



Coverity 2020.12 Upgrade Guide

For Coverity Analysis, Coverity Platform, and Coverity Desktop.

Copyright 2020 Synopsys, Inc. All rights reserved worldwide.

Table of Contents

Upgrading Coverity	iii
1. Upgrade Overview	1
2. Upgrading Coverity Connect	4
2.1. Upgrading instances that use an embedded database	4
2.2. Upgrading instances that use an external database	14
2.3. Coverity Connect upgrade environment variable parameters	19
2.4. Upgrading Coverity Desktop	19
2.5. Upgrading the Coverity Jenkins plug-in	20
2.6. Migrating and upgrading to Coverity Connect (Integrity Manager) from Defect Manager version 4.5.1	21
2.7. Rolling back from Backup-and-Restore or Intermachine Upgrades to the previous Coverity Platform version	21
3. Upgrading Coverity Analysis	23
3.1. Standard Upgrade Procedure	23
3.2. Understanding churn	24
3.3. Preview and production checkers	25
3.4. Upgrading and MISRA changes	25
3.5. New behavior when emitting web application archives	26
4. Test Advisor upgrade impact	27
4.1. False negative violations for impact history	27
4.2. Regenerating test metrics for Test Prioritization	28
4.3. Upgrading Function Coverage Instrumentation and Runtime Library	29
4.4. Test Advisor with Test Advisor QA Edition for C/C++	29
5. Important upgrade considerations	31
5.1. Upgrade considerations for 2020.12	32
5.2. Upgrade considerations for 2020.09	32
5.3. Upgrade considerations for 2020.06	32
5.4. Upgrade considerations for 2020.03	32
5.5. Upgrade considerations for 2019.12	32
5.6. Upgrade considerations for 2019.09	34
5.7. Upgrade considerations for 2019.06	34
5.8. Upgrade considerations for 2019.03	34
5.9. Upgrade considerations for 2018.12	34
5.10. Upgrade considerations for 2018.09	35
5.11. Upgrade considerations for 2018.06	35
5.12. Coverity Connect Web Services API Release and Deprecation Schedule	36
6. Updating Coverity Analysis with Incremental Releases	38
6.1. Installation	38
6.2. How Updates Work	38
6.3. Running cov-install-updates	39
6.4. Rolling Back an Update Session	39
6.5. Other Utilities	39
6.6. Upgrading Coverity Analysis Using Updates	40
A. Coverity Legal Notice	42
A.1. Legal Notice	42

Upgrading Coverity

This guide covers the process of upgrading and updating your Coverity deployment.

1. An upgrade brings your deployment up-to-date with a newer major release, e.g. 2018.12.
2. An update brings your deployment up-to-date with a newer update release, e.g. 2018.06-3.

Updates were introduced in the 2017.07-SP2 release, and provide the means to install new or updated features between major releases. At present, updates apply only to Coverity Analysis. Normally, updates apply only to the most recent major release, and are not back-ported to earlier major releases. Updates are only supported for these platforms: `linux64`, `linux`, `win32`, `win64` and `macosx`.

Chapter 1. Upgrade Overview

This chapter provides an overview of the process for upgrading the Coverity Platform and Coverity Analysis products to the latest version, with a few tips.

1. Upgrade your license files.

Your license must be set to a supported product version. The Downloads page only shows files that match your current license version. There is no cost to upgrade or downgrade your license version. You can do it yourself, at any time, on your Licenses page. If your Licenses page or Downloads page is blank, you must ask a designated Coverity License Administrator in your organization to download the files or modify the license for you, or to grant you the authority to do it yourself.

When your license is set to the right version, download the license to get the right entitlements for that version. If your license is node locked, you might need to use the Coverity Host ID utility to get your Host ID, and then rehost your license before downloading it.

2. Download installers and documentation.

All Coverity installer files are on the Coverity Community Portal [site](#). To download Coverity software, you must have a user account.

Coverity sent the Coverity Customer Center credentials to your license administrator. If you are a license administrator and did not receive the credentials, or if you do not know who your license administrator is, contact software-integrity-license@synopsys.com.

Download the files you need, based on your installation platform:

- **Coverity Platform package** - This is your Platform installer (server). Download for the OS platform on which you run Coverity Connect.
- **Coverity Analysis package** - This is your Analysis installer (client). Download for every OS platform on which you build and analyze your source code with Coverity analysis components.
- **Utilities package** - This is your Host ID utility. Download for every OS platform on which you plan to install Coverity Analysis or Coverity Platform, if your license for that platform requires a Host ID. Note that on some machines with complex or multiple network connections, the platform-specific Host ID Utility will not produce correct results. If you have problems, download and use the platform-independent Java Host ID utility.
- **Documentation package** - This is a subset of the complete documentation collection included in the installer files. This subset includes *Coverity 2020.12 Release Notes* (this guide) and the *Coverity 2020.12 Installation and Deployment Guide* (which contains installation instructions). Download the Installation Documentation bundle of your choice.

3. Understand the impact of changed or updated features for your upgrade to 2020.12.

Before upgrading your Coverity products, it is strongly recommended that you consult Chapter 5, *Important upgrade considerations* for a list of important changes that might affect your standard workflow.

4. Adjust performance settings.

Adjust your PostgreSQL and JVM performance tuning settings for best performance during the upgrade and subsequent production operations. Incorrect settings with a large database can cause an upgrade to take 5x to 10x longer, or to fail and require multiple restarts.

Tuning is especially critical when upgrading from a 5.x version to the latest version, because some steps in the upgrade process might require considerably more system resources than normal production operations. For more information, see Section 2.3, “Coverity Connect upgrade environment variable parameters”.

If you have any doubt about your tuning settings, send the following information to `software-integrity-support@synopsys.com` and ask for guidance:

- From what version are you upgrading? To what version are you upgrading?
- What is the size of your database on disk? What size is your database backup file?
- Is the machine dedicated to Coverity Connect (y/n)?
- What is the amount of system RAM in GB?
- Is your Coverity Connect database packaged with the Coverity Platform installer (an embedded PostgreSQL database), or is it an external PostgreSQL database?
- On what OS platform are you running Coverity Connect?
- What is the number of CPU cores on the machine, excluding hyper-threading?

5. Upgrade Coverity Platform.

Follow the process documented in Chapter 2, *Upgrading Coverity Connect*. This section covers the following:

- Upgrading Coverity Connect with an embedded database.
- Upgrading Coverity Connect with an external database.
- Upgrading a Coverity Connect enterprise cluster.
- Upgrading Coverity Desktop.

Tip

- **Running a test upgrade** - Run a test upgrade first, using your production database backup in a test environment. For large upgrades, this test should reveal possible problems and help you estimate the time that your production upgrade will require.
 - **Committing between versions** - You can commit analysis results from the same or earlier version of Coverity Analysis to the same or later version of Coverity Connect. However, if you want to incorporate the 7.0.x functionality of running a single, multi-language analysis
-

instead of continuing to run separate analyses divided by programming language, you must commit the analysis results to the same stream in a version 7.0.x instance of Coverity Connect.

- **Coverity Desktop plug-in versions** - Starting with version 7.7.x, the Coverity Fast Desktop plugins are supported to work with newer versions of the Coverity Connect server (within the documented support timelines) so that Coverity Connect can be upgraded without users needing to upgrade their 7.7.x IDE plugins and analysis tools at the same time.

Please note, however, that the Coverity Desktop and analysis tool versions must continue to match each other.

6. Upgrade Coverity Analysis.

Follow the process documented in Chapter 3, *Upgrading Coverity Analysis*.

Also, see the tips in the previous step about coordinating your upgrade activities. After upgrading, it is important to understand changes to commands and checkers, and to update any of your internal scripts that are affected by these changes.

Chapter 2. Upgrading Coverity Connect

Table of Contents

2.1. Upgrading instances that use an embedded database	4
2.2. Upgrading instances that use an external database	14
2.3. Coverity Connect upgrade environment variable parameters	19
2.4. Upgrading Coverity Desktop	19
2.5. Upgrading the Coverity Jenkins plug-in	20
2.6. Migrating and upgrading to Coverity Connect (Integrity Manager) from Defect Manager version 4.5.1	21
2.7. Rolling back from Backup-and-Restore or Intermachine Upgrades to the previous Coverity Platform version	21

This section describes how to upgrade a Coverity Connect deployment. (Note that Coverity Connect includes Coverity Policy Manager and downloadable packages for other components, including Coverity Reports and the Coverity Desktop plugins.)

Important!

Depending on the size of the Coverity Connect database and the number of versions in the Coverity Connect upgrade, the upgrade process might take several hours to complete. Also, before performing an upgrade on a Coverity Connect instance, consult Chapter 5, *Important upgrade considerations* for a list of important product changes that may affect your Coverity Platform workflow.

This section is applicable to both Coverity Connect standalone deployments (one Coverity Connect instance) and Coverity Connect clustered deployments (more than one Coverity Connect instance working together). For more information about deployments, see "Chapter 5.4, Coverity Connect deployment options" in *Coverity 2020.12 Installation and Deployment Guide* [↗](#).

Upgrading a Coverity Connect clustered deployment requires upgrading one Coverity Connect instance at a time. The procedures for deciding which instance to upgrade first and how to upgrade each instance vary depending on whether the instance uses an embedded database or an external database. (An embedded database is installed by default with Coverity Connect, while an external database is created and maintained separately.)

To upgrade each Coverity Connect instance in your deployment, consult the appropriate section:

- Section 2.1, "Upgrading instances that use an embedded database"
- Section 2.2, "Upgrading instances that use an external database"

2.1. Upgrading instances that use an embedded database

 **Important**

All upgrades must be performed using one of the upgrade options in the Coverity Connect installer. Do not perform an upgrade manually. For example, do not use command line techniques to backup an existing instance and then restore it to a new, upgraded instance.

Before using the installer, determine the type of upgrade, as described in this section.

