

RDK FAQ

Access

To set git username and password, use the following commands.

```
$git config --global user.name "Your Name"
$git config --global user.email "youremail@yourdomain.com"
```

Issue:

While trying to do a repository initialization, the user will be prompted for a password.

Example console log:

Password for '<https://code.rdkcentral.com/>': remote: Unauthorized fatal: Authentication failed for '<https://<username>@code.rdkcentral.com/r/manifests/>'

Possible reasons and solutions:

- Not having a valid RDK Credential
 - Obtain valid RDK credentials : Users must have a valid RDK Central Credentials. To register use the link below
 - **RDK Central Registration Link** - <https://wiki.rdkcentral.com/signup.action>
- User registration without valid company email address
 - Users must belong to a licensed company to be able to clone and checkout the manifest. You must register with valid company email address to gain access to manifest if your company is listed as a licensee. To verify if your company is licensed, please check in the following wiki page - [Licensees](#).
- Account is inactive/disabled -
 - Account may have been deactivated/disabled either due to inactivity (60 days) or password has expired - Accounts can be activated using self service. Login to wiki.rdkcentral.com with the username, you will see a message with a link to activate your account. Click on the link to activate your account. If you are still facing issues with activating your account, please email support@rdkcentral.com. Please update your password after it has been activated.
- Account is active, but still getting the password reset error.
 - Unable to login to code.rdkcentral.com with username (All LOWERCASE) and password - If you are getting authentication failure, it is possible that your email address is linked with another username that you have used previously or you have logged in to code.rdkcentral.com with a mixed case username. Either case, you can email to support@rdkcentral.com to mitigate this issue.
 - Able to login to code.rdkcentral.com with username/password - try and browse this repo -<https://code.rdkcentral.com/r/admin/repos/components/generic/mediaframework> . If you are not able to browse this repo, you can email to support@rdkcentral.com to mitigate this issue. if you are able to browse the repo, but unable to clone the it is probably an environment issue. Check your .netrc file

Update the credentials in \$HOME/.netrc file, a sample of the file is given below

.netrc

machine code.rdkcentral.com login <user-name> password <Password>

Issue:

In few incidents, the user may not be having the required permission to download the code although having a valid combination of username /password combination.

Example console log:

Username for '<https://code.rdkcentral.com/>': rdknewuser

Password for '<https://rdknewuser@code.rdkcentral.com/>':

fatal: remote error: Service not enabled

fatal: cannot obtain manifest <https://code.rdkcentral.com/r/manifests>

Possible Solution:

Approach RDK support with the necessary approval to get the requested privilege.

Github no more supports direct password login and allows only **Personal Access tokens**. To generate a new token,

- Goto <https://github.com> -> Settings -> Developer Settings -> Personal Access Tokens
- Click on Generate a new token. You should be able to see a access new token on your UI.
- Please copy the new token and add it to ~/.netrc file as given below
- machine github.com login <github-handle> password <new github access token>

Eg: machine github.com login sampleuser password ghp_73ozr1OhWvUI1htuRCVwPyWF4Va1234ABCD

GitHub Workflow Steps:

- **Create a Fork** by simply clicking on the 'fork' button of the repository page on GitHub.
- **Clone your Fork**, the clone command creates a local git repository from your remote fork on GitHub.

- `git clone https://github.com/USERNAME/REPOSITORY.git`
- *Modify the Code* in your local clone, and commit the changes to your local clone using the `git commit` command.
- *Push your Changes* by invoking the `git push` command, from your workspace, to upload your changes to your remote fork on GitHub.
- *Create a Pull Request* by clicking the 'pull request' button on the GitHub page of your remote fork.

Possible reasons and solutions:

If you are already using the Personal Access Token and getting a 403, chances are that the *token has auto-expired*.

You can generate a new token in Github by going to "<https://github.com> -> Settings -> Developer Settings -> Personal Access Tokens", Clicking on "Generate a new token" and then replacing your old token inside the `~/.netrc` file with the newly generated one.

Build

Mask the recipe in platform specific MACHINE conf file and run the source command again to build.

For example, to MASK `ccsp-gwprovapp-ethwan` in Rpi build, add below line in Rpi specific Machine conf file. Add below line in file

"meta-cmf-raspberrypi/conf/machine/raspberrypi-rdk-broadband.conf "

BBMASK .+= "!.meta-rdk-broadband/recipes-ccsp/ccsp/ccsp-gwprovapp-ethwan.bb"

Run the source command again to build

source meta-cmf-raspberrypi/setup-environment -> select respective conf file

If you encounter below error,

repo: warning: Python 2 is no longer supported; Please upgrade to Python 3.6+.

Downloading Repo source from <https://gerrit.googlesource.com/git-repo>

remote: Counting objects: 1, done

remote: Finding sources: 100% (41/41)

remote: Total 41 (delta 14), reused 41 (delta 14)

Unpacking objects: 100% (41/41), done.

Traceback (most recent call last):

File "<File Path>", line 56, in <module>

from subcmds.version import Version

File "<File Path>", line 38, in <module>

['%s' % name])

File "<File Path>", line 27, in <module>

from hooks import RepoHook

File "<File Path>", line 472

file=sys.stderr)

^

SyntaxError: invalid syntax

Upgrade the repo using below command to use with python3.

```
curl https://storage.googleapis.com/git-repo-downloads/repo-1 > ~/bin/repochmod a+x ~/bin/repopython3 ~/bin/repo init -u <repo URL>python3 ~/bin/repo sync
```

Make sure that the build machine is having required packages installed for specific Yocto flavor.

Eg: <https://docs.yoctoproject.org/3.2.3/ref-manual/ref-system-requirements.html>

Metadata represents the versions of the various components in a distribution, such as the particular versions of the Linux kernel or libraries. The project supplies an example set of metadata that can generate several example distributions. The actual metadata used for the construction of a custom distribution may be supplied by a commercial vendor or created by an embedded developer. The root filesystem is defined in the metadata for a given build of a distribution.

There are tools within BitBake that enable this level of details.

- "`bitbake -g targetname`" creates `depends.dot` and `task-depends.dot` files in the current directory. These files show which packages and tasks depend on which other packages and tasks and are useful for debugging purposes.
- "`bitbake -g -u depexp targetname`" shows results in a more human-readable, GUI style. A simple mount of the resulting root image will show how much storage space is being used.

In addition, the toaster is a new graphical user interface for BitBake that makes these tools much easier to use.

As with any complex system, the real answer is *it depends*, but of course that is not very helpful. The simplest method for adding a single package to your build is to add

the below line to `conf/local.conf`:

- `IMAGE_INSTALL_append = " package"`

Use your own package name in place of **package**. Note the leading space before the package name. If you want to add multiple packages, you can use multiple lines like the above, or list all packages on a single line with:

- `IMAGE_INSTALL_append = " package1 package2 package3"`

Although If you add in `local.conf`, that is not permanent change. For permanent addition of that package in final rootfs, you need to be added in image recipe or any package group which is included in the image recipe.

Prebuilds are handled internally by Yocto by using `sstate-cache`. If a prebuilt from a known good build is available, the build can point to that folder via the `conf` file inside the `./build<buildtype>/conf/` folder so that the prebuilds are picked up from the location

This depend entirely on multiple factors like capacity of build machine, first time build or repeated build in the same work space as well as changes in components on which the component in question depends on(if there is a change, the depending component is first built and then the dependent component) and hence cannot be answered directly.

When you checkout sandbox every component is independently buildable, and `bitbake` (OE's build engine) is responsible to identify and sort the component dependency chain and ensure its built along. if you were to build a single component the commands are

```
bitbake <component>
```

where component is in one to one relation with `.bb` (recipe) file that can be found in the Yocto/OE metadata (`meta-rdk*` layers) e.g. if you were to build `rdkbrowser` then you would see that its recipe is housed in generic layer called `meta-rdk-cmf` and recipe is called `rdkbrowser.bb` so the command would be

```
bitbake rdkbrowser
```

However this will only generate CMF component and for packaging it up into final image you still have to build the image component to repackage `rdkbrowser`

```
bitbake rdk-generic-hybrid-wpe-image
```

would generate the CMF generic image for hybrid devices. it will only rebuild the affected components when building the image if nothing has changed it will not recreate the image.

