PTC®

File Vaulting for Configuration Management Guide

PTC Integrity™ 10.8

PTC Inc., 140 Kendrick Street, Needham, MA 02494 USA

Contents

Overview of File Vaulting....7
Key Considerations for File Vaulting....8
Configuring the File Vault....9
Migrating Existing Revisions to the File Vault....13
File Vault Migration Logging....17
Managing the File Vault....19
Changing the File Vault Location....20
Disabling the File Vault....20
Getting Notified When the File Vault is Unavailable....21
Backing Up and Restoring File Vaults....21

1

Overview of File Vaulting

Key Considerations for File Vaulting.....8

File vaulting enables you to store data as files on the file system instead of in a database. The Configuration Management file vault stores Integrity revision data to a logical container called a vault. The vault is directly mapped to a file system available from the Integrity server.

You can configure and enable Configuration Management file vault to upload all new, eligible revisions for all or select configuration management projects. Eligible file vault data includes:

- revisions of binary archives
- revisions of text archives whose archive storage format is set to store by reference

For more information on the store text by reference archive format, including details on the maximum size for text revisions and working files, see the topic "General Policy Options" in the *PTC Integrity Server Administration Guide*.

Note

Existing revision data is not automatically added to the Configuration Management file vault. For more information on migrating existing revision data, see Migrating Existing Revisions to the File Vault on page 13.

Key Considerations for File Vaulting

In environments where database storage is limited or expensive, storing revision bulk data in a database can be prohibitive. File vaulting data can offer an effective control mechanism for database growth over time. However, PTC recommends that Integrity administrators weigh the following considerations carefully before implementing file vaulting.

Data Corruption Risk

To mitigate the risk of data corruption:

- PTC recommends a RAID (Redundant Array of Independent Disks) configuration for the configuration management file vault.
- Once the file vault is operational, PTC recommends that you synchronize the backup of both the database component and the file system including all vaulted data. For more information on file vault backups, see Backing Up and Restoring File Vaults on page 21.
- Ensure that the **Enable checksum verification of member revisions** configuration management policy is enabled in the Integrity Administration Client. This policy is enabled by default. This policy ensures data integrity between the Integrity server and the client by calculating checksums for each revision, and verifying those checksums against the database. For more information on this policy, see the topic "General Policy Options" in the *PTC Integrity Server Administration Guide*.

Existing Revision Data Migration

You can migrate existing revision data from the database to the Configuration Management file vault using a bulk migration diagnostic command. However, this optional one-time migration step is not bidirectional. Once Integrity revisions are stored in the file vault, there is no way to move those revisions back in to the database. Access to the file vault must be maintained in perpetuity to access vaulted revisions. For more information on migrating existing revisions, see Migrating Existing Revisions to the File Vault on page 13.

File Vault Security

When using the file vault to store revision data outside of the primary Integrity repository, securing the file vault location is the responsibility of the system administrator.

2

Configuring the File Vault

To configure the Configuration Management file vault on an Integrity server:

1. Stop the Integrity server.
2. Enable vault support by modifying the following property in the `si.properties` file under *`IntegrityServerinstalldir`*`/config/properties`:

   `si.vault.enable=true`

   When set to `true`, this property enables support for file vaulting on the Integrity server. When set to `false`, file vaulting is disabled.
3. Specify the vault location for this Integrity server instance. Modify the following property in the `si.properties` file under *`IntegrityServerinstalldir`*`/config/properties`:

   `si.vault.location=root directory of the Configuration Management file vault`

   For example:

   `si.vault.location=D:/FileVaultDirectory`

Note

The file vault location specified must be a directory other than the Integrity server installation directory.

The Integrity server requires read/write access to the file vault directory. Ensure that the account that is used to run the Integrity server has read/write access.

The path format used depends on which operating system the Integrity server is running on. If the server is running on Linux or some other form of Unix, the vault path must be a Unix-style path. On a machine running Windows, the vault path must be a Windows-style path, a direct UNC path, or a mapped drive for a network path. Forward slashes "\" are supported but they must be escaped with a second forward slash "\\".