2.1.1. Choosing the type of upgrade

This section helps you choose the type of upgrade. Note that each type of upgrade corresponds to the Coverity Connect *installer options* as shown in the following table. This chapter uses the terminology specified in the table to describe upgrade types:

Table 2.1. Installer options

Type of upgrade	Installer option(s)
In-place upgrade	"In-place Upgrade" option
Backup-and-restore upgrade	"Backup-and-restore" option
Intermachine upgrade	"Upgrade Preparation" option followed by "Intermachine Upgrade" option

Upgrade types are described in the following subsections.

Intermachine upgrade

Use the Intermachine upgrade if you want the upgraded Coverity Connect instance to exist on a new host machine. Otherwise, use the In-place upgrade or the Backup-and-restore upgrade.

In-place upgrade versus Backup-and-restore upgrade

An In-place upgrade is faster and uses less disk space than a Backup-and-restore upgrade. However, a Backup-and-restore upgrade keeps the old instance intact, which is useful if you want to create a staging environment.

An In-place upgrade transforms an existing Coverity Connect instance, including all of its data, into a new Coverity Connect instance in the same directory location on the same machine. Essentially, an In-place upgrade does the following:

- Optionally, backs-up the existing database
- For PostgreSQL major version changes, modifies the data storage format in-place
- If required by Coverity Connect, upgrades the database schema
- Updates non-database state (the configuration not stored in the database)

A Backup-and-restore upgrade backs-up the entire, existing Coverity Connect instance. Then the Backup-and-restore upgrade restores the Coverity Connect instance, including the database, to a new location (a different directory than where the existing instance is installed) on the same machine without affecting the existing instance. Essentially, a Backup-and-restore upgrade does the following:

- Backs-up the existing database
- Installs a new Coverity Connect instance in the new location
- Restores the database to the new location
- If required by Coverity Connect, upgrades the database schema in the new installation
- Copies non-database state (the configuration not stored in the database) to the new installation

2.1.2. Performing the upgrade: standalone deployments

The following sections explain the upgrade process.

2.1.2.1. Upgrade prerequisites

Make sure the instance you want to upgrade meets the following prerequisites:

- **Important:** Make sure you are not running anti-virus software on a system with a Coverity Connect database because it impacts performance and can even interfere with correct functioning of the database, possibly including data corruption. If you must run anti-virus software, disable it for the duration of the upgrade or exempt the <cc_install_dir>/<database> directory from anti-virus inspection.
- Make sure there is sufficient free space in these locations:
 - The location for the backup if one is needed
 - The volume where the new instance will be installed (if you are not doing an In-place upgrade)

The free space should be roughly 3x the size of the database, as determined by the size of the <cc_install_dir>/<database> directory's contents.

- Make sure you have enough disk space for the upgrade.
- Make sure that you have the following information prior to launching the installer:
 - Existing Coverity Connect installation directory
 - Destination installation directory (if you plan to perform a Backup-and-restore or Intermachine upgrade)
 - Desired backup directory location (if you plan to create a database backup)
 - Location of your Coverity Connect license file
 - Desired ports for Coverity Connect communications (for Backup-and-restore and Intermachine upgrades)

2.1.2.2. Performing an In-place upgrade (standalone instances)

This section describes how to perform an In-place upgrade on a Coverity Connect standalone instance that uses an embedded database.

 **Note**

Make sure you have read all of the preceding sections in this chapter before proceeding.

To perform an In-place upgrade on a standalone instance that uses an embedded database:

 **Note**

If you are performing an In-place upgrade, the upgrade may fail if the path of the chosen cov-platform installation directory is overly long. As reference, an upgrade on Windows may succeed with paths up to 107 characters, including slash characters. On Linux, 85 characters may be too long. This restriction is due to PostgreSQL.

 **Note**

On Linux systems, you must **not** be logged in as root to upgrade Coverity Connect.

1. Make sure you have full permissions (rwx) on the existing installation directory.
2. On the machine where you want to install Coverity Connect and other Coverity Platform components, download the Coverity Platform (this is the correct installer for Coverity Connect) installer file for your operating system.
3. Run the installer program for your operating system.

For Windows, we recommend that you install Coverity Connect with Windows Administrator privileges (installing Coverity Connect as a service requires it). To do so, right click the installer and choose Run as Administrator.

To install, double-click the .exe program.

For Linux, run the installer script in a Bourne shell, for example:

```
> ./cov-platform-linux64-[version].sh
```

 **Note**

Depending on the size of your database, the upgrade process can take a long time. If you are performing this operation by way of an SSH terminal, we recommend you use a persistent terminal (such as Screen) in case your session is interrupted.

The installer uses a text-based console mode or a graphical mode. The installation choices for graphical and console modes are equivalent. To install using graphical mode on Linux, append the `-g` option to the command above.

Note that you can change from graphical to silent installer (command line) as described in Chapter 1.2. Coverity Platform installation modes in the Coverity Installation and Deployment Guide. For details about the silent installer options and parameters, see 1.2.3. Coverity Connect silent installer, and in particular see the sub-section In-place upgrade parameters.

4. Complete the installation process:

- a. Select and accept the license agreement for your region of the world.
- b. Select the **In-place Upgrade** option.
- c. Follow the installation prompts.

 **Important!**

When you are prompted to choose a performance configuration, you can choose Production or Restore. If you choose Production, the installer runs `cov-admin-db tune`. This can improve Coverity Connect performance by re-tuning database performance. If you choose Restore, the installer does not run `cov-admin-db tune`. For details about `cov-admin-db tune`, see the *Coverity 2020.12 Command Reference*.

2.1.2.3. Performing a Backup-and-restore upgrade (standalone instances)

This section describes how to perform a Backup-and-restore upgrade on a Coverity Connect standalone instance that uses an embedded database.

 **Note**

Make sure you have read all of the preceding sections (excluding Performing an In-place Upgrade) in this chapter before proceeding.

Make sure you know the location for the new Coverity Connect installation directory prior to launching the installer.

To perform a Backup-and-restore upgrade on a standalone instance that uses an embedded database:

 **Note**

If you are performing a Backup-and-restore upgrade, the upgrade may fail if the path of the chosen `cov-platform` installation directory is overly long. As reference, an upgrade on Windows may succeed with paths up to 107 characters, including slash characters. On Linux, 85 characters may be too long. This restriction is due to PostgreSQL.

 **Note**

On Linux systems, you must **not** be logged in as root to upgrade Coverity Connect.

1. Make sure you have full permissions (rwx) on the existing installation directory and the new installation directory.
2. On the machine where you want to install Coverity Connect and other Coverity Platform components, download the Coverity Platform (this is the correct installer for Coverity Connect) installer file for your operating system.
3. Run the installer program for your operating system.

For Windows, we recommend that you install Coverity Connect with Windows Administrator privileges (installing Coverity Connect as a service requires it). To do so, right click the installer and choose Run as Administrator.

To install, double-click the `.exe` program

For Linux, run the installer script in a Bourne shell, for example:

```
> ./cov-platform-linux64-[version].sh
```

 **Note**

Depending on the size of your database, the upgrade process can take a long time. If you are performing this operation by way of an SSH terminal, we recommend you use a persistent terminal (such as Screen) in case your session is interrupted.

The installer uses a text-based console mode or a graphical mode. The installation choices for graphical and console modes are equivalent. To install using graphical mode on Linux, append the `-g` option to the command above.

Note that you can change from graphical to silent installer (command line) as described in Chapter 1.2. Coverity Platform installation modes in the Coverity Installation and Deployment Guide. For details about the silent installer options and parameters, see 1.2.3. Coverity Connect silent installer, and in particular see the sub-section In-place upgrade parameters.

4. Complete the installation process:
 - a. Select and accept the license agreement for your region of the world.
 - b. Select the **Backup-and-restore** option.
 - c. Follow the prompts, selecting the **Automated backup and non-database gathering** option when it becomes available to you.

 **Important!**

When you are prompted to choose a performance configuration, you can choose Production or Restore. If you choose Production, the installer runs `cov-admin-db tune`. This can improve Coverity Connect performance by re-tuning database performance. If you choose Restore, the installer does not run `cov-admin-db tune`. For details about `cov-admin-db tune`, see the *Coverity 2020.12 Command Reference*.

5. Copy modifications from your old to your new `server.xml` file.

If you made any modifications to the `<install_dir_cp>/server/base/conf/server.xml` file of your existing installation (for example, if you modified the `keystoreFile` or `keystorePass` properties), copy those modifications to your new installation.

 **Note**

Copy only the modifications; do not overwrite the entire file.

2.1.2.4. Performing an Intermachine upgrade (standalone instances)

This section describes how to perform an Intermachine upgrade on a Coverity Connect standalone instance that uses an embedded database.

An Intermachine upgrade is required if you want to relocate an existing Coverity Connect instance to a new host machine when you upgrade. Otherwise, perform a Backup-and-restore upgrade or an In-place upgrade. For more information, see Section 2.1.1, “Choosing the type of upgrade”.

To perform an Intermachine upgrade, you will need to run the installer two times. The first time you run the installer, use the Upgrade Preparation option to get a backup of the database and a backup of the non-database state. The second time you run the installer, use the Intermachine Upgrade option to install the new instance using the backups from the Upgrade Preparation step. The procedure in this section describes all of the steps in this process.

To perform an Intermachine upgrade on a standalone instance that uses an embedded database:

 **Note**

On Linux systems, you must **not** be logged in as root to upgrade Coverity Connect.

1. Make sure you have full permissions (rwx) on the existing installation directory and the new installation directory.
2. Download the correct Coverity Platform installer file for your operating system. (The Coverity Platform installer also installs Coverity Connect.)
3. Once the Coverity Platform installer download is complete, run the installer program for your operating system.

For Windows, we recommend that you install Coverity Connect with Windows Administrator privileges (installing Coverity Connect as a service requires it). To do so, right click the installer and choose Run as Administrator.

To install, double-click the .exe program.