`bitbake` has division of work into individual tasks for a component. Secondly, the recipes are wired to notice changes in upstream repository as well when you do `repo sync`. You can use below command to see what all individual tasks are available.

```
bitbake -c listtasks <component>
```

It will show an output like

<code>do_build</code> 'build' a recipe	Default task for a recipe - depends on all other normal tasks required to
<code>do_bundle_initramfs</code>	Combines an initial ramdisk image and kernel together to form a single image
<code>do_checkuri</code>	Validates the <code>SRC_URI</code> value
<code>do_checkuriall</code>	Validates the <code>SRC_URI</code> value for all recipes required to build a target
<code>do_clean</code>	Removes all output files for a target
<code>do_cleanall</code> target	Removes all output files, shared state cache, and downloaded source files for a
<code>do_cleansstate</code>	Removes all output files and shared state cache for a target
<code>do_compile</code>	Compiles the source in the compilation directory

You can rerun any of the above tasks

```
bitbake -C compile rdkbrowser
```

would force recompile of `servicemanager`, if you wish to perform all the build steps for a component you can do that too by

```
bitbake -c cleansstate rdkbrowser; bitbake rdkbrowser
```

Similarly, in general you can have:

```
bitbake <target> -c<task_to_be_executed>
```

This will ensure `do_<task_to_be_executed>()` will be called.

`task_to_be_executed` can `fetch`, `unpack`, `configure`, `compile`, `install`, `package` etc

If we were to draw parallels

```
--rebuild = bitbake -c cleansstate <component> ; bitbake <component>
```

```
--build-only = bitbake <component>
```

--skip-package is delegated to shared state mechanism to figure out at present.

Unless you do 'repo sync' sources will not be updated

'repo sync' can cover only components with external src support, it means that in cases when `SRCREV` is set to `AUTOREV` and component doesn't support external src, then `bitbake` will try to update sources from a remote repository during build time.

There is no documentation for AUTOREV, but it's doing only one function - check remote repository for any new sources updates.

```
AUTOREV = "${@bb.fetch2.get_autorev(d)}"
```

You can prevent this behaviour by changing `BB_SRCREV_POLICY` variable in you local `<sandbox>/conf/local.conf`

```
BB_SRCREV_POLICY = "cache"
```

BB_SRCREV_POLICY

Defines the behavior of the fetcher when it interacts with source control systems and dynamic source revisions. The `BB_SRCREV_POLICY` variable is useful when working without a network. The variable can be set using one of two policies:

`cache` - Retains the value the system obtained previously rather than querying the source control system each time.

`clear` - Queries the source controls system every time. With this policy, there is no cache. The "clear" policy is the default.

These are OE metadata variables, bitbake has preprocessing options where it expands all the local bitbake variables so you could take advantage of that option to figure it out

```
bitbake -e rf4ce | grep "^S ="
```

```
bitbake -e rf4ce | grep "^WORKDIR ="
```

It can be used to check for any bitbake variable, alternatively you can pipe the whole bitbake -e output to a file and inspect the file in your favourite editor

- You can find log files for each task in the recipe's temp directory. Log files are named log.taskname-Eg: *compile logs are present in the file log.do_compile*. Bitbake maintains logs separately for each of the tasks that run while building the component. These tasks are typically the fetch task, the compile task, install task and so on.
- Eg: For a RPI device, typically logs are present under the build-raspberrypi-rdk-hybrid/tmp/work/<*-rdk-linux> and under the component directory. Other devices would have logs present under similar folders. For instance, logs are present under build-raspberrypi-rdk-hybrid/tmp/work/<*-rdk-linux>/rdkbrowser/1.99+gitAUTOINC+8c3f17fdc6_fa6c8b4334_dc33f7d6bc_4bc1f2f4c9-r0/temp

The warnings like `-Werror -Wall -Wextra` are turned on for compiler for most of the Ccsp components. All these warnings must be fixed for successful compilation as all warnings are treated as errors.

Warnings due to unused parameters in code can be fixed by using:

`UNREFERENCED_PARAMETER(user_data);`, if `user_data` variable is not used in the scope of definition of the function.

* `satisfy_dependencies_for`: Cannot satisfy the following dependencies for packagegroup-....:

* `net-snmp` *

* `opkg_install_cmd`: Cannot install package packagegroup-....

The above error indicates that :

You asked for adding `net-snmp` (the package) not (the recipe) now `net-snmp` (the recipe) may generate a number of (packages) so you should add the packages (runtime items) to the package groups and not the recipes (build time items). Usually yocto/OE does generate a output package with same name as input recipe so for [net-snmp.bb](#) there will be a `net-snmp` ipk but thats just a common case not a hard and fast rule.

Now in this particular case when a package has nothing to emit into the `$(PN)` package the package is left empty and hence not emitted. If you want to emit the package regardless you have to add

`ALLOW_EMPTY_<package> = "1"` in the recipe, but this is less of a usecase to demand empty packages. If you expressed the packagegroup `RDEPENDS` correctly you would not need it.

`sstate-cache` is always adding new versions and hence is growing in size always, thankfully we have a tool in Yocto to manage it, here is a sample on how to do it for **raspberrypi** you can set `WORKSPACE` and `MACH` variables to point to the values for machine

sstate-manage.sh

```
# cd into top of workspace say ${WORKSPACE}
MACH=raspberrypi
MACHINE=${MACH} . ./meta-rdk/setup-environment
EXTRALAYERS=`bitbake -e | grep '^BBLAYERS=' | tr -s ' ' ',' | tr -d '"' | sed -e 's%^BBLAYERS=%%' -e 's%,,%%'`
# remove stale sstate on mirror
${WORKSPACE}/openembedded-core/scripts/sstate-cache-management.sh -v --yes \
    --extra-layer=${EXTRALAYERS} \
    --cache-dir=${WORKSPACE}/sstate-cache \
    --stamps-dir=${WORKSPACE}/build-${MACH}/tmp/stamps
```

To enable kernel/busybox features you can append metadata to the recipes (i.e. `.bb` files) by simply creating an append file (i.e. `.bbappend` files) and including metadata in it. If the features needs to be enabled across all the platforms then add in **meta-rdk-rpi** meta or if it's specific to a platform then append to the recipes available in OEM layer specific to the platform.

Below example shows how to enable IPsec on Yocto builds:

- Create a `stblinux_%.bbappend`
- Create a file called `enable_netkey.cfg` with following content
 `CONFIG_NET_KEY=y`
- Add `SRC_URI_append = "file://enable_netkey.cfg "`

Make sure `.bbappend` has the same root name as their corresponding `.bb` recipes.

- Sometimes when you have a working branch which is not checked out or has uncommitted changes then `repo` will fail when you try to sync to the latest code base.
- Sample failure logs:

```
error: generic/devicesettings/generic/: contains uncommitted changes
```

```
error: generic/rdkbrowser/: branch master is published (but not merged) and is now 11 commits behind
```

```
error: meta-rdk-oem-X/: contains uncommitted changes
```

- To resolve this you need to checkout the branch and rebase it to the master using below commands.

```
git rebase --abort
```

```
git rebase rdkgerrit/master ( or rdkgerrit/stable2)
```

- During the process `git` might throw conflict errors if it cannot merge files automatically. Then you need to merge manually using `Vim` or any other text editors. But it can be simple if you know exactly what changes needs to be saved / removed.
- For example you can use below command to keep your changes

```
git checkout --ours FILE
```

- If you want to run on multiple files then use below command.

```
grep -lr '<<<<<<<<' | xargs git checkout --ours
```

- Similarly, you can use below commands if you want to keep other changes.

```
git checkout --yours FILE
```

```
grep -lr '<<<<<<<' | xargs git checkout --theirs
```

- Verify the files from which recipe they are being generated under `build-<platform>/buildhistory/packages/mips32el-rdk-linux` folder
- Add the below line format under that particular recipe.

```
FILES_${PN}-dev += " file1 file2 etc"
```

- Verify that `cat <package>-dev/latest` should contain these files under `FILELIST`.

Yocto project build system has a utility which can provide information about which package (`ipk` or `rpm`) is providing a given file, this helps in finding further information on packaging e.g. if you want to do more finer packaging, run the following command in your build environment

```
oe-pkgdata-util find-path /lib/libc.so.6
glibc: /lib/libc.so.6
```

Common build failures are reported in Yocto builds. Some build failures are hard to analyze with logs, unless we get access to the failure workspace. In most cases they are hard to reproduce on local workspace. We go through multiple iteration of builds, lock down the node and then debug. To debug these failures use `Packages` file found under `tmp/deploy/ipk` directory on your local workspace.

