Upgrading RCE Core Software

The RCE core software is updated over the network using the SDK and its utilities.

The updates are made available to RCEs using two methods:

1. DTM based NFS server
2. Host based NFS server

Each RCE is externally commanded to download and apply the updates.

Note that the SDK release contains all files to be uploaded.

IMPORTANT!

Test update on a single RCE prior to updating an entire COB or shelf.

Make sure the update completes without error and the RCE successfully boots.

DTMs control the network connection for DPMs. Do not simultaneously broadcast an update to DTMs and DPMs.

If a DTM updates or reboots while a DPM is updating, the network connection will be lost, and the update will fail.

If you are supplying your own fpga.bit, you must put your file in the SDK directory.

If you do not want to update fpga.bit, remove the fpga.bit contained in the SDK before upgrading.

```
arm-linux-rceCA9/tgt/boot/dpm/fpga.bit
```

```
arm-linux-rceCA9/tgt/boot/dtm/fpga.bit
```

To perform the upgrade:

Download and install the latest RCE SDK.

Choose a COB to update.

Ensure all RCEs in COB have been freshly rebooted to the target OS.

Chose an update server method.

Update Server Method 1: Upload RCE core software to DTM

First cd to the SDK directory and prepare the DTM by executing the uploadSDK command.

For example, prepare DTM ID shasta/4/0/0 for the arm-linux RCEs using network interface eth1.204:

```
i86-linux-64/tools/uploadSDK arm-linux-rceCA9 shasta/4/4/0 eth1.204
Looking up shasta/2/4/0...
...Got it: 192.168.204.253
Uploading arm-linux-rceCA9 to shasta/4/4/0
Configuring SDK update on shasta/4/4/0
```

Update Server Method 2: Copy SDK to host NFS directory

Make sure the entire SDK directory is read accessible to the shelf network via NFS.

Ensure that the root path to the SDK is less than 32 characters.

If necessary, create a soft link in NFS space to the SDK directory:

```
i.e. ln -s /nfsexport/project/RCE/sdk/V1.10.0/arm-linux-rceCA9 /nfsexport/project/RCE/sdk
```

Command one or more RCEs to apply the update using the dsl_update command.

For example, upgrade RCE shasta/4/3/2 from DTM shasta/4/0/0 using interface eth1.204:

```
dsl_update --ip `atca_ip shasta/4/4/0 --ifname eth1.204` --ifname eth1.204 shasta/4/3/2
```

In this example, upgrade all shasta/2 DPMs using a host NFS exported SDK on interface eth1.204:

```
dsl_update --ip 192.168.204.12 --src /nfsexport/project/RCE/sdk --sm shasta-sm --ifname eth1.204 shasta/2/DPM
```

Where 192.168.204.12 is the IP address of the NFS server on the shelf network,

/nfsexport/project/RCE/sdk is a soft link to the NFS exported SDK directory,

and shasta-sm is the IP address/hostname of the shelf manager.

(For SDK versions V1.10.0 and up, this additional dsl_update argument is used to monitor the update status)

```
--sm <shelf_manager_ip>
```

(For SDK versions V1.5.0 or less, add these additional dsl_update arguments are required for server method 1)

```
--src /srv/nfs4/dsl --file tools/update-target.sh --tmo 4000
```

Wait 60 to 90 seconds for the upgrade to complete and for the RCE to reboot. (output shown for V1.10.0 and up using --sm switch)

```
Slot 4 : Had Responses
 RCE 3/2 : OS: RTEMS (ID:0x00000000 STATUS:0x0)

Waiting for update completion.....................
Slot 4 : Update complete!
```

Check RCE network connectivity using dsl_identify to ensure the upgrade was successful.

For example:

```
dsl_identify shasta/4/3/2 --ifname eth1.204
Slot 4 : Had Responses
 RCE 3/2 : OS: LINUX IP: 192.168.204.21 ID: 0x00000000 V1.10.0



Introduction
=============

Starting with V1.0.0, all RCE SDKs and corresponding SD images include tools
to update the core OS and system files residing on the target filesystem.

The source of all system updates are the files residing in the /tgt directory of
arm-rtems-rceCA9 or arm-linux-rceCA9 in the SDK.

Update Methods
===============

Software and firwmare updates to the RCE target are performed over the network
by copying files from a host nfs server to the target SD flash.

For host systems that do not support nfs servers, the RCE DTM can be used as
the nfs server. In this scenario, the updates are first secure copied (scp)
from the host to the DTM. The DTM then serves the updates to any RCE using
its internal nfs server.

Only one OS platform may be updated at a time. The RCE must be booted into the
appropriate OS prior to initiating the update.

Update Process
===============

Updating is initiated on the host by running the dsl_update executable.
The executable issues an update command that instructs RCEs to:

1. mount an nfs directory from the nfs server
2. execute a shell script from the nfs directory

The update shell script instructs RCEs to:

1. mount appropriate filesystem partitions
2. pull (copy) files from the nfs directory to its local filesystem
3. record SDK update version and identifier
4. reboot the RCE upon completion

Preparing the SDK
==================

RTEMS
------
The system update will copy __all__ files from the SDK /tgt/rtemsapp/config
directory to the RCE.

If you are using a modified app.svt and/or appinit.so, you have 2 choices:

1. Remove the default app.svt and appinit.so from the SDK /tgt/rtemsapp/config
directory prior to updating. You are then responsible for updating those files.

2. Place your modified app.svt and appinit.so in the SDK /tgt/rtemsapp/config
directory prior to updating. The system update will copy them to the RCE for
you.
```

