Upgrading ConfD and the CDB Schema

tail-f a Cisco company

Table of Contents

Upgrading ConfD and the CDB Schema

1. Introduction

Software upgrade is a common practice used to change or update software in order to provide new functionality, bug fixes, or to upgrade to a newly supported version of the software.

Sometimes there is a distinction between a software update and a software upgrade, whereby a software update (or patch) deals with very small or minor changes that don't change or add functionality. This usually doesn't result in a different software version. Whereas a software upgrade is used to add significant or completely new changes to the software and a new version number is used.

In other cases, every software update is also an upgrade, bumping up the version number of the software. This is the case for ConfD. Bug fixes or minor changes are released in maintenance or patch releases, with a different version number.

ConfD upgrades are necessary in order to benefit from new functionality or to apply bug fixes. It is also necessary to upgrade in order to deploy an actively supported version of ConfD. The support and maintenance windows for any given version of ConfD are limited. When a ConfD major release (two number release number: X.Y) reaches EOM (End of Maintenance), the code for that branch is no longer maintained (i.e. no more bug fixes) and no further releases from this branch will be made. Note that EOS (End of Support) happens sometime after EOM. During the support-only period, questions about the release will still be made. The ConfD EOM and EOS schedule can be found on the Tail-f website at https://www.tail-f.com/confd-eos-eom/.

To ensure a successful upgrade of software and a smoother process for the end user, there are several details that should be considered, depending on the target of the upgrade: full system software image upgrade, ConfD daemon upgrade, database schema upgrade, or a mix of everything.

A ConfD daemon upgrade is an upgrade of the ConfD runtime software.

A database schema upgrade occurs when the YANG data models change as the YANG data models define the database schema. Technically, there is no distinction between an upgrade and a downgrade. To ConfD, it is simply a YANG data model change detected when ConfD is restarted or reloaded. This is very common when new functionality is added to the northbound management interfaces via changes in the YANG data model.

Other software upgrades may include external libraries that ConfD depends on, ConfD client applications such as CDB subscribers using the CDB API, data providers using the DP API, or applications using the northbound MAAPI API.

In this application note, we will talk about some common ConfD software upgrade scenarios and point out some challenges and problems that can arise if not planned for ahead of time.

2. Upgrade Scenarios and Challenges

In production, software upgrades and downgrades are always a risky operation when it comes to ensuring an operational system post-upgrade without loss of configuration data.

In the case of embedded management software, and specifically ConfD which comes with a database for configuration data, there are multiple points of failure that can affect an upgrade and need to be considered:

- Incompatible CDB disk file format between two versions of ConfD. The introduction of new features in ConfD can result in the CDB disk file format to change between major releases.

- Database schema incompatible changes caused by introduced non-backward compatible changes in YANG data models

- Incompatible libraries used on the target. We won't delve into this as this is an obvious matter that needs to be checked during testing. For example, ConfD has dependency on gLibc and OpenSSL. Depending on which CPU architecture you use, ConfD may have different version requirements.

- Incompatible APIs. If ConfD is upgraded and ConfD client applications use an older version of the API that comes with ConfD, they also need to be upgraded. This also should be caught during testing of the upgrade and an upgrade strategy should be planned considering all the southbound clients and MAAPI applications running on the target.

All of these potential points of failure can be checked during testing and can be remediated if you are aware of these backward compatibility issues. When a non-backward compatible change is introduced, knowing the proper way to upgrade will ensure a successful operation.

Before diving into the upgrade scenarios and the potential issues that can occur, it is important to note that you should always backup the database persistent files before performing an upgrade.

Each software release for a network device is typically associated with a certain version of the configuration data layout, i.e. schema. In ConfD, the schema of the database is generated from the YANG data models. The smallest change in YANG can trigger a database schema upgrade.