The signal core dump that are generated under `/tmp` can be decoded using `gdb`

Procedure:

1. In `<image>.bbappend` file
 `IMAGE_INSTALL_append = " gdb"`
2. In `local.conf`
 `INCOMPATIBLE_LICENSE = "GPL-3.0 LGPL-3.0 AGPL-3.0"`
 `INCOMPATIBLE_LICENSE_pn-gdb = ""`
 `EXTRA_IMAGE_FEATURES += "tools-debug"`
 `EXTRA_IMAGE_FEATURES += "dbg-pkgs"`
3. In `<component>.bbappend`
 `CFLAGS += " -D_DEBUG -g "`
4. Compile and flash the binary to device
5. Run `gdb -c <path to signal dump> <binary>`

This should be because of architecture bit mismatch. To overcome this should either choose the right target platform or put the executable file as a tar file in `bb` file.

Issue:

While building the stack, the bitbake process will be aborted if the disk space runs low beyond the minimum requirement.

Example console log:

WARNING: The free space of /mnt/home/gpsahu01/cmfs/test/build-qemux86hyb/tmp (/dev/vdb) is running low (0.999GB left)

ERROR: No new tasks can be executed since the disk space monitor action is "STOPTASKS"!

WARNING: The free space of /mnt/home/gpsahu01/cmfs/test/downloads (/dev/vdb) is running low (0.095GB left)

ERROR: Immediately abort since the disk space monitor action is "ABORT"!

NOTE: Sending SIGTERM to remaining 1 tasks

Possible solution:

Yocto stack requires more than 30 GB of free disk space for build to complete, so it is required to keep sufficient disk space available before starting the build process.

Issue:

Bitbake complains about a missing "sys/cdefs.h" and the error can be encountered in random recipes when we move from one build host to other.

Example console log:

```
compilation terminated.
| In file included from /usr/include/stdio.h:27:0,
|     from /src/kern_head.c:13:
| /usr/include/features.h:374:25: fatal error: sys/cdefs.h: No such file or directory
| # include <sys/cdefs.h>
|     ^
| compilation terminated.
| In file included from /usr/include/stdio.h:27:0,
|     from /src/ssstrip.c:9:
| /usr/include/features.h:374:25: fatal error: sys/cdefs.h: No such file or directory
| # include <sys/cdefs.h>
|     ^
| compilation terminated.
| make: *** [encode] Error 1
| make: *** Waiting for unfinished jobs....
| make: *** [kern_head] Error 1
| make: *** [ssstrip] Error 1
| ERROR: oe_runmake failed
| WARNING: exit code 1 from a shell command
```

Possible Solution:

This issue may be caused by a missing "g++-multilib" package in the build host (observed in Ubuntu 14.4). Installing the package with "sudo apt-get install g++-multilib" should resolve this issue. Also the build machines should be configured following the procedure as per Setup guide to avoid similar issues.

Issue:

The bitbake process terminates after complaining about a non-existent path or environment variable.

Example console log #1:

```
ERROR: Function failed: iarmbus: LIC_FILES_CHKSUM points to an invalid file: ${RDK_ROOT_PATH}/components/generic/iarmbus/core/include/libIARM.h
```

Example console log #2:

```
ERROR: ParseError at /mnt/home/gpsahu01/cmfs/test/meta-virtualization/recipes-extended/images/cloud-image-controller.bb:29: Could not inherit file classes/image-vm.bbclass
```

Possible solution:

This may be due to a wrongly setup environment e.g. we have executed "meta-rdk/setup-environment" instead of sourcing "meta-cmf/setup-environment"

Issue:

The issue is observed during setup and machine selection stage, setup-environment script will throw unexpected error about non-existing layer paths.

Example console log:

```
gpsahu01@dvm-wdcc-tata-001:~/cmf/emulator-2.1-20160919$ source meta-cmf/setup-environment
```

```
1) meta-raspberrypi/conf/machine/raspberrypi0.conf      7) meta-rdk-bsp-emulator/conf/machine/qemux86hyb.conf
2) meta-raspberrypi/conf/machine/raspberrypi2.conf      8) meta-rdk-bsp-emulator/conf/machine/qemux86mc.conf
```

```
[...]
```

Please enter your choice of machine [1..11]: 7

Shell environment set up for builds.

Writing auto.conf ...

Writing versions.txt ...

-bash: cd: ../meta-browser//: No such file or directory

fatal: Not a git repository (or any parent up to mount point /mnt)

Stopping at filesystem boundary (GIT_DISCOVERY_ACROSS_FILESYSTEM not set).

-bash: ../patches/rdk-oe/meta-linaro/*.*patch: No such file or directory

-bash: cd: ../meta-openembedded//: No such file or directory

fatal: Not a git repository (or any parent up to mount point /mnt)

Stopping at filesystem boundary (GIT_DISCOVERY_ACROSS_FILESYSTEM not set).

Possible solution:

If the host PC has set colored terminal output for commands then it may cause unexpected errors being shown during execution of meta-cmf/setup-environment script. To fix the problem we can run following command:

gpsahu01@dvm-wcdcc-tata-001:~/cmf/emulator-2.1-20160919\$ alias ls='ls --color=no'

Issue:

Fetch timeout or failure can happen due to following network problems.

- Network not accessible
- Restriction in Firewall
- Invalid Proxy configuration
- Unable to resolve DNS in IPv6 networks

Example console log:

```
Fetching projects: 96% (91/94) Fetching project openembedded/meta-linaro
Fetching projects: 97% (92/94) fatal: read error: Connection timed out
fatal: read error: Connection timed out
fatal: read error: Connection timed out
error: Cannot fetch meta-virtualization
warn: --force-broken, continuing to sync
Fetching projects: 98% (93/94) error: Cannot fetch meta-java
warn: --force-broken, continuing to sync
Fetching projects: 100% (94/94)
error: Exited sync due to fetch errors
```

Possible solution:

- Using VPN may have some restrictions sometime it may not allow GIT access.
- Ensure that the ports for HTTPS, SSH, HTTP are opened by the firewall and the policy doesn't block common open source repositories.
- In case of IPv6 networks issues, force GIT to use IPv4.

Also Following options can be considered while debugging:

Option #1) Need to flush the IP rules:

enter the command

\$ iptables -F

and check

\$) git clone [git://git.lighttpd.net/lighttpd/lighttpd-1.x.git](https://git.lighttpd.net/lighttpd/lighttpd-1.x.git)

Option #2) Check for the port 22 is open or not by doing nmap

\$ nmap -p 22 10.11.107.0-255"

(check for ipaddress 10.11.106.62)

\$ ssh -v [git@github.com](https://github.com)

if it adds it will ask for input()yes/no- Type yes

\$ git clone [git://github.com/lighttpd/lighttpd1.4.git](https://github.com/lighttpd/lighttpd1.4.git);branch=lighttpd-1.5.x

Option #3) replacing the git// with https:// which uses port 443

\$ git config --global url."https://".insteadOf git://

and try **\$ "git clone [git://git.lighttpd.net/lighttpd/lighttpd-1.x.git](https://git.lighttpd.net/lighttpd/lighttpd-1.x.git)"**

Option #4) In a few cases, the access to GIT repository is via SSH. To use SSH URLs with GIT repository, an SSH key-pair must be generated on the build PC and add the public key to your GitHub account.

For information on setting up an SSH key-pair, see "Generating an SSH key."

<https://help.github.com/articles/generating-an-ssh-key/>

Issue:

When we try to build a very old branch of the code, the manifest file will not be up-to-date as few of the open-source URLs might not be continuing support of older branches or versions of software.

Example console log:

```
WARNING: Failed to fetch URL git://code.qt.io/qt/qtlocation.git;branch=stable, attempting MIRRORS if available
ERROR: Fetcher failure: Unable to find revision f28408346243cf090326f4738fd838219c21e00f in branch stable even from upstream
ERROR: Function failed: Fetcher failure for URL: 'git://code.qt.io/qt/qtlocation.git;branch=stable'. Unable to fetch URL from any source
```

Possible solution:

It is recommended to build with more recent branches, as the code will be well maintained and will have updated features.

Issue:

An Open source URL is broken either due to the website is down temporarily or it is permanently removed.

Example console log:

```
WARNING: Failed to fetch URL http://directfb.org/downloads/Core/linux-fusion/linux-fusion-9.0.3.tar.gz, attempting MIRRORS if available
ERROR: Fetcher failure: Fetch command failed with exit code 4, no output
ERROR: Function failed: Fetcher failure for URL: 'http://directfb.org/downloads/Core/linux-fusion/linux-fusion-9.0.3.tar.gz'. Unable to fetch URL from any source
```

Possible Solution:

Temporary workaround: In case of archives (.tar or .zip etc.), if the file is available from a previously built stack then it can be copied and an empty file with the name <archive-name>.done has to be created to bypass looking for downloading the file.