For Linux, run the installer script in a Bourne shell, for example:

```
> ./cov-platform-linux64-[version].sh
```

 **Note**

Depending on the size of your database, the upgrade process can take a long time. If you are performing this operation by way of an SSH terminal, we recommend you use a persistent terminal (such as Screen) in case your session is interrupted.

The installer uses a text-based console mode or a graphical mode. The installation choices for graphical and console modes are equivalent. To install using graphical mode on Linux, append the `-g` option to the command above.

Note that you can change from graphical to silent installer (command line) as described in Chapter 1.2. Coverity Platform installation modes in the Coverity Installation and Deployment Guide. For details about the silent installer options and parameters, see 1.2.3. Coverity Connect silent installer, and in particular see the sub-sections "Upgrade preparation parameters" and "Intermachine upgrade parameters".

4. To update your existing Coverity Connect instance, complete the Upgrade Preparation process:

 **Note**

This part of the process shuts down your Coverity Connect server. Plan a maintenance window that covers the remainder of the upgrade process.

- a. Select the **Upgrade Preparation** option.
 - b. Specify the current Coverity Connect installation directory, and select one destination directory for both the database backup and the non-database state backup. The backups in this directory will be used to complete the upgrade on the destination machine.
5. Complete the Intermachine Upgrade process:
 - a. Make sure that the destination machine has access to the backups from the previous step. You could do that by sharing the directory containing your backups or by copying it over to the destination machine.
 - b. On the destination machine, launch the installer and follow the on-screen prompts, selecting the **Intermachine Upgrade** option.
 - c. Enter your desired installation directory, then specify the location of the directory where you saved the database backup file and non-database state backup (as part of the Upgrade Preparation step)
 - d. Follow the on-screen prompts to complete the upgrade.

2.1.3. Performing the upgrade: clustered deployments

2.1.3.1. Important information for clustered deployments

Before upgrading a Coverity Connect instance in a clustered deployment, note the following:

- **Upgrade the coordinator first.** You must upgrade the coordinator first and make sure that it is running properly before you upgrade any subscriber. After the coordinator is successfully upgraded, you can upgrade its subscribers in any order.
- **To upgrade the deployment by more than one major version (for example 8.6 to 2017.07), and at the same time keep the deployment in active use (except for downtime off-hours to perform**

the upgrade), stagger the coordinator upgrade with the subscriber upgrades. When upgrading an active deployment, the Coverity Connect version on the coordinator cannot be allowed to get two major versions ahead of its subscribers. However, this is only applicable when the upgrade is planned to occur while the Coverity Connect deployment is in active use (except for a temporary suspension of activity across the deployment's user base while the upgrade is performed). If the deployment is in active use and the upgrade spans two major versions, upgrade the coordinator to the next major version, then upgrade the subscribers to that version, then upgrade the coordinator to the second major version.

To determine whether you are upgrading by more than one major version, refer to Table 1.5.1 "Upgrade considerations" in the Coverity Upgrade and Release Notes. In the x.y.z numbering, major versions have differing x.y from each other.

In the yyyy.mm version numbering, different major versions have different yyyy.mm.

- **Staggering coordinator and subscriber upgrades is not required for inactive deployments.** When upgrading an inactive deployment, you can upgrade the coordinator several versions and then upgrade each subscriber several versions.
- **Upgrade instances, including their databases, independently of each other.** It is important for each Coverity Connect instance to upgrade its own PostgreSQL database. For example, you should not upgrade the database for Subscriber1 and then attempt to apply that database to other subscribers within the cluster.
- **Re-form trust relationships after upgrading.** Coverity Connect uses SSL to authenticate and encrypt communication between the Coordinator and Subscribers. The trust relationships define which Coverity Connect instances can participate in the cluster. If you make a copy of an instance, using an Intermachine or Backup-and-restore upgrade, both the copy and the original will be able to participate in the cluster, but the original should *not* be allowed to participate in the cluster. Having that duplication can damage the cluster. You must prevent that from happening by re-forming the trust relationships to exclude the original instance. For details, see "Post-upgrade manual setup" in 3.5.1.2.2 Setting up SSL certificates and TrustStores in the Coverity Platform User and Administrator Guide.

2.1.3.2. Upgrade prerequisites

Make sure the instance you want to upgrade meets the following prerequisites:

- **Important:** Make sure you are not running anti-virus software on a system with a Coverity Connect database because it impacts performance and can even interfere with correct functioning of the database, possibly including data corruption. If you must run anti-virus software, disable it for the duration of the upgrade or exempt the <cc_install_dir>/<database> directory from anti-virus inspection.
- Make sure there is sufficient free space in these locations:
 - The location for the backup if one is needed
 - The volume where the new instance will be installed (if you are not doing an In-place upgrade)

The free space should be roughly 3x the size of the database, as determined by the size of the <cc_install_dir>/<database> directory's contents.

- Make sure you have enough disk space for the upgrade.
- Make sure that you have the following information prior to launching the installer:
 - Existing Coverity Connect installation directory
 - Destination installation directory (if you plan to perform a Backup-and-restore or Intermachine upgrade)
 - Desired backup directory location (if you plan to create a database backup)
 - Location of your Coverity Connect license file
 - Desired ports for Coverity Connect communications (for Backup-and-restore and Intermachine upgrades)

2.1.3.3. Performing an In-place upgrade (clustered instances)

This section describes how to perform an In-place upgrade on Coverity Connect clustered instances (Coordinators and Subscribers) that use an embedded database.

To perform an In-place upgrade on a clustered instance that uses an embedded database:

1. Upgrade the instance using the installer. For details, see Section 2.1.2.2, “Performing an In-place upgrade (standalone instances)”.
2. (Perform this step on subscriber instances only) If the address of the Coordinator has changed as part of the upgrade, specify the new address of the Coordinator by editing the `remoteconfig.coordinator` property in the `cim.properties` file in the new installation directory.

2.1.3.4. Performing a Backup-and-restore upgrade (clustered instances)

This section describes how to perform a Backup-and-restore upgrade on Coverity Connect clustered instances (Coordinators and Subscribers) that use an embedded database.

Note

After using the installer to perform a Backup-and-restore upgrade on an instance, you need to update SSL certificates and Trust Stores. For details, see 3.5.1.2.2 Setting up SSL certificates and TrustStores in the Coverity Platform User and Administrator Guide.

To perform a Backup-and-restore upgrade on a clustered instance that uses an embedded database:

1. Upgrade the instance using the installer. For details, see Section 2.1.2.3, “Performing a Backup-and-restore upgrade (standalone instances)”.
2. (Perform this step on subscriber instances only) If the address of the Coordinator has changed as part of the upgrade, specify the new address of the Coordinator by editing the `remoteconfig.coordinator` property in the `cim.properties` file in the new installation directory.

3. Update SSL certificates and Trust Stores as needed. For more information, see 3.5.1.2.2 Setting up SSL certificates and TrustStores in the Coverity Platform User and Administrator Guide.

2.1.3.5. Performing an Intermachine upgrade (clustered instances)

This section describes how to perform an Intermachine upgrade on Coverity Connect clustered instances (Coordinators and Subscribers) that use an embedded database.

Note

After using the installer to perform an Intermachine upgrade on an instance, you need to update SSL certificates and Trust Stores. For details, see 3.5.1.2.2 Setting up SSL certificates and TrustStores in the Coverity Platform User and Administrator Guide.

To perform an Intermachine upgrade on a clustered instance that uses an embedded database:

1. Upgrade the instance using the installer. For details, see Section 2.1.2.4, “Performing an Intermachine upgrade (standalone instances)”.
2. (Perform this step on subscriber instances only) If the address of the Coordinator has changed as part of the upgrade, specify the new address of the Coordinator by editing the `remoteconfig.coordinator` property in the `cim.properties` file in the new installation directory.
3. Update SSL certificates and Trust Stores as needed. For more information, see 3.5.1.2.2 Setting up SSL certificates and TrustStores in the Coverity Platform User and Administrator Guide.

2.2. Upgrading instances that use an external database

This section describes procedures for upgrading a Coverity Connect instance that uses an external database. If you are using an embedded database with Coverity Connect, see Section 2.1, “Upgrading instances that use an embedded database”.

2.2.1. Upgrading in place on an instance that uses an external database

In addition to creating a separate Coverity Connect instance, this procedure requires you to upgrade your external database before running the installer to complete the upgrade.

End-of-Life notice: Support for PostgreSQL 9.3.x and 9.4.x is removed as of the Coverity 2017.07 release.

To perform an in-place upgrade on an instance that uses an external database:

1. If your database predates PostgreSQL 9.5.0, perform the following steps:
 - a. Either put Coverity Connect into maintenance mode or stop Coverity Connect completely.
To put Coverity Connect into maintenance mode use the `cov-im-ctl maintenance` command as follows:

For Linux:

```
> <OLD_cc_install_dir>/bin/cov-im-ctl maintenance
```

For Windows:

```
> <OLD_cc_install_dir>\bin\cov-im-ctl.exe maintenance
```

To stop Coverity Connect:

For Linux:

```
> <OLD_cc_install_dir>/bin/cov-im-ctl stop
```

For Windows:

```
> <OLD_cc_install_dir>\bin\cov-im-ctl.exe stop
```

- b. Upgrade your external database manually.

PostgreSQL 10.12 is preferred. The minimum requirement is PostgreSQL 9.5.0.

For more information, see the documentation at <http://www.postgresql.org/docs/9.5/static/upgrading.html> .

2. Download and run the Coverity Platform installer for your operating system to install the upgraded version of Coverity Connect in the same location as the previous version.

 **Important!**

When running the installer, it is important to select the new, upgraded version of your external PostgreSQL database.

For more complete details, see the installation instructions in the *Coverity 2020.12 Installation and Deployment Guide* .

2.2.2. Upgrading with backup-and-restore on an instance that uses an external database

This procedure requires you to create a new external database and to manually restore your old external database into it. You will also upgrade the database schema manually.