CDB saves a copy of the schema associated with the data it holds. Every time ConfD starts, CDB will compare the content of the FXS files with its own copy of the schema files. If the schema and the FXS files don't match, ConfD will automatically initiate an upgrade transaction towards CDB. In the simplest case, CDB automatically resolves the changes and commits the new data before ConfD reaches Start Phase One. If it can't automatically upgrade, ConfD will fail startup unless an upgrade client is provided. This should be expected and handled appropriately for non-backward compatible changes in YANG data models.

Often, the major requirement when upgrading ConfD in production is to preserve the existing configuration data that is stored in CDB. The operation may be interrupted if data is lost. In some cases, configuration may reside outside of CDB. In this case, the persistence problem is handled outside of ConfD. The CDB schema depends solely on the YANG data model. Before upgrading ConfD in the field, it is important to make sure existing CDB files will be compatible with the new version of ConfD.

2.1 ConfD Daemon Upgrade

For simplicity, let's call the currently running version of ConfD, version X. A newly upgraded version would then be X+m and an older version would be X-n, with m and n being any positive number within the supported versions of ConfD.

When upgrading an image in production to a new software version that contains a new version of ConfD (X+m), ConfD X+m should be able to read the old CDB data (version X) and upgrade that data to the new CDB format (X+m), if there is a CDB format change too. This is not to be confused with configuration data format (or database schema format), which we refer to here as Schema, and is based on YANG data models. In this scenario, we have the same YANG modules as we had in version X, unless the YANG data models that come with ConfD are also upgraded, usually in a backward compatible way. If they were upgraded in a non-backward compatible way, this will be known ahead of time thanks to the release notes that come in the CHANGES file with each distribution of ConfD.

What do we need to worry about in this case? The challenge is when the CDB file format has changed in a non-backward compatible way and ConfD can't understand the old CDB file format. This happens rarely but can happen, especially between a very old version of ConfD and a new one. Attention should be made to the release notes found in the CHANGES file that comes with each ConfD distribution.

In this case, to mitigate the issue, one should backup CDB in XML files before starting the upgrade process in the field, remove the CDB files, and load the saved configuration data once the system starts running with ConfD X+m.

ConfD distributions include a command line utility called "confd_load" which is used to save the database contents or to load previously saved database contents. In the simplest form "confd_load <filename>" will save the current database contents to the named file in XML format and "confd_load -l <filename>" will load the database from the named file. See the confd_load(1) man page for full information about using confd_load.

This kind of backward incompatibility with CDB files will be documented in the CHANGES files that comes with ConfD. It is also something that can be discovered during testing and before asking the end customer to upgrade.

Note that the CDB file removal is not always needed. ConfD will try to preserve the previous data, but, if it can't, it will start with a fresh CDB. Data would need to be loaded from XML files later on, after ConfD has started in phase 0 at least.