Fixing the recipe: If the problematic recipe is available from any other alternative mirror, update the same in SRC_URI part of the recipe. Few components may be available in common mirrors such as github, web.archive.org, oipf.tv etc.

Looks like memory issue hence changing Ubuntu to 64 bit version should resolve the issue. The below are the Ubuntu configurations,

- Ubuntu 16.04 - 64 bit
- 6GB RAM

16GB Swap

To include a specific feature that is not available in base build, enable the feature specific DISTRO flag in platform specific config file. For example to include USPA feature in Rpi build,

Add the DISTRO specific flag in Rpi platform specific conf file

In File `meta-cmf-raspberrypi/conf/distro/include/rdk-rpi.inc`

Add `DISTRO_FEATURES_append = " usppa"` (to include the feature if not there)

`DISTRO_FEATURES_remove = " usppa"` (to remove the feature)

Make sure the recipe is part of the package build

In File `meta-rdk/recipes-core/packagegroups/packagegroup-rdk-ccsp-broadband.bb` must be included as a DISTRO protected feature `RDEPENDS_packagegroup-rdk-ccsp-broadband += "\ ${@bb.utils.contains('DISTRO_FEATURES', 'usppa', 'usp-pa', '', d)}`"
ERROR: ParseError at /home/a1602446/build-raspberrypi-rdk-broadband/conf/local.conf:247: Could not include required file conf/distro/include/##RDK_FLAVOR##.inc

Above error occurs intermittently, which can be fixed by retrying the source command for setup_environment file.

In case of Rpi, it is

- `source meta-cmf-raspberrypi/setup-environment`

Please refer [Compile-time Build Variants Flags](#)

Issue:

Bitbake fails on populate sysroot state when building with an un-clean stack.

Example console log:

```
ERROR: Function failed: llvm_sysroot_preprocess (log file is located at /mnt/home/gpsahu01/cmf/test/build-qemux86hyb/tmp/work/i586-rdk-linux/llvm3.3/3.3-r0/temp/log.do_populate_sysroot.9648)
```


Possible solution:

This may happen when a previous build process was unexpectedly terminated or aborted. Re-build after cleaning the problematic recipe or image (bitbake <recipe> -c cleanall) would fix the issue.

Issue:

Bitbake terminates the compilation process on 'do_patch' task. This may happen in following cases:

- When using an old recipe file where the SRC_URI link has updated its folder structure.
- Wrongly formatted patch file (run dos2unix for conversion)
- Incorrect patch level (p0, p1, etc.)

Example console log:

```
ERROR: Command Error: exit status: 1 Output:
Applying patch 0001-src-Makefile.am-Dont-check-if-we-are-cross-compiling.patch
can't find file to patch at input line 18
Perhaps you used the wrong -p or --strip option?
The text leading up to this was:
-----
```

```
diff --git a/src/Makefile.am b/src/Makefile.am
index eb50e37..c1e3d64 100644
--- a/src/Makefile.am
+++ b/src/Makefile.am
-----
No file to patch. Skipping patch.
2 out of 2 hunks ignored
Patch 0001-src-Makefile.am-Dont-check-if-we-are-cross-compiling.patch does not apply (enforce with -f)
ERROR: Function failed: patch_do_patch
```

Possible Solution:

From the above output it seems that the file to which patch will be applied is not found, possible reason may be the source folder structure doesn't match with the destination folder structure. E.g. the source directory or **\${S}** starts from the relative path 'src' folder and we are trying to patch outside of it.

By default bitbake patches the files with patch level 'p1' so creating a patch file which matches destination folder structure would solve this issue. Another option is to alter the patch level. E.g.

SRC_URI += file://docsis_3383.patch;striplevel=0

Issue:

Bitbake complains about md5sum mismatch when a recipe has retained old md5sum value while the source file is updated.

Example console log:

```
ERROR: gst-plugins-playersinkbin-noop: md5 data is not matching for file://gstplayersinkbin.c;md5=0f518921aef846c156f91ce4dd6b7c76
ERROR: gst-plugins-playersinkbin-noop: The new md5 checksum is 958142c8f2783c6c4f357f561585b4da
```

Possible solution:

Update the new md5sum value of the file in recipe. This can be done using following steps:

```
.../meta-rdk/recipes-extended/gst-plugins-playersinkbin/files$ md5sum -t gstplayersinkbin.c
958142c8f2783c6c4f357f561585b4da gstplayersinkbin.c
```

Now change the above value in recipe:

LIC_FILES_CHKSUM = "file://gstplayersinkbin.c;md5=958142c8f2783c6c4f357f561585b4da \"

Example Console Log :

```
repo: warning: Python 2 is no longer supported; Please upgrade to Python 3.6+.
```

```
Downloading Repo source from https://gerrit.googlesource.com/git-repo
```

```
remote: Finding sources: 100% (32/32)
```

```
remote: Total 32 (delta 14), reused 32 (delta 14)
```

```
Unpacking objects: 100% (32/32), done.
```

```
File "/mnt/home /cmf/.repo/repo/main.py", line 79
```

```
file=sys.stderr)
```

```
^
```

```
SyntaxError: invalid syntax
```

If you're using an older system without Python 3.6+, try downloading an older version of the Repo Launcher that still supports Python 2.7.

Possible Solution :

create a bin directory

```
mkdir ~/bin
```

```
export PATH=~/bin:$PATH
```

```
curl https://storage.googleapis.com/git-repo-downloads/repo-1 > ~/bin/repo
```

```
chmod a+x ~/bin/repo
```

```
ERROR: trower-base64-git+AUTOINC+fb9440ae2-r0 do_fetch: Fetcher failure: Unable to find revision fb9440ae2bc1118866baefcea7ff814f16613dd in branch master even from upstream
```

```
ERROR: trower-base64-git+AUTOINC+fb9440ae2-r0 do_fetch: Fetcher failure for URL: 'git://github.com/Comcast/trower-base64.git'. Unable to fetch URL from any source.
```

```
ERROR: trower-base64-git+AUTOINC+fb9440ae2-r0 do_fetch: Function failed: base_do_fetch
```

```
ERROR: Logfile of failure stored in: /builds/__repo/build-brcm968360GW/tmp/work/cortexa7t2-vfp-rdk-linux-gnueabi/trower-base64/git+AUTOINC+fb9440ae2-r0/temp/log.do_fetch.18310
```

```
ERROR: Task (/builds/__repo/meta-rdk-ext/recipes-support/trower-base64/trower-base64_1.0.bb:do_fetch) failed with exit code '1'
```

Few points to check here,

- First check if the SRC_URL can be accessed manually in browser
- Check the path of the URL is proper and check the branch detail as well

For example, above error can be fixed by appending **branch=main** in the SRC_URL path

Where to Find

RDK Home :

- [RDK Central Wiki](#)

RDK-V :

- [RDK Video Documentation](#)
- [RDK-V Features](#)
- [RDK-V Porting Guide](#)
 - [RDK-V 4.0 - SOC Porting Guide](#)
 - [RDK-V 4.0 - OEM Porting Guide](#)
- [RDK-V Raspberry Pi Build Guide](#)
- [RDK-V Emulator Build Guide](#)

RDK-B :

- [RDK Broadband Documentation](#)
- [RDK-B Features](#)
- [RDK-B Porting Guide](#)
- [RDK-B Raspberry pi Build guide](#)
- [RDK-B Turris Omnia](#)
- [RDK-B Emulator Build Guide](#)

RDK Reference Platform :

- [RDK Reference Platform](#)
- [RDK-B Reference Platform](#)
 - [RDK-B Emulator](#)
 - [RDK-B Raspberry Pi](#)
 - [RDK-B Turris](#)
- [RDK-V Reference Platform](#)
 - [RDK-V Emulator](#)
 - [RDK-V Raspberry Pi](#)
- [RDK-C Reference Platform](#)
 - [RDK-C Raspberry Pi](#)
 - [RDK-C Raspberry Pi Build guide](#)

RDK Accelerator :

- [RDK Accelerator Home Page](#)
- [Product Specifications](#)
- [RDK Video Accelerator Certification](#)
- [Getting Started - OEM](#)
- [Getting Started - SoC](#)
- [Getting Started - MSO](#)

RDK Device Management :

- [RDK Device Management](#)
 - [Telemetry](#)
 - [Telemetry 2.0: Report Profiles](#)
 - [LogUpload in RPI \(RDK-V\) using XConf Simulator](#)

WEBINARS Home :

- [WEBINARS Home](#)
- [RDK Videos](#)

Test Development Kit (TDK) :

- [Test Development Kit \(TDK\) Home](#)
 - [TDK-V Documentation](#)
 - [TDK-B Documentation](#)
 - [TDK-C Documentation](#)
- [RDK Video Certification](#)
- [TDK-V Porting on Raspberry Pi](#)
- [TDK FAQ Document](#)

Release details:

- [RDK Releases](#)
 - [RDK-B Releases](#)
 - [RDK-V Releases](#)
 - [RDK-C Releases](#)
- [Reference Platform Releases](#)

Reference Platform

The dmcli command for emulator platform use simu instead of eRT and this is the reason for failing in emulator.