1. If you are backing up and restoring a version of Coverity Connect that has configured Coverity Connect email or set up an LDAP or Jira integration, you should save a copy of the `cim.ldap.key` value found in the `cim.properties` file now.

Later in this procedure, you will supply this value to the new Coverity Connect instance you create in order to preserve any email, Jira, or LDAP passwords that were set up in the old Coverity Connect instance. Note that if this value is not available for some reason, you can simply provide the passwords to the new Coverity Connect instance after completing the upgrade.

2. Use the following steps to back up your external database in preparation for the upgrade:

- a. Put your existing Coverity Connect instance into maintenance mode using the `cov-im-ctl maintenance` command.

For Linux:

```
> <OLD_cc_install_dir>/bin/cov-im-ctl maintenance
```

For Windows:

```
> <OLD_cc_install_dir>\bin\cov-im-ctl.exe maintenance
```

- b. Using your preferred method, create a backup of the external database that you are using with Coverity Connect.

You will use the backup file later in the upgrade procedure.

- c. Stop Coverity Connect and (optionally) verify the status:

For Linux:

```
> <OLD_cc_install_dir>/bin/cov-im-ctl stop
```

```
> <OLD_cc_install_dir>/bin/cov-im-ctl status
```

For Windows:

```
> <OLD_cc_install_dir>\bin\cov-im-ctl.exe stop
```

```
> <OLD_cc_install_dir>\bin\cov-im-ctl.exe status
```

- d. Stop the external database that was running with Coverity Connect.
- e. (Windows only)

If you are running Coverity Connect as a service, you must remove the service from the `<OLD_cc_install_dir>\bin` directory. To do so, run the following command as Administrator:

```
> sc delete cov-im-service
```

3. Create a *new* PostgreSQL database into which you will later restore the backup.

Coverity recommends that you create this database using a PostgreSQL 10.12 format. The minimum requirement is PostgreSQL 9.5.0.

 **Note**

It is required that you create the database with the following encoding settings:

```
--encoding UTF8 --locale C
```

These settings might not be available on some PostgreSQL installations, depending on the version, the operating system, and the options used when initially creating the database cluster. For more information, consult your PostgreSQL documentation.

If you get an error such as the following when using the required encoding settings, try adding `-T template0` to the command line.

```
=====  
Command used was  
  
createdb covtmp_703 -encoding UTF8 --locale=C --owner=coverity  
  
createdb: database creation failed: ERROR: new collation (C) is incompatible  
with the collation of the template database (en_US.UTF-8)  
=====
```

For more information about creating an external database, see *Using an external PostgreSQL database with Coverity Connect* in the *Coverity 2020.12 Installation and Deployment Guide* [↗](#).

4. Download and run the Coverity Platform installer for your operating system to create a new instance of Coverity Connect that uses your newly created external database.

 **Important!**

When running the installer, select your newly created external database, not the old external database. Later in this upgrade procedure, you will restore your backup into this empty database.

5. Use the following steps to restore the backup of your old database into the new database.

- a. Stop your *new* database.
- b. Restore the previously created backup to the new Postgres format using a tool such as `psql` or `pg_upgrade`. For more information, see the documentation at <http://www.postgresql.org/docs/9.5/static/upgrading.html> [↗](#).

For compatibility settings, see:

<http://www.postgresql.org/docs/9.5/static/runtime-config-compatible.html> [↗](#).

- c. Restart the new database.

It now contains the data from your old database.

6. Use the following steps to upgrade your database schema.

- a. Put your new Coverity Connect instance into maintenance mode.

For Linux:

```
> <NEW_cc_install_dir>/bin/cov-im-ctl maintenance>
```

For Windows:

```
> <NEW_cc_install_dir>\bin\cov-im-ctl.exe maintenance>
```

- b. Upgrade the PostgreSQL schema with the `cov-admin-db upgrade-schema` command.

For Linux:

```
> NEW_cc_install_dir/bin/cov-admin-db upgrade-schema
```

For Windows:

```
> NEW_cc_install_dir\bin\cov-admin-db.exe upgrade-schema
```

For more information about this command option, see the *Coverity 2020.12 Command Reference*.

7. If you saved a copy of the `cim.ldap.key` value in the first step, copy it over to the `cim.properties` file for your new Coverity Connect instance.

The value you copied over should replace the `cim.ldap.key` value that was automatically generated when you installed the new Coverity Connect instance.

This step will prevent the need to reset passwords for your Coverity Connect email, Jira plugin, and/or LDAP server functionality through the Coverity Connect configuration screens after you restart your new Coverity Connect instance.

8. Restart the new Coverity Connect instance:

For Linux:

```
> NEW_cc_install_dir/bin/cov-im-ctl start
```

For Windows:

```
> NEW_cc_install_dir\bin\cov-im-ctl.exe start
```

9. After integrating the upgraded Coverity Connect into your production environment, you can remove or archive the old instance.

 **Note**

The backup-and-restore upgrade does not transfer certificates that are associated with SSL, settings related to SSL, or custom settings that you might have made to certain files in your installation directory. If you need to perform a backup-and-restore upgrade, you can contact software-integrity-support@synopsys.com.

2.3. Coverity Connect upgrade environment variable parameters

Defect event processing

If you have a large number of events per defect (over 250) or are performing upgrade on a machine with limited memory it is recommended that you use the defaults or use slightly lower values (.5x) for these variables, particularly for DB_UPGRADE_EVENTS_LATEST_BATCH. If you have a large number of defects (and moderate numbers of events per defects) and are sensitive to upgrade speed, you can use a slightly higher value (2-4x) for DB_UPGRADE_EVENTS_NON_LATEST_BATCH.

Table 2.2. Defect event processing

Property	Description
DB_UPGRADE_EVENTS_LATEST_BATCH	<p>The default is 1000, however the recommended value is 32000 (with 32GB+ RAM).</p> <p>Controls upgrade batch processing of latest defects lower values allow the upgrade to succeed on machines with lower RAM thresholds at the expense of speed.</p> <p>This parameter is needed only when upgrading a database from version 6.0.3 or earlier.</p>
DB_UPGRADE_EVENTS_NON_LATEST_BATCH	<p>The default is 5000. The recommended value is 2000000 (with 32GB+ RAM).</p> <p>Controls upgrade batch processing of no-latest defects lower values allow the upgrade to succeed on machines with lower RAM thresholds at the expense of speed. The settings for these values are conservative. For machines with 32GB+ RAM, the values should be set substantially higher to reduce the upgrade time by ~75%.</p> <p>This parameter is needed only when upgrading a database from version 6.0.3 or earlier.</p>

COVERITY_RESTORE_JOBS

This option is only available with version 6.5.1 and later. It controls the level of parallelism in the `pg_restore` command. The default value is the number of processor cores. Alternatively, you can pass in the `-j` parameter to `cov-admin-db restore`.

2.4. Upgrading Coverity Desktop

This section describes the basic upgrade procedures for Coverity Desktop for Eclipse and Coverity Desktop for Microsoft Visual Studio. Prior to upgrading to the latest version, consult Chapter 5, *Important*

upgrade considerations for a list of important product changes that may affect your Coverity Desktop workflow.

For general installation procedures, see the Coverity Installation and Deployment Guide.

 **Note**

The 'Classic' variant of the Coverity Desktop plug-ins is no longer supported. If you are upgrading from Coverity Desktop Classic, you must uninstall it completely from your IDE, and then install the current Coverity Desktop version.

2.4.1. Upgrading Coverity Desktop for Eclipse and Wind River through the update site.

After Coverity Desktop is installed, you can check the update site to see if there are any updates to the plug-in. Select Help → Check for Updates, and the IDE will alert you if there are updates available, and give you the choice to install (upgrade) them. Alternatively, you can configure the IDE to automatically check for updates.

 **Note**

As with the *Install New Software...* mechanism for Wind River Workbench, you must go to the *Device Debug* perspective to make sure that it is present in the Help menu.

2.4.2. Upgrading Coverity Desktop for Visual Studio

Installing the latest version will upgrade automatically from versions 6.0.2 and later. For older versions choose `Uninstall` from the control panel before installing the newer version of the plug-in.

If you used the Gallery option to install Coverity Desktop, after you configure the new version's gallery you will have to run the update manually. To do so, navigate to Tools → Extensions and Updates → Updates and select the name you used for the gallery. Click **Update**, then restart when prompted to complete the upgrade.

See the *Coverity 2020.12 Installation and Deployment Guide* for more information on installing Coverity Desktop using a gallery.

2.4.3. Upgrading Coverity Desktop for IntelliJ and Android Studio

After Coverity Desktop is installed, you can check the update site to see if there are any updates to the plug-in. Select Help → Check for Update, and the IDE will alert you if there are updates available, and give you the choice to install (upgrade) them. Alternatively, you can configure the IDE to automatically check for updates.

2.5. Upgrading the Coverity Jenkins plug-in

For the Coverity Jenkins plug-in, follow the standard process for upgrading any Jenkins plug-in. Jenkins plug-in versions are not tied to other Coverity product versions. Although coordinating with your Coverity Platform upgrade is usually not necessary, we normally recommend using the latest version of the Jenkins plug-in.

2.6. Migrating and upgrading to Coverity Connect (Integrity Manager) from Defect Manager version 4.5.1

If you are upgrading Defect Manager from version 4.5.1, you must first download the 5.5.3 release of Integrity Manager and migrate version 4.5.1 to it. Instructions for completing this task are in the *5.5.3 Integrity Manager Migration Guide*, and other important information is in the *5.5.3 Release Notes*. After migrating, you need to upgrade Integrity Manager from version 5.5.3 to the latest Coverity Connect version.

If you have an active license for version 5.5, you can download the documentation set from the Coverity Community Portal [🔗](#).

2.7. Rolling back from Backup-and-Restore or Intermachine Upgrades to the previous Coverity Platform version

After an upgrade is complete, or if an upgrade fails, you can roll back to the previous version of Coverity Platform.