4. Ensure that the vault location directory exists on the file system before starting the Integrity server. If the vault location does not exist, the Integrity server will not start.
5. Start the Integrity server.
   At startup, Integrity performs a check of vault content to ensure that it is synchronized with the database. The sanity check validates that the latest vaulted revision content exists in the specified vault location. If the server startup check fails and the Integrity server fails to start, an administrator can bypass the server startup check to start the server. Modify the following property in the `si.properties` file under *`IntegrityServerinstalldir`*`/config/properties`:
   `si.vault.bypassVaultSanityChecksUponStartup=true`

   When set to `true`, the server startup check is bypassed on startup. By default, this property is set to `false`.

Caution

When the server startup check fails, the server failing to start can be an indication of a more serious issue, such as file vault corruption. PTC strongly recommends that you restore the contents of the file vault from a viable backup before attempting to bypass the failure and start the server. For more information, see Backing Up and Restoring File Vaults on page 21.

6. In the Integrity Administration Client, enable the **Use File Vault for store by reference revisions** configuration management policy. Revision data is not moved to the file vault until this policy is set globally or is set on specific configuration management projects. For more information on setting configuration management policies, see the topic “To set general policies” in the *PTC Integrity Server Administration Guide*. For more information on this policy, see the topic “General Policy Options” in the *PTC Integrity Server Administration Guide*.

Once the Integrity server is started, all newly created, eligible revisions are added to the file vault location for all or specific configuration management projects.

Note

Existing revision data is not automatically added to the Configuration Management file vault. For more information on migrating existing data, see Migrating Existing Revisions to the File Vault on page 13.

3

Migrating Existing Revisions to the File Vault

File Vault Migration Logging 17

You can extract eligible revisions from the Integrity database, and move those revisions to the Configuration Management file vault in a single bulk operation. Eligible revisions include:

- revisions of binary archives
- revisions of text archives whose archive storage format is set to store by reference

When migrating revisions from the database to the vault, only the canonical archive location determines where revisions are stored. Consider the following example:

ProjectA is configured to store revisions in the file vault.
ProjectB is configured to store revisions in the database.
A subproject from ProjectA is shared to ProjectB as a shared subproject.

Because the canonical location of the shared subproject is in ProjectA’s hierarchy, the canonical archive location of members of that shared subproject are under ProjectA as well. The shared subproject's revisions are therefore stored in the file vault.

Migrating Existing Revisions to the File Vault

To migrate existing revisions into the file vault in the CLI:

Note

Administrators require the `AdminServer` permission to migrate existing revisions into the file vault.

Caution

This optional one-time migration step is not bidirectional. Once the revisions are stored in the Configuration Management file vault, there is no way to move those revisions back into the database.

1. Ensure that the file vault is properly configured, and that the file vault directory exists on the Integrity server file system. For more information, see Configuring the File Vault on page 9.
2. Run the file vault migration command to migrate all eligible revisions in the repository to the file vault: `si diag --diag="migrateRevisionsFromDBToFileVaultAsPerPolicy"`.

The `migrateRevisionsFromDBToFileVaultAsPerPolicy` diagnostic command has the following options:

- `path` specifies the configuration management project path to migrate to the file vault. If this parameter is specified, only eligible revisions from the specified project path are migrated. You can specify only a single project using this parameter. If this parameter is not specified, then all eligible configuration management projects are considered for migration.

Note

The project specified in this path parameter must have the **Use File Vault for store by reference revisions** configuration management policy set. If the policy is disabled at the project level, then eligible revisions in that project are not migrated. For more information on setting configuration management policies, see the topic "To set general policies" in the *PTC Integrity Server Administration Guide*.

The project path in this parameter must be specified as the canonical path of the project. For example, to migrate the `demoMigrate` project whose location is `ProjectDirectory/demoMigrate/project.pj`, then in the CLI, exclude `/project.pj` to specify only the canonical project path in the command syntax. For more information on specifying projects using a canonical path, see the CLI man pages.

- `count` specifies the maximum number of archives to be processed and migrated into the file vault. The migration attempts to move all of an archive's revisions, and the oldest revisions receive the highest priority during migration.
- `before` specifies the server date and time after which revisions are not migrated.