A sample command is :

dmcli simu getv Device.

1.RDK-B

The devices can connect over wifi to Raspberrypi by knowing the SSID and the corresponding password of the RaspberryPi Wifi network, which will be WPA2-PSK protected.

1.1.Emulator

As of now, RDKM is not providing a ready-to-use pre-built image for emulator. But you can easily create an emulator build in an Ubuntu linux machine by following the instructions given at [RDK-B Emulator Build Instructions](#) .

1.2.Turris Omnia

```
mkdir tmp/;
cd tmp/
wget https://repo.turris.cz/hbl/omnia/packages/turrispackages/u-boot-omnia_2019-07-7_arm_cortex-a9_vfpv3.ipk
tar xf u-boot-omnia_2019-07-7_arm_cortex-a9_vfpv3.ipk
tar xf data.tar.gz
cd usr/share/omnia/
flash_eraseall /dev/mtd0
nandwrite /dev/mtd0 uboot-devel
cd ../../..
rm -rf tmp
mkdir tmp; cd tmp/
wget https://repo.turris.cz/hbl/omnia/packages/turrispackages/rescue-image_3.2-1_arm_cortex-a9_vfpv3.ipk
tar xf rescue-image_3.2-1_arm_cortex-a9_vfpv3.ipk
tar xf data.tar.gz
cd usr/share/rescue-image/
flash_eraseall /dev/mtd1
nandwrite /dev/mtd1 image.fit.lzma
cd ../../..
rm -rf tmp
```

When we pass Legacy zImage(kernel) image with newer model of turris Omnia we get this error. We need to provide FIT rescue image for newer model of Turris Omnia

Go into u-boot prompt and get back to OpenWRT (failsafe) OS

```
=>env set yocto_mmclod false
```

```
=>saveenv
```

```
=>reset
```

Format /dev/mmcbk0p3 and /dev/mmcbk0p5 and new RDKB firmware into p3 and p5 partitions

To get back to RDKB image

```
=>env set yocto_mmclod setenv bootargs "\"$yocto_bootargs cfg80211.freg=${$regdomain}\"; ext2load mmc 0:3 0x01000000 zImage\; ext2load
```

```
mmc 0:3 0x02000000 armada-385-turris-omnia.dtb
```

```
=>saveenv
```

```
=>reset
```

Follow "Flashing with Medkit & Sysupgrade images" section in <https://wiki.rdkcentral.com/pages/viewpage.action?pageId=114986683>

Legacy and FIT image

TurrisFwUpgrade.sh is written with two models in mind(older and newer)Older model has this partitions in internal flash

mmcbk0p1 - openwrt bootfs

mmcbk0p2 - openwrt rootfs

mmcbk0p3 - rdk bootfs

mmcbk0p4 - Extended partition

mmcbk0p5- rdk rootfs(1)

mmcbk0p6 - nvram

mmcbk0p7 - rdk rootfs(2) for firmware upgradeNewer model has this partitions in internal flash

mmcbk0p1 - rdk bootfs

mmcbk0p2 - rdk rootfs(1)

mmcbk0p3 - rdk rootfs(2) for firmware upgrade

mmcbk0p4 - Extended partition

mmcbk0p5 - nvramThis TurrisFwUpgrade.sh script will fill new rootfs in alternate partition swap the rootfs partition and copy zImage (in bootfs) in internal flash memory.

2.RDK-Camera

We can change the resolution in rms.conf file that is there in /usr/local/rms/bin/ directory.

Yes, we have used AWS server to store 24*7 data for CVR.

Use gstreamer plugin mainly, v4l2src plugin to capture data from camera.

AS of now we used confidential page to check this feature,we are trying to support google based signalling server to resolve this conflict.

h264 encoded format is supported for RMS live streaming and Continuous video recording(CVR).

RDK-V

No. An organization must be a member of the [Digital Living Network Alliance](#) to obtain a DLNA license.

Media-streamer module provides content for DLNA. The implementation is a part of the media-streamer component, and is implemented on top of the libgupnp code.

Yes.

Use below command format -

- `repo init -u https://code.rdkcentral.com/r/manifests -m rdkv.xml -b rdkv-20200207`

(The quarterly release name can be given with -b option)

It is possible to consume mediaplayer events in Lightning -http://cdn.dashjs.org/latest/jsdoc/streaming_MediaPlayerEvents.js.html

Example:

```
const events = ["playbackStateChanged", "playbackCompleted", "playbackSpeedChanged",  
"playbackFailed", "mediaMetadata", "timedMetadata", "playbackProgressUpdate",  
"playbackStarted", "bufferingChanged", "durationChanged", "decoderAvailable" ];  
  
events.forEach(event => {  
  
  player.addEventListener(event, (e) => {  
  
    this.fire(event, {event: e});  
  
  });  
  
});
```

IARM bus interface are used to get the events notification from the system level like IR key, power, storage space, etc. could be better used in native apps.

ServiceManager acts as a subscriber who will receive the events from IARM Bus and [post it to MSO Backend](#). The ServiceManager is the one well known facility for cloud-based applications to gain access to device vended functionality whether they are written in HTML . The better alternative is of course the "AbstractService" class as it is the newer version and derived from the "Service" class and has a superset of features of the service class. However, pure virtual functions may be useful for enforcing OOPS convention of data hiding and abstraction.

First modify the config/xml file which will be there in the component you are writing objects and then respective handlers you have to call in xml.

- WPE doesn't support WebAssembly.
- It won't be faster for general use cases and memory management is worse since it's a fixed heap without garbage collection.
- Web assembly is interesting but currently they're targeting big monolithic codebases in c++ etc like games but more on desktop memory profiles. All the feedback we got thus far has told us their experience hasn't been a win on perf or memory

WPE WebKit has limited support for `HTMLMediaElement.canPlayType()` and `MediaSource.isTypeSupported()`. It implements checks for container, video width/height/framerate, and basic verification of codec (it doesn't check for profile & level)

Make sure DASH.js uses utf-8 as a content type format for license requests (playready challenge is utf-8 encoded)

- Yes, WPE media element: "aamp://" schema WPE AAMP: UVE API https://firebolt.app/aamp/UVE/index.html
- WPE: no native subtitles support (608/708 could be integrated from main UI application)

WPE media element: no audio track selection; WPE MSE Google Shaka: `selectAudioLanguage(language, role)`, or `selectVariantTrack(track)`; WPE AAMP UVE: set

Inject the following JS code to handle MoneyBadger requests (some)

```
window.ServiceManager = {};  
window.ServiceManager.version = '2.0';  
window.ServiceManager.getServiceForJavaScript = function(name, serviceReadyCb) {  
  class BridgeObject  
  {  
    JSMessagesChanged(msgStr)  
    {  
      let msg = JSON.parse(msgStr);  
      console.log('badger action=', msg.action, 'pid=', msg.pid);  
      if (msg.action === 'deviceCapabilities' || msg.action === 'info') {  
        setTimeout((msg) => {  
          let caps = {  
            videoDimensions: [3840, 2160],  
            hdr: { settopHdrSupport: 'DolbyVision', tvHdrSupport: 'DolbyVision' },  
            hdcp: { connected: true, currentHDCPVersion: '2.2' }  
          }  
          window.$badger.callback(msg.pid, true, caps)  
        }, 0, msg);  
      } else {  
        window.$badger.callback(msg.pid, false, {})  
      }  
    }  
    return true  
  }  
}  
if (name === 'com.comcast.BridgeObject_1')  
  serviceReadyCb(new BridgeObject);  
}
```

Open <https://www.ssllabs.com/ssltest/viewMyClient.html> on box to get a report of supported TLS versions and ciphers

Yes, it is possible to use a remote control with rdkbrowser2.sh

`touch /opt/remote_input_enable` and reboot.