 **Note**

If the same port values were used for both the old instance and the new instance of Coverity Connect, only one instance can be running at a time.

 **Note**

When the Coverity Platform installer attempts to back up an instance of Coverity Connect, the installer may display an error message of "Starting the database failed" if a different instance that uses the same ports is currently running.

 **Note**

It is assumed that you have not put the new instance into production yet. If you have already begun using the new instance, then rolling back will mean using Coverity Connect from the point in time when you stopped the old instance and thus any commits or changes made to the new instance will not have been applied to the old instance.

If the old instance has not been removed, since the new Coverity Platform instance was installed in a new location, the old instance can still be used. Stop the new instance by running `cov-stop-im`, restart the old instance by running `cov-start-im`, and you will have rolled back.

If the old instance has been removed but you have already used the Upgrade Preparation installer type to back up the old instance's internal database, rolling back can be achieved by reinstalling the previous version with the Intermachine Upgrade installer type. First, stop the new instance by running `cov-stop-im`. Then, follow the same process as an Intermachine Upgrade, except:

- Use the installer for the previous version, instead of the installer for the new version.
- Make sure that the database backup file was created by the previous Coverity Connect version, not by the new version.

- It is recommended that you use a brand new installation directory that had not been used by a previous instance.

Chapter 3. Upgrading Coverity Analysis

Table of Contents

3.1. Standard Upgrade Procedure	23
3.2. Understanding churn	24
3.3. Preview and production checkers	25
3.4. Upgrading and MISRA changes	25
3.5. New behavior when emitting web application archives	26

There are now two ways to upgrade from a previous release of Coverity Analysis:

- Follow the standard upgrade procedure (detailed in Section 3.1, “Standard Upgrade Procedure”)
- Install an incremental release that contains the upgrade (detailed in Chapter 6, *Updating Coverity Analysis with Incremental Releases*)

You must install the Coverity Analysis 2020.12 version in its own location, rather than overwriting your existing installation.

Unless otherwise specified, the steps apply to Coverity Analysis for all supported languages.

 **Note**

Prior to upgrading to the latest version, see Chapter 5, *Important upgrade considerations* for a list of important product changes that might affect your Coverity Analysis workflow.

3.1. Standard Upgrade Procedure

To upgrade Coverity Analysis:

1. Install Coverity Analysis 2020.12 into a new directory.
Do not upgrade over an existing installation directory.
2. Recreate a configuration for each compiler and programming language of the source code you intend to analyze.

 **Note**

Important! Coverity recommends that you run the following command for Java:

```
> cov-configure --java
```

3. Move your manual configuration changes (if any) from the configuration files in your previous version of Coverity Analysis to the configuration files for 2020.12.

Additionally, for C/C++, move your customization changes (if any) in the `<install_dir>/config/user_nodefs.h` file for a previous version, to the `user_nodefs.h` file for 2020.12.

 **Note**

Do not move `.xmlldb` files (located in `<install_dir>/config`) that were generated by the previous version of Coverity Analysis. You will instead regenerate them for the latest version of Coverity Analysis in the next step.

4. Compile your custom models (if any) from the previous version with the `cov-make-library` command for the latest version.
5. Delete all old intermediate directories.
6. Update your existing scripts and paths to point to the new Coverity Analysis location.

3.2. Understanding churn

Churn is a measure of change in defect reporting between two sequential Coverity Analysis Feature releases (eg 2017.07 and 2018.01).

For most programming languages, churn can be discovered through a two-step process:

1. Using the two Coverity Analysis releases to separately analyze the same code base.
2. Using Coverity Connect to triage the resulting CIDs into the following classifications: False Positive and Bug

Churn does not apply to the CIDs that have Pending or Intentional classifications.

Churn formula:

```
(New False Positives + Lost True Positives)/Total Number of CIDs
```

 **Note**

Coverity expects the churn between contiguous Coverity Analysis releases to be less than 5% for the following programming languages:

- C/C++
 - C#
 - Java
 - JavaScript
- **New False Positives (FPs):** The number of CIDs that are reported by the more recent version of Coverity Analysis and that developers marked with an FP classification in Coverity Connect.
 - **Lost True Positives (TPs):** The number of CIDs that are reported by the earlier version of Coverity Analysis and that developers marked with the Bug classification in Coverity Connect.

- Total Number of CIDs: The number of CIDs produced by the separate analyses. This total includes CIDs that are common to each version of Coverity Analysis and any unique CIDs from either version. Common CIDs are the result of defect merging in Coverity Connect.

3.3. Preview and production checkers

In releases before 2018.06, we distinguished between *preview* and *production* checkers to signal a difference in the level of tuning that had been done for each checker: preview checkers were more likely to produce false negatives/positives than production checkers.

With the current release, we have removed this distinction and improved the tuning on nearly all the checkers that were deemed “preview.” These checkers are now enabled by default.

The following checkers remain *disabled* by default: if no language is specified in the list below, the checker is disabled for all languages; if a language is specified, it is only disabled for that language.

- ATOMICITY (Java)
- COM.ADDROF_LEAK
- COM.BSTR.ALLOC
- COM.BSTR.BAD_COMPARE
- COM.BSTR.NE_NON_BSTR
- INTEGER_OVERFLOW
- LOCK_INVERSION (C#)
- MIXED_ENUMS
- RISKY_CRYPT0 (C/C++/Objective-C/Objective-C++ only)
- USE_AFTER_FREE (Java)

If you previously ran `cov-analyze` with the `--all` or `--preview` command line option, you can enable the same set of checkers using the `-en` option on the `cov-analyze` command line (instead of using `--all` or `--preview`).

```
-en <CHECKER_NAME>
```

Note

Note: The `--preview` option has been deprecated.

3.4. Upgrading and MISRA changes

In version 2017.07, we introduced a completely re-written MISRA engine that vastly improved the coverage and quality of MISRA findings. This change might affect the number of defect reports – both

overall and with specific MISRA checkers. This section summarizes these changes and provides some upgrade recommendations.

The new MISRA engine allows us to cover every single rule that can be checked with static analysis. The quality of findings for many rules has also greatly improved. What you are likely to see as a result is an increase in issues that are true positives. The following recommendations might help you manage these changes:

- Make sure you upgrade at an appropriate point in your project.
- After the upgrade, migrate the triage data to the new Coverity instance.

A Python script is provided to migrate triage information for defects found by MISRA checkers in versions of Coverity Analysis prior to the 2017.07 version. You can find the script and its documentation in the Customer Portal.

- Investigate all findings to see where new defects are coming from. Are these all true positives?

For example, one common source of defects is the inclusion of system header files. Rule 2.5 specifies that a project should not contain unused macro declarations. It's possible these defects are found in system headers or other third party code. If that is the case, you can exclude that source from the scan.

- Finally, if you are overwhelmed with defects you do not have time to triage, keep the code base on Coverity version 8.7. Do this only if you have exhausted all other options. You can keep the analysis engine at this version while you continue to upgrade the Coverity server to the latest version. Keep in mind however that to get all the improved benefits of MISRA and other checkers, you should scan all new projects with the latest version of the analysis engine.

For additional details about these MISRA changes since version 2017.07, please consult the *Release Notes Archive*.

3.5. New behavior when emitting web application archives

Emitting a web application archive (.WAR file, .EAR file, or equivalent unpacked directory) with `cov-emit-java --war`, `--ear`, or similar now emits JavaScript code in that webapp archive. Users who emit such files might see an increase in lines of code analyzed, analysis runtime, and defects reported. You can disable emission of JavaScript code by specifying the `--skip-emit-war-javascript-source` option with the `cov-emit-java` command.

Chapter 4. Test Advisor upgrade impact

Table of Contents

4.1. False negative violations for impact history	27
4.2. Regenerating test metrics for Test Prioritization	28
4.3. Upgrading Function Coverage Instrumentation and Runtime Library	29
4.4. Test Advisor with Test Advisor QA Edition for C/C++	29

The following sections highlight several suggested procedures when upgrading to a new version of Test Advisor.

4.1. False negative violations for impact history

When upgrading Test Advisor from a previous release, the impact history is migrated to the latest format. However, some false negative violations are possible for impact-related defects unless you execute some steps before you upgrade. This section describes these procedures. Note that these procedures are only relevant if your Test Advisor policy contains impact history filters such as `recently_impacted`. For more information about the impact filters, see the *Test Advisor 2020.12 User and Administrator Guide*.

The high-level steps are as follows:

1. Before upgrade, make one last Test Advisor run.

This includes performing an analysis run (including a build, download, analyze, and commit of the current source code), as described in the *Test Advisor 2020.12 User and Administrator Guide*.

Make a note of the source timestamp. For the step where you run `cov-analyze`, you should pass the timestamp as an argument. For example:

```
cov-analyze --code-version-date '2014-01-01 12:00' ...
```

2. Upgrade your Coverity Connect (Coverity Platform) and Test Advisor (Coverity Analysis) installations.
3. For the first analysis run after the upgrade, you must use the exact same version of the source code that you did just before the upgrade (as described in step 1).

Again, use `--code-coverage-date` to pass the timestamp to `cov-analyze`. This allows migration of the impact history with no loss of fidelity.

4. Execute all subsequent Test Advisor analysis runs as normal.

The consequences of ignoring these steps are most likely small. The false negatives arise from source functions that have changed between the last analysis run before upgrade and the first analysis run after the upgrade. During the first run after upgrade, all source changes are ignored when calculating impact.

The following example shows a possible false negative:

```
void foo(int x) {
    if (x == 1) {           // covered
        noImpact();        // covered
    } else {               // not covered
        hasImpact();       // not covered
    }
}
```

Assume that tests call `foo(1)` such that only the first branch is covered. Next, assume that the policy file requires coverage on all lines that call recently-impacted functions. If the `hasImpact()` function is changed so that its impact is different between the last analysis run before upgrade and the first analysis run after upgrade, then no violation is reported even though this line is not covered.

