct MS-SQL)
2. Create a Git repository on the local server instance (i.e. EC2)
3. Begin the database export using the mysqldump tool for MySQL databases or the tool that we developed specifically for MS-SQL databases, saving it in the Git repository
4. Commit the backup file to the Git repository
5. Create a tarred and gzipped archive of the Git repository
6. Upload the archive to Amazon S3 which simultaneously encrypts the archive and all of its content
7. Record the statistics from the backup (number of lines/rows added and deleted)

Monitoring

Now we monitor your site daily for changes, and send a ChangeAlert email to you if there are any changes detected.

Monitoring Overview

CodeGuard notifies you anytime anything changes within the source code of your website. Additions, modifications, and deletions between each version of your website can be viewed in your dashboard. **Email notifications are also sent to inform users of what has changed.** This is accomplished through two main vehicles: the front-end comparison analysis engine, and version control systems on the back-end.
To reduce the loads on our users' servers, CodeGuard only transmits the files that have changed, as opposed to complete backups each time. The way CodeGuard determines which files have changed is with our comparison analysis engine. This engine lists and then compares the files with the previous backup. If any changes are found, only the changed files are transmitted.
The changed files are received in CodeGuard's servers, and then placed in git, creating a new backup for the website. Since database files can be reduced to small sizes, no up-front comparison is performed on the database. CodeGuard simply compresses the databases and sends for analysis by the back-end version control system.

Static Content

FTP/SFTP

CodeGuard uses S/FTP to list all of the files and folders in your site, just like you would with an FTP client. Any folders, and the files within them, that were unselected during the initial backup are not checked. The duration of this process varies based on the number of files in your site and their sizes.
Once the full list of files has been created, CodeGuard compares the list of files to the list received during the previous backup. Any files that have changed are marked to be downloaded
Changed files are based on a number of factors:

- File size
- Last modification time
- User or group ownership
- File permissions

Dynamic Content

MYSQL/MS-SQL

There isn’t a ‘Monitoring’ process for MySQL or MS-SQL. We get all of the content all the time because there is no generic way to detect the changes since the last backup. Once we've downloaded the database, we will then compare it to the previous backup for changes. These changes are not included in your ChangeAlert notifications.
It is noted within the user dashboard if changes were detected. At the current time, users are not notified via email when changes are made to the database

Backup Again

After monitoring we continuously backup your site again and again if we detect changes.

Backup Overview

If the monitoring process detects any changes to the website source code, the files that changed or were added are transmitted to the CodeGuard repository. If any files are deleted, a record of the deletions is also transmitted to the repository. For deletions, those files are removed from the repository; additions are added and changes are overwritten to form a new version of the backup, which is shown within the user's dashboard.
Since database files can be reduced to small sizes, CodeGuard transmits the entire database, which is compared with the previous version; if changes are found, a new version appears in the user's dashboard.

Backup Comparisons

FTP/SFTP SOURCE

All backups after the first only retrieve the files that were marked as changed during the monitoring process. If no changes are detected, the process ends after the monitoring step. This is roughly the same process as the initial backup with a few exceptions. This process also includes the monitoring step which is detailed above. Steps that differ from the initial backup are in **bold**:

1. Test the connection to the site using the same protocol (FTP/SFTP) that the backup will use
2. **Download the existing repository archive from S3 and extract it on the local server instance (i.e. EC2)**
3. Build a list of files for the site using the methods described previously
4. **Monitor the files, add only the changed ones to the download queue**
5. Download each file to the Git repository (on the local machine)
6. Commit all of the downloaded files to the Git repository
7. Create a tarred and gzipped archive of the Git repository (which contains all of the backed up site files)
8. Upload the archive to Amazon S3 which simultaneously encrypts the archive and all of its content
9. **Remove any previous versions of this backup that exist on S3**
10. Build the file mix data
11. Record the statistics from the backup

MySQL/MS-SQL SOURCE

This process is the similar to the initial backup, with the addition of these two steps:

1. Instead of creating a new Git repo -- Download the existing repository archive from S3 and extract it on the local server instance (i.e. EC2)
2. After uploading to S3 -- Remove any previous versions of this backup that exist on S3

Restore

Finally, you can restore your site back to any previous version with the click of a button

FTP/SFTP Whole Site Restore

- Can be time consuming since we do a backup before doing the restore
- The ‘pre-restore’ backup is necessary to get the current state of the site. Things may have changed dramatically since the last completed backup
- Process:
    1. The user selects the version to restore
    2. A backup is performed using the same process as ‘Backup Again’
    3. Two lists of site content are built: one of the version to be restored and one of the current state of the site
    4. The difference between those two sites produces discrete operations for files and folders. These operations are placed in a queue for processing
    5. The backup is downloaded (if it doesn’t already exist) and the version to be restored is checked out
    6. Open a connection to the host site
    7. Process the operations queue and log the results of each operation in this order: delete files, delete folders, create folders, upload files, and finally, change file permission
    8. Record the statistics from the restore
    9. Send email notifications

FTP/SFTP Individual File Restore

- Individual file restore happens without a ‘pre-restore’ backup
- User can select files using a typeahead that searches the files for a particular backup version
- Process:
    1. User selects 1 - n files in the UI, which are directly added to the operations queue
    2. The site repo is downloaded and the version that contained the selected files is checked out
    3. Connection tested and opened using the appropriate client (FTP/SFTP)
    4. The selected files are uploaded, and their permissions are set
    5. Record the statistics from the restore
    6. Send email notifications

MySQL/MS-SQL Database Restore

- This process uses the MySQL or MS-SQL client to perform the restore
- Process:
    1. A pre-restore backup is performed
    2. Download repo from S3
    3. Checkout target restore version
    4. Open connection to remote database using MySQL directly, through a SSH tunnel, or MS-SQL
    5. Execute the database backup file
    6. Drops and recreates each table
    7. Imports table data
    8. Record the statistics from the restore