By default local storage is disabled. If the app requires this support, you need to contact Project management to enable it for the specific app.

You can modify the script to avoid the deletion of older files:

<https://code.rdkcentral.com/r/plugins/gitiles/rdk/components/generic/crashupload/+/refs/heads/rdk-next/uploadDumps.sh#321>

Client requests a tuner through URL(http,live), TRM server receives the request and checks for the valid reservation and reserves the tuner so that the client is provided with the service requested. Client also can extend or delete the reservation. It is also possible for a client to request a list of the active reservations.

For more details, Please refer <https://wiki.rdkcentral.com/display/RDK/TRM#TRM-HowTRMworks>

`getDeviceInfo()` - Retrieves the device information of the device .

To get a mac address `getDeviceInfo()` needs to be called with string "getMacAddress" as an argument like `getDeviceInfo("macAddress")`

Yes , Please refer this link : [Session Manager](#)

No

End state architecture has scalability and security built into it.Refer [RDK Alexa Echo Dot Voice Application](#) .

This is a sample pipeline to play dtcp encrypted content using `gst-launch`

`gst-launch-1.0 httpsrc location="http://127.0.0.1:8080/hnStreamStart?live=ocap://0x2c23&continueTSB=true" blocksize=131072 ! dtcpdec dtcp-
src-ip="127.0.0.1" dtcp-port=5000 buffersize=131072 ! playersinkbin is-live=true.`

- All include header files of a component are present in <https://code.rdkcentral.com>.
- After login into <https://code.rdkcentral.com> search for the respective component and go to the component repo which shows all the corresponding files.
- The coredump files will be available in following directory for a systemd based box - `/var/lib/systemd/coredump/`
- Additionally for system logs, you can browse through following files in `/opt/logs` –
 1. `core_log.txt`
 2. `rebootInfo.log`
 3. `messages.txt`
- Previous boot Logs can be also seen from `/opt/logs/PreviousLogs/`

There is no package manager support for downloading/installing packages. However you can manually install ipk packages.

RDK Device Settings is the component which handles the following configurations:

- Audio Output Ports (Volume, Mute, etc.)
- Video Output Ports (Resolutions, Aspect Ratio, etc.)
- Front Panel Indicators
- Zoom Settings
- Display (Aspect Ratio, EDID data etc.)
- General Host configuration (Power managements, event management etc.)

These properties are persisted in the STB and are read/applied on each boot-up.

For example: On a RDK emulator, the device setting properties are persisted in `'/opt/persistent/ds/hostData'` .

There are few sample applications available to test/force the settings e.g. `setResolution` can be used to force the resolution settings.

The Comcast RDK requires kernel logs, DOCSIS ECM logs, and syslog messages to be logged into specific files. These details can be provided by Comcast on request. The RDK set-top diagnostics and troubleshooting infrastructure requires these logs to be present and accessible via the ESTB interface to aid in the rapid troubleshooting of the device during development and deployment. Flash and hard disk drive (HDD) requirements for RDK devices will differ depending on the deployment configuration. For example, a DVR will have an HDD but some other devices will not.

Flash is used to store the advanced bootloader (ABL), primary and secondary firmware images, serialization data, and other data that need to persist, including logs. Where possible on HDD devices, the HDD is leveraged to store dynamic content leaving flash on those devices to be primarily read-only. The partitioning scheme for on-board flash and HDD will therefore differ depending on the physical makeup of the device.

Please contact Comcast for more information.

libcrypto, openssl and libpgp can be ported as it is. However the SoC security APIs need to be implemented to support Comcast security requirements.

There are no requirements for hardware optimization.

RDK-V UI

- Connect your device with lan and get the box/device ip, make sure your dev pc and device is connected with the same network, so that you will be able to access your device from dev pc.
- From dev pc: \$ scp -r <dist or build folder which contains index.html, appBundle.js and other files> root@<your box/device ip>:/home/root>
- Login into box: \$ ssh root@<your box ip>
- Go to dir: \$ cd /lib/rdk
- Edit residentApp.sh: \$ vi residentApp.sh
- You will see Residentapp = "https://rdkwiki.com/rdk-apps/accelerator-home-ui/index.html#splash" and offlineApp="http://127.0.0.1:50050/lxresui/index.html"

Execute \$ npm run watch and then npm run serve separately in separate terminal.(tried on VS code)

For this, Install windows node env globally:

```
$ npm install -g win-node-env
```

RDK-B

- The rootfs doesn't contain [libpcap.so](#), as tcpdump utility is not appended in the packagegroup-rdk-oss-broadband.bbappend file in the Raspberry-pi build whereas you can find the tcpdump utility appended for the corresponding file for Emulator build.
- So, if we include tcpdump utility in the above mentioned file, libpcap. So file will be present in the rootfs of the raspberry-pi build.
- See below output from the raspberry-pi device after adding tcpdump -

```
root@RaspberryPi-Gateway:/# find . -iname libpcap* ./usr/lib/libpcap.so.1.7.4 ./usr/lib/libpcap.so.1
```

Currently no such pre-built images are available for download. You can follow the simple build instructions to generate your own build.

Eg:

- [RDK-B Raspberrypi - Host SetUp and Build Instructions](#)
- [RDK-B Emulator Build Instructions](#)

The eRT (embedded router) subsystem is generally used to perform dmcli operations on Raspberry pi which can function as a basic router. To perform dmcli operations in emulator we use simu.

We're simply building the application using go build (without any Yocto recipes) as shown below -

```
env CC=arm-linux-gnueabi-hf-gcc LD=arm-linux-gnueabi-hf-ld GOOS=linux GOARCH=arm GOARM=7 CGO_ENABLED=1 go build
```

- RDK officially doesn't support GO language compatibility. But you may always try GO binaries as most RDKM code bases are based on Yocto open-embedded builds and the binaries work out of the box.
- You can simply build the application using go build (without any Yocto recipes) as shown below -
 - `env CC=arm-linux-gnueabi-hf-gcc LD=arm-linux-gnueabi-hf-ld GOOS=linux GOARCH=arm GOARM=7 CGO_ENABLED=1 go build`
- Please refer to <https://code.rdkcentral.com/r/plugins/gitiles/rdkb/components/opensource/ccsp/CcspWifiAgent/+refs/heads/rdk-next/config-atom/TR181-WiFi-USGv2.XML>. This gives objects, parameters list for WiFi along with functions call. The XML file (TR181-WiFi-USGv2.XML) file provides details regarding the data models available under Device.WiFi.
- In older releases, the XML files are available under /usr/ccsp/<component>. But with the latest code, instead of reading and parsing the XML at runtime, the new approach includes conversion of XML to .C.cpp with the help of XML2C and creating a shared library ([libWi-Fi.so](#)). These changes are done for rdkb to reduce image size and improve boot times.

Please refer <https://wiki.rdkcentral.com/display/RDK/RDKB+Containerization+in+RPI+-+User+Manual+-+2019+M3>.

? Unknown Attachment

Below are the two ways to enable debug logs for component

1. adding prints in the respective place of code

2. using rdkb logger(LogAgent) .
 LoggerEnable - to enable/disable logs for particular component. If it is TRUE logs are enabled otherwise logs are disabled.
 LogLevels - To set different log levels for each component. By default all modules log level is 4 (RDK_LOG_INFO).

The steps are –

- Port valgrind to the platform (if not already available)
- Compile the component with debug symbols enabled (i.e, avoid stripping the component executable/library. alternatively, you may replace the stripped binary with unstripped binary if it is not a read-only image)
- Stop the service and start the binary as part of valgrind (just like we load any normal binary using valgrind)
- Once you have enough data collected (or observed the issue you were trying to debug), you may stop it and examine the xml file for details.

For reference please look into this attachment :

? Unknown Attachment

- ipset is available in real RDK-B devices(like the XB6 devices for example). To enable ipset for RDKB emulator, you can do that as the yocto recipes for ipset are already available at /meta-openembedded/meta-networking/recipes-extended/ipset/38.bb
- You need to add the recipe to the package group for emulator to use it up during build time (you may add them in ./meta-rdk-bsp-emulator/recipes-core/packagegroups/packagegroup-rdk-ccsp-broadband.bbappend)

Follow the below steps to customize kernel –

- Need to Create config file within linux bb/bbappend file available directory
 Path: meta-raspberrypi/recipes-kernel/linux/linux-raspberrypi
 (XXX.cfg) - Example:camera.cfg
- Configuration parameter addition: Add config data to created config file so as to enable it. For example, add the below line to our camera.cfg file
 CONFIG_VIDEO_BCM2835=y
- Need to add created config file under SRC_URI of linux bb/bbappend file mentioned in step 1
 SRC_URI = "file://camera.cfg"
- Do compilation and Installation
 bitbake "component-name" -c compile -f
 bitbake "component-name" -f

Yes, igmp_snooper program is present in RDK-B .