Note that non-impact violations will still be reported as normal. If the policy requires 80% coverage for every function, then the function above would have a violation because it is only 50% covered. The upgrade makes no difference for this kind of violation.

If it is not possible to follow this recommended upgrade procedure, you can minimize the number of false negatives by upgrading when few changes are made to the code (such as in the early morning or on a weekend). Make the first analysis run as soon as possible after upgrade. The upgrade procedure is especially recommended if you do not make Test Advisor analyses very often (such as weekly), or if you strongly care about seeing all possible impact violations.

To fix false negatives

If you have upgraded without following the procedure above and you want to remove the false negative impact violations, use the following steps:

1. Use your source control management (SCM) to check out a version of your code with the exact timestamp as that used in the last analysis run before the upgrade. Using the upgraded Test Advisor, make an analysis run (build, download, analyze, commit) on this code.

You should pass the timestamp as an argument to `cov-analyze`. For example:

```
cov-analyze --code-version-date '2014-01-01 12:00' ..."
```

2. Run the analysis procedures for the most recent code as normal. The impact history is fixed and there are no false negative impact defects.

4.2. Regenerating test metrics for Test Prioritization

When upgrading to a newer version of Test Advisor, it is recommended that Test Prioritization users regenerate test metrics (using `cov-analyze --enable-test-metrics`) rather than continue to use the test metrics generated by a previous version of Test Advisor. While reusing older versions of test metrics is expected to still work correctly, doing so may cause new features in the upgraded version of Test Advisor to be disabled.

It is generally recommended to regenerate test metrics on a regular basis. In this case, the Test Advisor upgrade should be performed immediately before the regularly-scheduled regeneration of test metrics.

4.3. Upgrading Function Coverage Instrumentation and Runtime Library

To upgrade Function Coverage Instrumentation, complete the following steps:

1. Upgrade your Coverity Runtime Library, following the instructions in Part 4 of the *Coverity Extend SDK 2020.12 Checker Development Guide*. Any customization of the Runtime Library should be applied to the new version before redeployment.
2. Regenerate your compiler configurations, as described in the *Coverity Analysis 2020.12 User and Administrator Guide*, "Configuring compilers for Coverity Analysis".
3. Rebuild your instrumented programs, as described in the *Test Advisor 2020.12 User and Administrator Guide*, "Coverity Function Coverage Instrumentation".

4.4. Test Advisor with Test Advisor QA Edition for C/C++

When using Test Advisor QA Edition for C/C++ and upgrading to a newer version of Test Advisor, it is recommended that you take the following steps when you want to generate your first scan using the new Test Advisor version:

1. Identify the version of your source code that you last scanned and uploaded to the Cockpit with the previous Test Advisor version.
2. Using the new version of Test Advisor, build this same version of your source code using `cov-build`.
3. Generate a new scan file using `cov-manage-emit`.
4. Upload this new scan file to the Cockpit.

 **Note**

Note that you will need to specify a new version for this scan when uploading.

5. Now build the latest version of your source code using `cov-build`.
6. Generate a new scan file using `cov-manage-emit`.
7. Upload this new scan file to the Cockpit.

 **Note**

Here again you will need to specify a new version when uploading.

At this point you will be able to properly identify changes between the previous version of your source code (scan generated in steps 3-4), and the latest version (scan generated in steps 6-7). Failure to follow these steps will result in a large number of false changes when uploading your scan of the latest version of the source code.

It is important to note that comparing scans generated by different version of Test Advisor may result in many false changes being reported.

For more detailed information on the above commands, please refer to Part 4, "Using Test Advisor with Test Advisor QA Edition for C/C++" in the *Test Advisor 2020.12 User and Administrator Guide*.

Chapter 5. Important upgrade considerations

Table of Contents

5.1. Upgrade considerations for 2020.12	32
5.2. Upgrade considerations for 2020.09	32
5.3. Upgrade considerations for 2020.06	32
5.4. Upgrade considerations for 2020.03	32
5.5. Upgrade considerations for 2019.12	32
5.6. Upgrade considerations for 2019.09	34
5.7. Upgrade considerations for 2019.06	34
5.8. Upgrade considerations for 2019.03	34
5.9. Upgrade considerations for 2018.12	34
5.10. Upgrade considerations for 2018.09	35
5.11. Upgrade considerations for 2018.06	35
5.12. Coverity Connect Web Services API Release and Deprecation Schedule	36

This section provides information about features that might impact the workflow of your Coverity products after you upgrade your Coverity installation. This information will help you understand what features have changed (from release to release), plan for the changes after you upgrade, and locate the information in the documentation that describes the necessary steps to successfully incorporate the changes into your workflow.

In the following table, locate the "Upgrade considerations" link for both the release that you are upgrading to and the release you are upgrading from. Read the "Upgrade considerations" section for the release you are upgrading to and then read each subsequent section until you reach the section for the release you are upgrading from. For example, if you are upgrading from 2018.3 to 2018.12, you should read the "Upgrade considerations" sections for 2018.12, 2018.09, and 2018.06.

Table 5.1. Upgrade considerations per release

Section 5.1, "Upgrade considerations for 2020.12"
Section 5.2, "Upgrade considerations for 2020.09"
Section 5.3, "Upgrade considerations for 2020.06"
Section 5.4, "Upgrade considerations for 2020.03"
Section 5.5, "Upgrade considerations for 2019.12"
Section 5.6, "Upgrade considerations for 2019.09"
Section 5.7, "Upgrade considerations for 2019.06"
Section 5.8, "Upgrade considerations for 2019.03"
Section 5.9, "Upgrade considerations for 2018.12"

Section 5.10, "Upgrade considerations for 2018.09"

Section 5.11, "Upgrade considerations for 2018.06"

To learn about upgrade considerations since version 6.0, you can review the upgrade notes in the archived 6.6.2 version of this guide (available at Coverity Customer Center [link](#), requires login). You can contact software-integrity-support@synopsys.com for guidance.

5.1. Upgrade considerations for 2020.12

For information about deprecated and dropped support, other updates, known issues, and fixed bugs, see the [Coverity 2020.12 Release Notes](#).

5.2. Upgrade considerations for 2020.09

For information about deprecated and dropped support, other updates, known issues, and fixed bugs, see "Coverity 2020.09 Release Notes" (and the sections for associated hot fixes) in the [Coverity Release Notes Archive](#).

5.3. Upgrade considerations for 2020.06

For information about deprecated and dropped support, other updates, known issues, and fixed bugs, see "Coverity 2020.06 Release Notes" (and the sections for associated hot fixes) in the [Coverity Release Notes Archive](#).

5.4. Upgrade considerations for 2020.03

For information about deprecated and dropped support, other updates, known issues, and fixed bugs, see "Coverity 2020.03 Release Notes" (and the sections for associated hot fixes) in the [Coverity Release Notes Archive](#).

The following issues affect upgrading to 2020.03.

5.4.1. Coverity Connect

5.4.1.1.

The default value for the `cim.cleanup.stream.delay.min` property (specified in the `cim.properties` file) has changed from 30 to 2. If you have explicitly set this property to a significantly higher value than 2, and you delete large numbers of streams, we recommend that you set it to 2. For more information about this property, refer to the [Coverity 2020.03 Platform User and Administrator Guide](#).

5.5. Upgrade considerations for 2019.12

For information about deprecated and dropped support, other updates, known issues, and fixed bugs, see "Coverity 2019.12 Release Notes" (and the sections for associated hot fixes) in the *Coverity Release Notes Archive* [🔗](#).

The following issues affect upgrading to 2019.12.

5.5.1. Coverity Analysis

5.5.1.1. Analyzing translation units in `node_modules`

The behavior of `cov-analyze` has changed in that translation units in `node_modules` directories for JavaScript or TypeScript source are no longer analyzed unless the new `--analyze-node-modules` option is specified.

Even when using the `--tu` or `--tu-pattern` options, you must specify the `--analyze-node-modules` option in order to analyze translation units in `node_modules`.

5.5.1.2. Buildless capture

The `--dir` option to the `cov-capture` command was previously optional but is now required. This option specifies the location of the intermediate directory. You must ensure that all calls to `cov-capture` include this option.

The `--dot-coverity-location` option to the `cov-capture` command is no longer supported. If you currently use this option to specify a location for diagnostics and related data for `cov-capture`, you must remove it. The diagnostic directory is now always created in the intermediate directory and is called `cov-capture`.

5.5.1.3. Crossworks MSP430 support change

Affects: Customers who use Crossworks MSP430 compilers.

Changes:

- The compiler type `crossworks:430` has been converted to `crossworks:cc` for compiler driver `cc`.
- The compiler type `crossworks:hcc` has been added for compiler driver `hcc`.
- The compiler type `crossworks:hcl` has been added for compiler driver `hcl`.

Impact: Users must regenerate compiler configurations because of significant restructuring of how these compilers are handled.

Upgrade procedure:

Run `cov-configure` with the compiler type (option `--comptype`) set to one of:

- `crossworks:cc` for compiler driver `cc`

- `crossworks:hcc` for compiler driver hcc
- `crossworks:hcl` for compiler driver hcl

As always, a configuration template is recommended.

5.5.2. Coverity Desktop

5.5.2.1. Updating the IBM RTC plugin

The IBM RTC plugin has been replaced with the Eclipse plugin.

Starting in release 2019.09, when upgrading the IBM RTC plugin, you might encounter a screen that prompts you to evaluate the changes to be applied before continuing. The changes consist of the renaming of the plugin feature from `com.coverity.desktop.java.ibm.feature.feature.group` to `com.coverity.desktop.java.feature.feature.group`.

When the installer tells you "Your original request has been modified" and shows you in the **Details** pane that Coverity Desktop Java Analysis has already been installed, you should click **Next** and proceed with the installation.

5.6. Upgrade considerations for 2019.09

For information about deprecated and dropped support, other updates, known issues, and fixed bugs, see "Coverity 2019.09 Release Notes" (and the sections for associated hot fixes) in the *Coverity Release Notes Archive* [🔗](#).