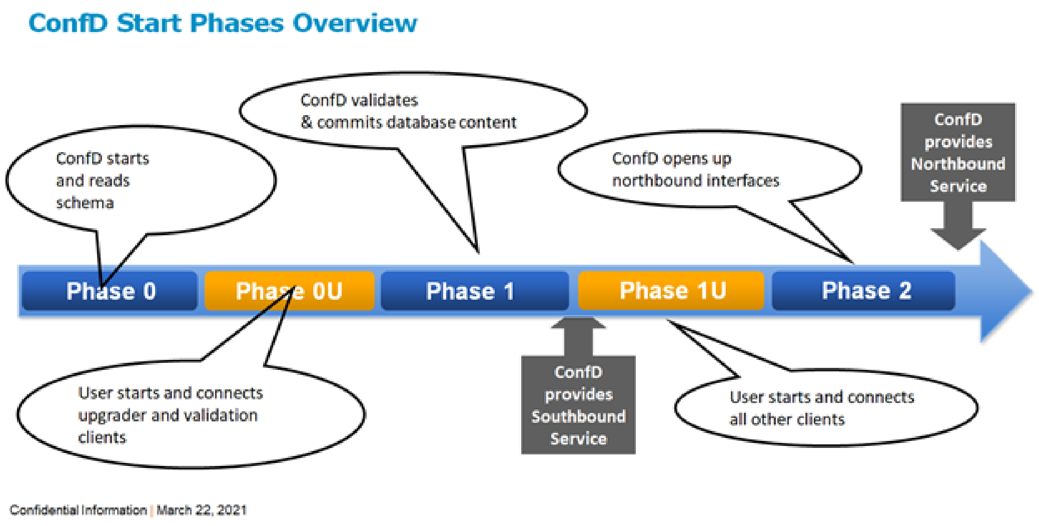
2.2 CDB Schema Upgrade

CDB supports automatic schema upgrades for backward compatible changes in YANG data models. When the end customer upgrades to a new software image that contains the same ConfD version but a different database schema (generated from new YANG data models), ConfD will initiate an upgrade transaction towards CDB during start up. The upgrade transaction's role is to rewrite the existing data following the new schema layout. In some cases, this automatic upgrade process can't be accomplished on its own and that's the case of when non-backward compatible changes are introduced in the new data model. For example, adding a mandatory leaf without a default value is a non-backward compatible change and ConfD will not know what value to assign to this leaf. In this case, the automatic upgrade process fails and ConfD can't start. The logs will contain the reason of the failure.

In the case of non-backward compatible changes in the CDB Schema, driven by non-backward compatible YANG data model changes, ConfD needs external intervention via a user provided upgrade client. ConfD exposes its transaction engine to applications during bring up and an API allows external applications to participate in the upgrade transaction initiated by ConfD when there is a new schema. This is used in scenarios like that described above, to provide ConfD with the missing data during the upgrade.

When non-backward compatible YANG data model changes are introduced, an upgrade client needs to connect to ConfD in Start Phase 0 and attach to the upgrade transaction initiated by ConfD in order to compute the new values for the new database schema.

ConfD start phases and CDB upgrade process:


ConfD Start Phases Overview
ConfD starts and reads schema
ConfD validates & commits database content
ConfD opens up northbound interfaces
ConfD provides Northbound Service
Phase 0
Phase 0U
Phase 1
Phase 1U
Phase 2
User starts and connects upgrader and validation clients
ConfD provides Southbound Service
User starts and connects all other clients
Confidential Information | March 22, 2021

In Start Phase 0, ConfD will:

1. Read the confd.conf configuration file
2. Read .fxs files (the complied YANG data models)
3. Start CDB
    a. CDB in-memory content is blank at this point
    b. Start transaction
    c. Read CDB data from the filesystem
    d. Check if the persistent data is consistent with the new schema
        i. If not, start automatic schema upgrade
            1. Note: CDB doesn't differentiate between upgrade or downgrade; it only detects a schema/FXS difference.
            2. Note: CDB does not rely on the version in the data model namespace declaration.

A CDB automatic schema upgrade can be sufficient for backward compatible schema changes. For example:

- Elements
- Elements removed from the schema
- Elemtns with trivial type changes such as 32-bit integers changing to a 64-bit integers in the new schema
- etc.

If the automatic schema upgrade is not sufficient, a CDB upgrade client needs to be provided, for example when:

- New mandatory field added such as a MTU, with a constraint on the new value being 1500 for Eth interfaces and 48 48 for ATM interfaces. In this case, the upgrade client will read the configured list of interfaces and will write the new MTU values for each instance based on type.
- A type change from string to enumeration.
- Units (unmodelled) change, such as a "Time" field changed from seconds to nanoseconds.

Start Phase 0U represents the User part of Start Phase 0.

In this phase, the upgrades client connects to CDB in order to resolve any data compatibility issues that the automatic upgrade couldn't handle on its own.

- Complement to the automatic schema upgrade
- Connect in Start Phase 0U
- Compute values for the new database values
    - ◊ Read any old or new database values
    - ◊ Compute new values
    - ◊ Write/overwrite new values

In this phase, we can also connect data validator applications and optionally connect Data Provider clients.