- There is a shell script named "service_mcast Snooper.sh" under /etc/utopia/service.d directory, and in the script, it will execute "igmp_snooper \$sw_opt \$if_opt \$querier_opt

The data model can be managed via TR-069 , TR-181, management protocols like SNMP , WebUI, WEBPA .

- The eCM and eRT subsystems will always coexists in a **DOCSIS based WAN frontend gateway devices**. Any request for eCM parameters will be routed to eCM Message Bus Adapter component which uses **SNMP or other protocols** to talk to the cable modem firmware in the backend.
- In Rpi, we have only eRT and in emulator we have simu.
- eCM is applicable only to DOCSIS based devices , which uses SNMP based OIDs.

Use NTP sync to run in the background to sync time with NTP servers.Update the NTP servers in ntp conf file if needed.

Minimum Hardware requirement :

- A Raspberry Pi 3B with the below configuration can be an ideal minimum requirement to support most of the functionalities for RDK-B.
- The configuration of RPi 3B is CPU: Quadcore 1.2 GHz Broadcom BCM2837 64bit with 1GB RAM.
- R-Pi 3B would be sufficient for the basic wireless functionality using 2.4GHz. In case of 5 GHz band support R-Pi 3B+ would be good (Note : R-Pi 3B+ doesn't support simultaneous dual band operations).
- Flash: The minimum size is around 285 MB. Anything above this should work just fine for loading/storing the build.
- RAM: As far as the logs/database is concerned, anything around 16 MB should be sufficient.

The above requirement supports all the basic functionalities of RDK-B.

- RPC download is not supported in RDK-B.
- RDK-B never had "cisco.spvtg.ccsf.fu.Configuration". This part of the code has come through initial code drop and is deprecated.

Yes. RDKB has the below options to modify SSID name, password, security protocol etc.

- A WebUI, where you can modify these items just like the UI for most normal routers.
- 'dmcli' command line utility which will execute TR69 commands to modify the mentioned items.

Yes. You can invoke the data models via dmcli commands from your service or via Cdm_GetParam set of APIs from your code. The TR-181 data model specifications that are available in the TR-181 data model XML for each component can be referred.

The CA certificate file and device certificate/private key can be configured in ccsp_tr069_pa_cfg.xml for https support.

There are already reserve SSID for Guest WiFi network. Just enable the SSID using the respective DM to get the features in place.

Duktape is a lightweight javascript engine (<https://duktape.org/>). Alone it does not provide a web development engine like php or node.js. Therefore, on top of duktape there is a templating engine(called jst) and web server api. To make migration as easy as possible the style of templating and api signature will match php as closely as possible. Many php functions and variables will be rewritten in javascript, so that changes to the exist code is minimized.

It actually defines what kind of Internet traffic is allowed or blocked. For more details, Please Refer : [Firewall - Rule persistence](#)

Please refer [Integration Guide for third-party applications into RDK-B stack](#) .
Use journalctl -xu parodus or can look for parodus.log in the log folder i.e. /rdklogs/logs/.
In <image>.bbappend file , upgrade python version using ROOTFS_POSTPROCESS_COMMAND
Code:

```
#python upgrade
IMAGE_INSTALL_append = " python3 "
ROOTFS_POSTPROCESS_COMMAND += "enable_python3; "
enable_python3() {
ln -sf /usr/bin/python3 ${IMAGE_ROOTFS}${bindir}/python
}
```

On compiling the code and flashing the image , python3 will be available in your board . Check with below command.

```
root@RaspberryPi-Gateway:~# python --version
Python 3.5.2
```

Checksum mismatch error is related to Fetcher failure for URL . The ways to fix this error are as follows

1. the url might be deprecated . Hence should use the right URL with right path
 2. md5 checksum must be pointing to older one , which should be replaced with the latest checksum
 3. if the component is not required (or not maintained) , then they can be masked from inc file
- For this you need to convert all C/C++ logging statements(like printf) to use standard logging functions CcspTraceInfo, CcspTraceWarning, CcspTraceError etc. CcspTraceXxxx functions are based on open-source 'log4c'.
 - By using the above apis the logs are automatically written to logfiles.
 - After conversion to standard logging functions ,you can verify the logs under /rdklogs/logs/.

RDK-Common

yes, all logs are maintained in persistent path (opt/logs) log rotation available if size goes beyond. Log server are in place. [More details on how logging is done in RDK](#) can be found here: [Log Upload, RDK Logger](#)

Telemetry

Yes, Because telemetry itself is a subset of Device Configuration management aka DCM and we need to configure the DCM logupload settings before setting up telemetry configuration

By using T2 Report profiles, we can setup multiple telemetry profiles. Each profile can have different set of telemetry markers , telemetry upload location URL and telemetry upload schedule. The different telemetry JSON data will be uploaded separately. Thus multiple telemetry profiles is achieved.

Yes, the T2 report profile will have the upload URL along with the telemetry markers, The upload repository in different profiles can be same or different. So based on the configured URL in these T2 report profiles, the data will be uploaded to the respective URLs .

RDK-Cloud

To know the Complete steps for xconf local setup , Please refer : [Xconf Server - User guide for configuration and feature validation](#)

In service.properties, you can add the following properties. The "seeds" property is just a comma-separated list of a few Cassandra hosts. If you are using a single host, then it's just that one. The other properties may not be needed.

```
connectionPools.SharedConnectionPool.seeds=192.168.0.100,192.168.0.101
connectionPools.SharedConnectionPool.localDatacenter=DC_NAME
connectionPools.SharedConnectionPool.readConsistencyLevel=CL_LOCAL_QUORUM
connectionPools.SharedConnectionPool.writeConsistencyLevel=CL_LOCAL_QUORUM
connectionPools.SharedConnectionPool.autoDiscoverHosts=true
```

Yes, it can be run on separate nodes. The admin and data service do not need to discover each other. They both use the Cassandra database, so they need to be able to reach it. If you install the admin and data service on separate nodes, you will probably have Cassandra on a third node. The service.properties file for each service should include the "seeds" property.

Steps -

1. Go to the Firmware menu and select Download Location Filter.
2. Select Edit.
3. Enter the FQDN for the firmware download server.
4. Enter the full http location for the firmware download server.
5. Enter an IP address for TFTP and enter 100%. (This download method will not be used.)
6. Select Save.

Refer [Xconf Server - User guide for configuration and feature validation#FeatureValidation\(Firmwareupdate\)](#) for more details.

There should be a webpa server running somewhere (to which the device can contact). If the server is in place, you can run the commands to fetch data models from a PC that can connect to server or even from the RPi terminal itself.

- As such they don't have any connection between them.
- While xconf is for pushing device specific configurations to CPE, webPA is for remote device management.
- You can refer the [xconf server section in wiki](#) and [WebPA](#) for more details.

As of now we don't have a provision to update the webPA url from xconf.
This is setting a parameter using webPA

- Setting the SSID Password :

```
curl -X PATCH http://35.155.171.121:9003/api/v2/device/mac:b827eb5681cd/config -d '{"parameters": [{"dataType": 0, "name": "Device.WiFi.SSID.10001.SSID", "value": "Testing"}]}' -H 'Authorization:Basic d2VicGFAMTlzNDU2Nzg5MAo='
```

- Getting the SSID for the board.

```
curl -X GET 'http://35.155.171.121:9003/api/v2/device/mac:b827eb5681cd/config?names=Device.WiFi.SSID.10001.SSID' -H 'Authorization:Basic d2VicGFAMTlzNDU2Nzg5MAo='
```

You can delete a model using below command

- \$ curl -i -X DELETE http://xconf_server_ip:9092/delete/models/XYZ123

Retrieve list of models & check the particular entry is deleted:

- \$ curl -i http://xconf_server_ip:9092/queries/models

.ns list for mac list

Themis was not part of the community XMidT server, we will discuss with developer and update the wiki.

- WebPA is the communication channel from Cloud to RDK based home gateway devices. It helps to manage devices from Cloud. It was built from the ground up specifically with security and performance as priorities.
- WEBPA protocol provides functionality of read/write access to device management parameters.
- The "PA" in WebPA stands for "Protocol Adapter." It's a component used to connect devices to clouds in a way that RDK licensees call "trivially scalable."