5.7. Upgrade considerations for 2019.06

For information about deprecated and dropped support, other updates, known issues, and fixed bugs, see "Coverity 2019.06 Release Notes" (and the sections for associated hot fixes) in the *Coverity Release Notes Archive* [🔗](#).

5.8. Upgrade considerations for 2019.03

For information about deprecated and dropped support, other updates, known issues, and fixed bugs, see "Coverity 2019.03 Release Notes" (and the sections for associated hot fixes) in the *Coverity Release Notes Archive* [🔗](#).

5.9. Upgrade considerations for 2018.12

For information about deprecated and dropped support, other updates, known issues, and fixed bugs, see "Coverity 2018.12 Release Notes" (and the sections for associated hot fixes) in the *Coverity Release Notes Archive* [🔗](#).

The following issues affect upgrading to 2018.12.

5.9.1. Coverity Analysis

Users of Test Advisor and Dynamic Analysis who do test capture with `cov-capture` must change their command lines to use `cov-build --test-capture`. The tool `cov-capture` has been replaced in this release with a new tool that captures source code without wrapping a build.

5.10. Upgrade considerations for 2018.09

The following issues affect upgrading to 2018.09.

5.10.1. Deprecated notices

Support has been deprecated for products listed in "Important information for 2018.09" in the *Coverity Release Notes Archive* [🔗](#).

5.10.2. End-of-Life (EOL) notices

Support has been dropped for products listed in "Important information for 2018.09" in the *Coverity Release Notes Archive* [🔗](#).

5.10.3. Coverity Platform

See "Coverity Platform 2018.09" in the *Coverity Release Notes Archive* [🔗](#).

5.10.4. Coverity Analysis

See "Coverity Analysis 2018.09" in the *Coverity Release Notes Archive* [🔗](#).

5.10.5. Coverity Desktop

See "Coverity Desktop 2018.09" in the *Coverity Release Notes Archive* [🔗](#).

5.11. Upgrade considerations for 2018.06

See the Coverity 2018.06 release notes.

5.11.1. Deprecated notices

Support has been deprecated for products listed in "Important information for 2018.06" in the *Coverity Release Notes Archive* [🔗](#).

5.11.2. End-of-Life (EOL) notices

Support has been dropped for products listed in "Important information for 2018.06" in the *Coverity Release Notes Archive* [🔗](#).

5.11.3. Coverity Platform

5.11.4. Coverity Analysis

5.11.4.1. 115851 FlexNet libraries and lmgd v11.15.0+

We have upgraded the lmgd and lmutl libraries on all platforms. Customers are recommended to upgrade lmgd and lmutl to v11.15 in order to fix several security vulnerabilities. The lmgd and lmutl libraries can be found in the CoverityAnalysis bin folder.

5.11.5. Coverity Desktop

5.12. Coverity Connect Web Services API Release and Deprecation Schedule

The following table lists the release history and deprecation schedule for each version of the Coverity Connect Web Services API. Specifically, the table lists the following:

- WS - The Web Services API version.
- Released - The Coverity Connect (formerly Integrity Manager) release in which the version of the Web Services API was introduced.
- Date introduced - The date that the version of the API was initially released.
- Deprecated - The Coverity Connect version in which the Web Services API was deprecated. Deprecation denotes the last supported version.
- Dropped - The Coverity Connect release in which support for the Web Services API version was dropped.

Table 5.2. Web Services API release and deprecation schedule

WS	Release	Date introduced	Deprecated	Dropped
v1	5.2.1	06/2010	6.0.0	6.5.0
v2	5.3.0	12/2010	6.0.0	6.5.0
v3	5.4.1	06/2011	6.5.1	6.6.0
v4	5.5.0	10/2011	7.5.0	7.6.0
v5	6.0.0	04/2012	7.6.0	7.7.0
v6	6.5.0	09/2012	7.7.0	
v7	6.5.1 (modified in 6.6)	12/2012		
v8	7.0 (modified in 7.5.0)	12/2013		

Important upgrade considerations

WS	Release	Date introduced	Deprecated	Dropped
v9	7.6.0	12/2014		

This guide also provides version-specific release and upgrade notes related to the API for each release. For details about changes to the API from release to release, see the Coverity Platform Web Services API Reference. The API History and related logs in the reference document these changes.

Chapter 6. Updating Coverity Analysis with Incremental Releases

Table of Contents

6.1. Installation	38
6.2. How Updates Work	38
6.3. Running <code>cov-install-updates</code>	39
6.4. Rolling Back an Update Session	39
6.5. Other Utilities	39
6.6. Upgrading Coverity Analysis Using Updates	40

6.1. Installation

Coverity Analysis now includes a utility (`cov-install-updates`) for applying incremental updates. These updates include timely enhancements to `cov-analysis` as they become available. Since incremental updates include only the difference between the current and preceding versions, installer packages are expected to be smaller and installation times shorter compared to a full upgrade.

Incremental releases are intended to avoid changes that will introduce churn (Section 3.2, “Understanding churn”). However, critical bug fixes and security enhancements may still cause churn. It is recommended to review the incremental updates being offered (using the `list` subcommand) and avoid those that may disrupt your workflow.

Several updates may be applied in one session. This is done automatically, so a deployment can easily be brought up-to-date with the latest incremental release. Normally, updates apply only to the latest major release. However, it is possible to install upgrades as well as updates by specifying the desired target version explicitly.

6.2. How Updates Work

After a new (major) release of `cov-analysis`, incremental releases will be published when enhancements to that release become available. These enhancements can be viewed on the Customer Portal and are provided as part of our support for the Coverity Analysis product.

Once each day, each Coverity Connect instance contacts the Customer Portal and automatically downloads update packages, storing them locally. These updates are then available for downloading and installation on connected Coverity Analysis platforms.

When the results of an analysis run are uploaded to Coverity Connect using the `cov-commit-defects` command, the Coverity Connect instance will check if it has any updates that can be installed on that Coverity Analysis client. If so, the client is informed and the `cov-commit-defects` prints a message to that effect before exiting.

Since even small changes can be disruptive in certain development environments, we leave it up to the Coverity Analysis administrator when to install updates. You can choose the most appropriate time to run `cov-install-updates` to complete the update process.

6.3. Running `cov-install-updates`

This section provides an overview of the `cov-install-updates` command. For full details, see the `cov-install-updates` entry in the *Coverity 2020.12 Command Reference*.

The `cov-install-updates` utility is used to list and install incremental updates. It can also be used to check if updates are available and to roll back an undesired update.

To list the updates that are available for a given Coverity Analysis client, use the `cov-install-updates list` command on that client and specify the options necessary to connect to a Coverity Connect instance. Minimally, it is necessary to provide hostname and port information (using the `--host` and `--port` options) and login credentials. It is recommended to obtain an access token from the Coverity Connect instance, and use that (`--auth-key-file`) with your user name (`--user`), in preference to a plaintext password (`--password`).

To install the available updates, use the `cov-install-updates install` command, and again specify the required connection options.

6.4. Rolling Back an Update Session

Because `cov-install-updates` installs update packages transactionally, it is unlikely to leave the installation in an unusable state. On the other hand, it is possible that updates will affect your analysis results in ways that you find undesirable.

To handle this scenario, `cov-install-updates` provides a `rollback` subcommand. This command will roll back all of the updates added in the most recent update session.

 **Note**

An update session is one invocation of the `cov-install-updates install` command. Subsequent invocations overwrite the rollback data, so only data from the last session is retained.

Only files which were actually changed during the last installation session are rolled back. Other files which were modified after the update session will not be affected by a rollback.

6.5. Other Utilities

You can use the `cov-install-updates check` command to check if there are any updates available for a given client. This command requires connection options, so it can query a connected Coverity Connect instance for available update information. The `check` subcommand only looks for updates that have the same base version as the current installation. To see if there are upgrade installers available, use `cov-install-updates list --show=upgrades`.

For use in scripts, `cov-install-updates check` will return 0 if updates are available, and 1 if not. An error code of 2 indicates that the inputs were invalid.

You can use the `cov-install-updates version` to determine which version (including updates) is currently installed. Major versions have the format YYYY.MM; update versions are indicated by a hyphen

followed by a decimal number. Service pack release versions are indicated by a suffix `-SP1`, `-SP2`, etc. If present, the service pack number will precede any update release version number.

6.6. Upgrading Coverity Analysis Using Updates

Upgrading Coverity Analysis using incremental updates involves running the update utility and specifying a target version (`--end-version`) that contains or depends on an incremental installer containing an update.

As with the standard update procedure, Coverity recommends that you install the updated distribution in its own location. When using the incremental installer utility, this involves making a verbatim copy of the existing `cov-analysis` distribution in a new directory and applying the updates there.

Note

It is not necessary to wait until the availability of an update is indicated at the end of a commit. The listing and installation of updates can be performed at any time.

The update installer will only download and install updates that are appropriate for the client on which it is run.

To update Coverity Analysis:

1. Optionally, make a backup copy of the existing installation.

This step is not required, but is strongly recommended.

2. Run `cov-install-updates list <connection-options>` to obtain a list of available updates.

This command uses the version of the current installation to determine which update packages are appropriate.

If multiple updates are available, they are listed in the order in which they will be installed. If no updates are available, the list will be empty.

Normally, only updates with the same base version as the current installation will be listed. To see all available updates, including those that contain or require an upgrade to a newer major release version, use the `--show upgrades` option.

3. Run `cov-install-updates install <connection-options>` to install the available updates.

By default, all available updates having the same base version as the current installation will be installed. The `--end-version` option can be used to alter this behavior.

Selecting an end-version that is older than the latest available update will install updates up to and including the specified end-version. Newer updates will continue to be offered by the `list` subcommand and by `cov-commit-defects`.

Selecting an end version with a newer base version than the current installation will cause the installation to be upgraded as part of the update process.