For example, to migrate the 100 oldest demoMigrate revisions in the repository to the file vault, run the following command:

```
si diag --diag="migrateRevisionsFromDBToFileVaultAsPerPolicy"
--param=path="/demoMigrate" --param=count=100
```

To migrate all demoMigrate revisions with a creation timestamp before January 1, 2015 at noon, run the following command:

```
si diag --diag="migrateRevisionsFromDBToFileVaultAsPerPolicy"
--param=path="/demoMigrate" --param=before="2015/01/01 12:00:00"
```

Note

If no parameters are specified, then all eligible revisions in the system are permanently migrated from the database to the file vault.

After running the migration command, a migration summary displays as follows:

```
--------Archives Statistics--------
Migrated= , Skipped=
--------Revision Statistics--------
```

```
Migrated= , Failed= , Skipped=
```

Additional details, including revision statistics, are logged to the `DB_to_FileVault_Migration.log` file located under *`IntegrityServerinstalldir`*`/log`.

Migrating Older Revisions to the File Vault

You can migrate older revisions out of the database while keeping new revisions in the database. This special configuration mode requires you to leave the file vault property disabled, and to configure a valid file vault location. Then, you run the migration to vault eligible revisions. All newly created revisions are stored in the database.

To migrate only legacy revisions to the file vault:

1. Stop the Integrity server.
2. In the `si.properties` file under *`IntegrityServerinstalldir`*`/config/properties`, specify the vault location using `si.vault.location=`, and ensure that vault support is disabled as follows: `si.vault.enable=false`.
3. Start the Integrity server.
4. In the Integrity Administration Client, enable the **Use File Vault for store by reference revisions** configuration management policy globally or for a specific project.
5. Run the file vault migration command `si diag migrateRevisionsFromDBToFileVaultAsPerPolicy`. Include the `path` option if vaulting a specific project, and include other supported options as needed, such as the `before` option.

After migrating legacy revisions to the file vault, you can remove the **Use File Vault for store by reference revisions** policy if you do not plan to migrate additional revisions later.

Note

To migrate older revision data when the file vault is already enabled, disable the file vault temporarily and enable the **Use File Vault for store by reference revisions** policy for the project you want to vault. Then, run the migration command for that project. Following the migration, remove that policy and re-enable the file vault to resume vaulting of new revisions.

File Vault Migration Logging

The `DB_to_FileVault_Migration.log` file includes information about the number of revisions that were successfully migrated, as well as revisions that were skipped or were not migrated.

The following columns of information are captured in this log file:

- `Timestamp`: The date and time when each revision in the archive is processed and migrated. Each migrated revision is listed on a separate line under the migration summary of the archive itself.
- `ArchivePath`: The canonical path of the archive that was migrated.
- `Revision`: The archive revision. For example, Revision: 1.15.
- `Status`: Revision status. Can be one of: `SUCCESS`, `FAILED`, or `PARTIAL`.
- If a revision is skipped during migration, one of the following status comments is logged: `Before timestamp filter Criteria not met` or `Already Migrated`.
- If an archive is skipped during migration, the following comment is logged: `Project is not eligible for migration`.
- If some revisions are skipped or fail migration, then the partially migrated archive is logged. For example:

  ```
  2015-07-10 21:44:23.822, ArchivePath:/defectMigrate/demo.txt,Status:PARTIAL,
  Comment:Archive is not completely migrated due to issue encountered during
  migration of some revisions
  ```

- If a revision fails migration, one of the following reasons is logged:
  - Failure due to an exception. For example:

    ```
    2015-07-10 02:25:59.713,ArchivePath:/defectMigrate/1KB_9.txt,
    Revision:1.4,Status:FAILED,Comment:D:\FILE_VAULT_DEFAULT\0\535\351_tmp
    (The system cannot find the file specified)
    ```

  - Failure due to a checksum mismatch. For example:

    ```
    2015-07-10 20:38:52.656,
    ArchivePath:/projmock01n/projmock01n_tempmock01n/127KB_2.txt,
    Revision:1.2,Status:FAILED,Comment:Failed to migrate due to
    Checksum mismatch, computed checksum of 29837 does not match the
    stored checksum of -2392966040148960768.
    Please refer to file 'D:\FILE_VAULT_DEFAULT\4\4473\1920_checksumfailed'
    for further investigation or contact PTC - Integrity Support
    ```

  Each failed file migration is stored in the file vault location, and its filename is appended with `_checksumfailed`. The original file remains in the database. The appended file in the file vault can be used for further investigation, and can then be deleted.