Refer [WebPA](#) for more information.

Refer Environment Setup in [RDK-C : WebPA Support#C:WebPASupport-EnvironmentSetup](#)

TR-69 polls wide-and-deep across a device landscape, on a less frequent basis whereas WebPA can precision-poll for the most useful data, much more quickly. That's mainly because it's lightweight, and because the load can be redistributed into all the apps needing to access the data.

Example 1: Fetch device or feature parameter detail from client(RPI) device through parodus by using webpa server.

- curl -H 'Authorization:Basic dXNlcjB3ZWJwYQo=' -i 'http://192.168.2.75:9003/api/v2/device/mac:b827eb2e722b/config?names=Device.DeviceInfo.X_RDKCENTRAL-COM_IMAGENAME'

Example 2: Setting the SSID Password :

- curl -X PATCH <http://35.155.171.121:9003/api/v2/device/mac:b827eb5681cd/config> -d '{"parameters": [{"dataType": 0, "name": "Device.WiFi.SSID.10001.SSID", "value": "Testing"}]}' -H 'Authorization:Basic d2VicGFAMTlzNDU2Nzg5MAo='

Example 3: Getting the SSID for the board.

- curl -X GET 'http://35.155.171.121:9003/api/v2/device/mac:b827eb5681cd/config?names=Device.WiFi.SSID.10001.SSID' -H 'Authorization:Basic d2VicGFAMTlzNDU2Nzg5MAo='

Refer WEBPA Validation Procedure Steps in [RDK-C : WebPA Support#C:WebPASupport-WEBPAValidationProcedure](#) for more information.

Yes ,it is possible to configure TFTP ipv4 and ipv6 locations in Xconf. Follow the steps specified here [Xconf Server - User guide for configuration and feature validation#AddTFTPLocation](#)

Xconf doesn't store any data, be it logs or telemetry data. It only provides the configuration to the client device regarding where to upload the data, when to upload the data and what data it needs to be uploaded. So Xconf server doesn't need any big data processing tools.

Xconf

There is no direct interaction, both Xconf and WebPA serve different purpose. While Xconf is used by devices to fetch and configure the rules, webPA can be used to push/set the rules/attributes on a CPE device.

No, xconf doesn't directly facilitate upload logs/telemetry data. It provides devices information on where/when to upload the log files or telemetry json files.

No, the maven-frontend-plugin plugin downloads/installs Node and NPM locally for your project, runs npm install. It's supposed to work on Windows, OS X and Linux.

Yes, If anyone needs to override the firmware download location, it's best to do it with a define property rule. It will only affect those devices identified in the rule.

Just like tftp location is overridden like this - [Xconf Server - User guide for configuration and feature validation#AddTFTPLocation](#)

Miscellaneous

The roadmap is always available to the Preferred and Preferred Plus members of RDK at <https://wiki.rdkcentral.com/display/ASP/Roadmap> . To become a preferred member as well as to know the benefits, please visit <https://rdkcentral.com/memberships/> . For the non preferred members, you can get answers to specific queries on Roadmap items by contacting support@rdkcentral.com or asking at <https://wiki.rdkcentral.com/display/FORUMS/FORUMS+Home> .

Please refer [Breakpad steps](#)

On 15 August 2013 Comcast Cable and Time Warner Cable [jointly announced](#) the formation of RDK Management, LLC to fulfill the role of the "RDK Entity".

We will maintain compatibility with the previous RDK version as described by the guidelines in the agreement. Additional details will be posted on the RDK Wiki as they become available.

Licensee are not required to take a new baseline.

RDKM does not assess run-time royalties for the Generic RDK.

The infrastructure to support DVB exists today; but additional components may need to be developed to support the particular network type. Typical enhancements include converting a URI to a network-specific identifier in the SI handling, and enhancing the security processor to implement CA that doesn't use the cable card.

RDK supports CEA-708 and EIA-608.

The font-type used for Closed Captioning belongs to an opensource family called Droid with specific glyphs for closed captions. In case operators are using licensed fonts they have to meet the exclusive license requirements for those fonts.

OEM vendors will need to work with font vendors to provide fonts that comply with FCC standards. RDK Support is looking into whether FCC-compliant open-source fonts are available, and will provide that information in a future update.

To add VOD support for a different system, the VOD client would be rewritten to talk to the appropriate backend to provide content. This is an independent implementation and not coupled to any specific MSO configuration. The client is responsible for tuning to the content, connecting it to a sink, and starting streaming. It comes into play only after a VOD request is received by the receiver, the purchase is completed, and the connection is created (i.e., broadcast has started from servers side). Because time shift buffers are not allowed for VOD content, trickplay commands are merely forwarded to the content source, which is responsible for taking the appropriate actions.

Yes, RDK supports these tags in the current version.

Device Settings is integrated with IARM, which invokes the HAL API to effect changes requested by the application. Interface specifications are available for application developers.

The IP client provided by RDK works with the RDK Server, not the general DMS. This is the direction for now. CVP-1 and CVP-2 certification are planned for RDK Servers.

Separate RDK release packages will be available for IP and Hybrid versions. The requirements will draw a clear distinction between IP and Hybrid components

Control of front-panel brightness and other front-panel controls such as LED displays, power-button and similar controls will be specified as part of the Comcast Manufacturer Interface Specification.

The answer can be "extsrc" is used to facilitate easy development work. If it is preferred otherwise, there is a XXX-nosrc.xml manifest file which will be retained after the first build and hence saving time

Many developments & improvements happening in RDK .Few are listed below.

- Doxygen : Video & Broadband documentations are available in the following locations.
 - For Broadband: <https://wiki.rdkcentral.com/doxygen/rdkb/index.html>
 - For Video: <https://wiki.rdkcentral.com/doxygen/rdkv-opensourced/index.html>
- Github : <https://github.com/rdkcentral/>
- Systemd

For latest updates see Announcements at <https://wiki.rdkcentral.com/> page and " [What is the Roadmap for the RDK?](#)" question in this page to know about the Roadmap of RDK.

1. git commit your changes in both the local repos
2. Create a branch and check it out in both repos make sure it has the SAME NAME (i.e. git checkout -b my_special_changes)
3. Then finally do a git review in both repos (or more specifically git review master or git review stable2)

What this does:

In gerrit the changes will be put up for review in the branch you specified in your git review command (master or stable2). At the same time it will set the Topic field to the name of your local branch (i.e. my_special_changes). When gerrit sees topic fields with identical names it binds them together. So it forces project maintainers to press the "Submit Together" button so they go in at the same time once given +2.

Install SDK if it is not already installed by following the steps under the previous heading. Install Groovy version 2.0.7 / 2.0.6 (until M75) using the below command

- `sdk install groovy 2.0.6`

Check groovy version :

- `groovy -version`

From M76 release onwards, the supported groovy version is groovy 2.3.0. Install SDK if it is not already installed by following the steps under the previous heading. Install Groovy version 2.3.0 using the below command –

- `sdk install groovy 2.3.0`

Check groovy version :

- `groovy -version`

Follow are the commands to install MySQL

- `sudo apt-get install mysql-server`

mysql installation will ask for the password during installation. (give the root password)

If validate_password plugin is present for Mysql, set the password policy as LOW using the below command

- set GLOBAL validate_password_policy=LOW; flush privileges";

Login to mysql prompt

- sudo mysql -u root -p

It will ask for password, give the root password given during installation.

There is no size restrictions imposed by RDK as such, the limits depends on SoC and OEM factors including Flash size. The firmware image will be compressed using SquashFS with LZMA compression.

No advanced options are currently supported by the IARM. It is mainly an event bus with a pub/sub metaphor. A lot of what is available is driven by the need. If some of the features are needed we should see a feature request. We are open to discussion of improvement as long as there is a case for it. No metrics have been captured at this time outside of the overall cloud metrics where IARM is part of the overall latency.

The various components in RDK have dependencies on other components, and some of these have similar functions. As long as the claims on memory and other system resources is small (e.g., Mongoose is a single file) it may be expedient to retain these redundant components.

It is optional and some devices is known to have HDMI 2.0 such as accelerator.

Checkout relevant code repos based on your requirements from <https://code.rdkcentral.com/r/admin/repos>

1. For Open sourced components, search for required component in the filter option to get the relevant code repo.
2. For Licensed components, sign in to <https://code.rdkcentral.com/r/admin/repos> using username and password, and search for required component in the filter option to get the relevant code repo.

Help & Support

Still not able to find what you are looking for? Please reach out to RDK Support by sending an email to support@rdkcentral.com .