2.3 ConfD Clients and ConfD Upgrade

Another possible challenge has to do with ConfD clients that use a ConfD API version X and try to talk to the new ConfD (X+m) or the opposite when new ConfD API clients connect to an older ConfD (X-n). There is a guarantee of backward compatibility in only one case and that’s when an older client (using API version X) connects to the new ConfD (X+m), while m is a maximum of one major release later. This is useful during certain upgrade scenarios, e.g. micro-services.

ConfD clients should also be upgraded to use the same API version as the running ConfD daemon version.

2.4 ConfD and Schema Upgrade

In some cases, there may be two types of upgrades that need to happen. One is a new ConfD version upgrade and the other is a database schema upgrade.

It is a risky operation to both at once in one step as many things can go wrong when performing these upgrades at the same time. The recommendation is do the upgrades sequentially; upgrade one or the other first, then finish the upgrade with the remaining one. ConfD can be upgraded first. This allows ConfD to upgrade the CDB format as well. Once that is successful, a schema upgrade can follow.

The reason for separating these upgrades is to ensure a smoother process with checkpointing in the middle to verify that the CDB format is ready for the new ConfD.

The steps to follow are:

1. Save data to XML files: This step is only needed when a new CDB format has been introduced in the new ConfD version, to be able to load the data after ConfD is upgraded.
2. Stop ConfD X.
3. Start ConfD X+m with the same previous database schema (version X) and data: This succeeds if the CDB format hasn't changed in a non-backward compatible way and ConfD can read and preserve the data. This is more often a problem if upgrading from a very old version of ConfD to a newer one. Step 1 would then be necessary to preserve the previous CDB data.
4. Restart ConfD X+m with the new schema (YANG data models) and rely on the automatic upgrade or use a programmatic upgrade client to migrate the data as seen above.
5. Load data from the XML files into CDB, if applicable (see step 1 and step 3), otherwise ConfD and CDB data should be operational after step 4.

If the new ConfD (X+m) can't read the old CDB format (X), due to a non-backward compatible change in CDB, which is likely to happen if we are upgrading from a very old version of ConfD to a newer one, the following steps will be necessary:

1. Save data to XML files.
2. Stop ConfD X.
3. Delete CDB files.
4. Start the ConfD X+ with the new schema (FXS files).
5. Reload the configuration into the new CDB from XML files, , after preprocessing the XML data if the new schema is non-backward compatible with the previous schema.

If the schema change is non-backward compatible, the saved XML files would need to be preprocessed before loading them into the new CDB.

2.5 ConfD and CDB Downgrade

Downgrading ConfD should always rely on backing up the configuration data to XML files first as it is not possible to guarantee that the new CDB format will be compatible with the older version of ConfD.

With regards to the CDB Schema downgrade, as mentioned above, it's the same as an upgrade. There is no distinction between an upgrade or a downgrade when it comes to the CDB Schema. What matters most is the nature of the changes introduced in the YANG data models; Whether it's a backward or non-backward compatible change.

3. Summary

This application note has discussed various upgrade scenarios and issues to consider when upgrading ConfD. This discussion has been at a high-level and general. For lower-level details, please, consult the User Guide references provided below.

4. For More Information

For more information about ConfD, visit https://www.tail-f.com

For more detailed information, consult these sections of the ConfD User Guide:

- Section 7.9. Automatic schema upgrades and downgrades
- Section 7.10. Using initialization files for upgrade
- Section 7.11. Using MAAPI to modify CDB during upgrade
- Section 7.12. More complex schema upgrades
- Chapter 16. In-Service Data Model Upgrade
- Section 31.4. Starting ConfD

a Cisco company

www.tail-f.com
info@tail-f.com

Corporate Headquarters

Sveavagen 25
111 34 Stockholm
Sweden
+46 8 21 37 40