Note

If the file vault migration fails for any reason, the revisions are available in the database.

4

Managing the File Vault

Changing the File Vault Location.....20
Disabling the File Vault 20
Getting Notified When the File Vault is Unavailable 21
Backing Up and Restoring File Vaults.....21

Changing the File Vault Location

Administrators can change the location of the file vault in the file system on the Integrity server. PTC recommends that you always back up file vault content before changing the vault location and moving the contents of the vault.

To change the file vault location:

1. Stop the Integrity server.
2. Create the new file vault directory in the new location. If the vault location does not exist, the Integrity server will not start.
3. Manually copy the contents of the file vault from the current location to the new file vault location.
4. Update the vault location for this Integrity instance to point to the new file vault location. Modify the following property in the `si.properties` file under *`IntegrityServerinstalldir`*`/config/properties`:

   ```
   si.vault.location=root directory of the new
   Configuration Management vault file location
   ```

   Note

   The file vault location specified must be a directory other than the Integrity server installation directory.

   The Integrity server requires read/write access to the file vault directory. Ensure that the account that is used to run the Integrity server has read/write access.

   The path format used depends on which operating system the Integrity server is running on. If the server is running on Linux or some other form of Unix, the vault path must be a Unix-style path. On a machine running Windows, the vault path must be a Windows-style path, a direct UNC path, or a mapped drive for a network path. Forward slashes "\" are supported but they must be escaped with a second forward slash "\\".

5. Start the Integrity server.

Disabling the File Vault

Once revision content is migrated to the Configuration Management file vault, that content is vaulted forever. Disabling file vaulting can prevent any new revisions from being vaulted. When you disable file vaulting, existing vaulted revisions remain in the vault, and new revisions are stored in the database again.

Note

Once the file vault is operational, access to the file vault must be maintained in perpetuity to access vaulted revisions. Also, there is no way to move vaulted revisions back in to the database.

To disable the file vault:

1. Stop the Integrity server.
2. Disable vault support by modifying the following property in the `si.properties` file under *`IntegrityServerinstalldir`*`/config/properties`:

   ```
   si.vault.enable=false
   ```

   When set to `false`, file vaulting is disabled on the Integrity server.
3. Start the Integrity server.

Getting Notified When the File Vault is Unavailable

If the Configuration Management file vault is not accessible, details are logged to *`IntegrityServerInstallDir`*`\log\FATAL.log`.

To be notified of such events, you can configure e-mail notifications with a log level of FATAL in the `mksis.logging.email.threshold` property file. For more information on configuring e-mail notifications, see the topic “Setting Up E-mail Notifications in the *PTC Integrity Server Administration Guide*.

Backing Up and Restoring File Vaults

When preparing for and executing data backups, you must make arrangements to back up both the database contents and the Configuration Management file vault contents. It is the system administrator’s responsibility to:

- Establish appropriate backup and restoration processes and protocols for both the file system and the database.
- Ensure the configuration and use of appropriate tools and technologies to execute the backup and restoration strategies.
- Test all backup and restoration processes frequently to ensure both data integrity and data synchronization between all data stored in file system and the database.

Once the file vault is operational, database backups and file vault backups must be kept in synchronization. At a minimum, the file vault content must match, or be more recent than, backed up database content.

Restoring from a File Vault Backup

To restore a file vault backup, you must stop the Integrity server before copying the files from the backup into the file vault.

When you start the Integrity server, the restored file vault contents are accessible as long as the database content is synchronized. Revisions that were recently added to the file vault are inaccessible if the database content is older than the file vault content.

Note

Integrity does not provide a way to restore files automatically that would have been stored to the file vault if a recent database backup is not available.