 **Note**

Since there are dependencies between Coverity Analysis, Coverity Connect and the desktop plugins, it is still recommended that you follow the standard upgrade procedure Chapter 3, *Upgrading Coverity Analysis* to upgrade a Coverity Analysis installation. However, in some cases it may be more convenient to upgrade analysis clients using `cov-install-updates`.

4. Review the console output from `cov-install-updates` and ensure that the desired updates were installed. `cov-install-updates` is designed to install each update package in its entirety or not at all.

If the current installation has become corrupted, `cov-install-updates` will determine this before the installation of a given package begins. In that case, it will error out before attempting to install that package.

5. Continue with Step 2 of Chapter 3, *Upgrading Coverity Analysis*.

Appendix A. Coverity Legal Notice

Table of Contents

A.1. Legal Notice 42

A.1. Legal Notice

The information contained in this document, and the Licensed Product provided by Synopsys, are the proprietary and confidential information of Synopsys, Inc. and its affiliates and licensors, and are supplied subject to, and may be used only by Synopsys customers in accordance with the terms and conditions of a license agreement previously accepted by Synopsys and that customer. Synopsys' current standard end user license terms and conditions are contained in the `cov_EULM` files located at `<install_dir>/doc/en/licenses/end_user_license`.

Portions of the product described in this documentation use third-party material. Notices, terms and conditions, and copyrights regarding third party material may be found in the `<install_dir>/doc/en/licenses` directory.

Customer acknowledges that the use of Synopsys Licensed Products may be enabled by authorization keys supplied by Synopsys for a limited licensed period. At the end of this period, the authorization key will expire. You agree not to take any action to work around or override these license restrictions or use the Licensed Products beyond the licensed period. Any attempt to do so will be considered an infringement of intellectual property rights that may be subject to legal action.

If Synopsys has authorized you, either in this documentation or pursuant to a separate mutually accepted license agreement, to distribute Java source that contains Synopsys annotations, then your distribution should include Synopsys' `analysis_install_dir/library/annotations.jar` to ensure a clean compilation. This `annotations.jar` file contains proprietary intellectual property owned by Synopsys. Synopsys customers with a valid license to Synopsys' Licensed Products are permitted to distribute this JAR file with source that has been analyzed by Synopsys' Licensed Products consistent with the terms of such valid license issued by Synopsys. Any authorized distribution must include the following copyright notice: **Copyright © 2020 Synopsys, Inc. All rights reserved worldwide.**

U.S. GOVERNMENT RESTRICTED RIGHTS: The Software and associated documentation are provided with Restricted Rights. Use, duplication, or disclosure by the U.S. Government is subject to restrictions set forth in subparagraph (c)(1) of The Rights in Technical Data and Computer Software clause at DFARS 252.227-7013 or subparagraphs (c)(1) and (2) of Commercial Computer Software – Restricted Rights at 48 CFR 52.227-19, as applicable.

The Manufacturer is: Synopsys, Inc. 690 E. Middlefield Road, Mountain View, California 94043.

The Licensed Product known as Coverity is protected by multiple patents and patents pending, including U.S. Patent No. 7,340,726.

Trademark Statement

Coverity and the Coverity logo are trademarks or registered trademarks of Synopsys, Inc. in the U.S. and other countries. Synopsys' trademarks may be used publicly only with permission from

Synopsys. Fair use of Synopsys' trademarks in advertising and promotion of Synopsys' Licensed Products requires proper acknowledgement.

Microsoft, Visual Studio, and Visual C# are trademarks or registered trademarks of Microsoft Corporation in the United States and/or other countries.

Microsoft Research Detours Package, Version 3.0.

Copyright © Microsoft Corporation. All rights reserved.

Oracle and Java are registered trademarks of Oracle and/or affiliates. Other names may be trademarks of their respective owners.

"MISRA", "MISRA C" and the MISRA triangle logo are registered trademarks of MISRA Ltd, held on behalf of the MISRA Consortium. © MIRA Ltd, 1998 - 2013. All rights reserved. The name FindBugs and the FindBugs logo are trademarked by The University of Maryland.

Other names and brands may be claimed as the property of others.

This Licensed Product contains open source or community source software ("**Open Source Software**") provided under separate license terms (the "**Open Source License Terms**"), as described in the applicable license agreement under which this Licensed Product is licensed ("**Agreement**"). The applicable Open Source License Terms are identified in a directory named `licenses` provided with the delivery of this Licensed Product. For all Open Source Software subject to the terms of an LGPL license, Customer may contact Synopsys at `software-integrity-support@synopsys.com` and Synopsys will comply with the terms of the LGPL by delivering to Customer the applicable requested Open Source Software package, and any modifications to such Open Source Software package, in source format, under the applicable LGPL license. Any Open Source Software subject to the terms and conditions of the GPLv3 license as its Open Source License Terms that is provided with this Licensed Product is provided as a mere aggregation of GPL code with Synopsys' proprietary code, pursuant to Section 5 of GPLv3. Such Open Source Software is a self-contained program separate and apart from the Synopsys code that does not interact with the Synopsys proprietary code. Accordingly, the GPL code and the Synopsys proprietary code that make up this Licensed Product co-exist on the same media, but do not operate together. Customer may contact Synopsys at `software-integrity-support@synopsys.com` and Synopsys will comply with the terms of the GPL by delivering to Customer the applicable requested Open Source Software package in source code format, in accordance with the terms and conditions of the GPLv3 license. No Synopsys proprietary code that Synopsys chooses to provide to Customer will be provided in source code form; it will be provided in executable form only. Any Customer changes to the Licensed Product (including the Open Source Software) will void all Synopsys obligations under the Agreement, including but not limited to warranty, maintenance services and infringement indemnity obligations.

The Cobertura package, licensed under the GPLv2, has been modified as of release 7.0.3. The package is a self-contained program, separate and apart from Synopsys code that does not interact with the Synopsys proprietary code. The Cobertura package and the Synopsys proprietary code co-exist on the same media, but do not operate together. Customer may contact Synopsys at `software-integrity-support@synopsys.com` and Synopsys will comply with the terms of the GPL by delivering to Customer the applicable requested open source package in source format, under the GPLv2 license. Any Synopsys proprietary code that Synopsys chooses to provide to Customer upon its request will be provided in object form only. Any changes to the Licensed Product will void all

Coverity obligations under the Agreement, including but not limited to warranty, maintenance services and infringement indemnity obligations. If Customer does not have the modified Cobertura package, Synopsys recommends to use the JaCoCo package instead.

For information about using JaCoCo, see the description for `cov-build --java-coverage` in the *Command Reference*.

LLVM/Clang subproject

Copyright © All rights reserved. Developed by: LLVM Team, University of Illinois at Urbana-Champaign (<http://llvm.org/>). Permission is hereby granted, free of charge, to any person obtaining a copy of LLVM/Clang and associated documentation files ("Clang"), to deal with Clang without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of Clang, and to permit persons to whom Clang is furnished to do so, subject to the following conditions: Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimers. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimers in the documentation and/or other materials provided with the distribution. Neither the name of the University of Illinois at Urbana-Champaign, nor the names of its contributors may be used to endorse or promote products derived from Clang without specific prior written permission.

CLANG IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE CONTRIBUTORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH CLANG OR THE USE OR OTHER DEALINGS WITH CLANG.

Rackspac Threading Library (2.0)

Copyright © Rackspac, US Inc. All rights reserved. Licensed under the Apache License, Version 2.0 (the "License"); you may not use these files except in compliance with the License. You may obtain a copy of the License at <http://www.apache.org/licenses/LICENSE-2.0>.

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

SIL Open Font Library subproject

Copyright © 2020 Synopsys Inc. All rights reserved worldwide. (www.synopsys.com), with Reserved Font Name fa-gear, fa-info-circle, fa-question.

This Font Software is licensed under the SIL Open Font License, Version 1.1. This license is available with a FAQ at <http://scripts.sil.org/OFL>.

Apache Software License, Version 1.1

Copyright © 1999-2003 The Apache Software Foundation. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Coverity Legal Notice

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The end-user documentation included with the redistribution, if any, must include the following acknowledgement: "This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>)."

Alternately, this acknowledgement may appear in the software itself, if and wherever such third-party acknowledgements normally appear.

4. The names "The Jakarta Project", "Commons", and "Apache Software Foundation" must not be used to endorse or promote products derived from this software without prior written permission. For written permission, please contact apache@apache.org.
5. Products derived from this software may not be called "Apache" nor may "Apache" appear in their names without prior written permission of the Apache Group.

THIS SOFTWARE IS PROVIDED ``AS IS" AND ANY EXPRESSED OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE APACHE SOFTWARE FOUNDATION OR ITS CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Apache License Version 2.0, January 2004 <http://www.apache.org/licenses/>
Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at: <http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.

Results of analysis from Coverity and Test Advisor represent the results of analysis as of the date and time that the analysis was conducted. The results represent an assessment of the errors, weaknesses and vulnerabilities that can be detected by the analysis, and do not state or infer that no other errors, weaknesses or vulnerabilities exist in the software analyzed. Synopsys does NOT guarantee that all errors, weakness or vulnerabilities will be discovered or detected or that such errors, weaknesses or vulnerabilities are discoverable or detectable.

SYNOPSYS AND ITS SUPPLIERS DISCLAIM ALL WARRANTIES, CONDITIONS AND REPRESENTATIONS, EXPRESS, IMPLIED OR STATUTORY, INCLUDING THOSE RELATED

TO MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, SATISFACTORY QUALITY, ACCURACY OR COMPLETENESS OF RESULTS, CONFORMANCE WITH DESCRIPTION, AND NON-INFRINGEMENT. SYNOPSIS AND ITS SUPPLIERS SPECIFICALLY DISCLAIM ALL IMPLIED WARRANTIES, CONDITIONS AND REPRESENTATIONS ARISING OUT OF COURSE OF DEALING, USAGE OR TRADE.