RTEMS and LINUX

The system update will copy __all__ files from the SDK /tgt/boot/dpm directory
to the RCE.

If you are using a non-default firmware bitfile, have 2 choices:

1. Remove the default fpga.bit from the SDK /tgt/boot/dpm directory prior to
updating. You are then responsible for updating the bitfile as fpga.bit on the
RCE.

2. Place your custom firmware bitfile in the SDK as /tgt/boot/dpm/fpga.bit
directory prior to updating. The system update will copy them to the RCE for
you.

Preparing the Host NFS Server
==========

For hosts supporting nfs servers, the SDK platform directory must be mounted
and prepared for access by RCEs on the network.
Note that this requires sudo priveleges.

Create nfs server directory

```
mkdir -p /srv/nfs4/dsl
chmod -R a+r /srv/nfs4
```

Add directory to /etc/exports

```
/srv/nfs4 192.168.204.0/24(rw,no_subtree_check,all_squash,anongid=2660,insecure)
/srv/nfs4/dsl 192.168.204.0/24(rw,no_subtree_check,all_squash,anongid=2660,insecure)
```

Note: replace 192.168.204 with your subnet

Mount the SDK platform directory

```
mount --bind <path_to_sdk>/<platform> /srv/nfs4/dsl
```

i.e. mount --bind /opt/RCE_SDK/V1.0.0/arm-rtems-rceCA9 /srv/nfs4/dsl

Refresh nfs server

```
/sbin/service nfs restart
```

Preparing the DTM NFS Server
==========

A utility script is provided in the i86-linux-64 directory which uploads the updates and
prepares the nfs server on the DTM.

i.e. i86-linux-64/tools/uploadSDK /opt/RCE_SDK/V1.0.0/arm-rtems-rceCA9 <shelf/slot/bay/rce> [<ifname>]

uploadSDK

```
usage: i86-linux-64/tools/uploadSDK <rce_sdk> <shelf/slot/bay/rce> [<ifname>]
 Configures the RCE update server with an SDK package

params: <shelf/slot/bay/rce> The location of the RCE in ATCA coords
 <rce_sdk> The full path of the RCE SDK
 <ifname> The first interface to check for the RCE (optional)
```

Launching the Update
==========

Now that the nfs server is configured, the update command can be issued to one or more RCES.

EXTREME WARNING!!!!
==========
DO NOT POWER CYCLE OR REBOOT RCES AFTER LAUNCHING UPDATE!!!!!
This will cause severe damage to the RCE filesystem and require manually removing the SD card.
RCEs will reboot once the update has finished.

RCE single update example:

```
dsl_update --ip 192.168.204.1 --src /srv/nfs4/dsl --file tools/update-target.sh
 --ifname em2 --id 0x12345678 --tmo 8000
 egbert/1/0/0
```

RCE DPM update example:

```
dsl_update --ip 192.168.204.1 --src /srv/nfs4/dsl --file tools/update-target.sh
 --ifname em2 --id 0x12345678 --tmo 8000
 egbert/1/0
```

RCE COB update example (excludes DTM):

```
dsl_update --ip 192.168.204.1 --src /srv/nfs4/dsl --file tools/update-target.sh
 --ifname em2 --id 0x12345678 --tmo 8000
 egbert/1/DPM
```

dsl_update
==========
```
usage: dsl_update --ip ip --src src --file file [options] location

-v Verbose output: report things which don't respond.
 --ip IP address of update server
 --sm Hostname or IP address of shelf manager
 --src Path of update source directory on server (max 31 chars)
 --file Path of update file relative to source directory (max 31 chars)
 --ifname The name of a NIC which can see the shelf.
 --id 32-bit identifier for update transaction
 --tmo Response timeout in milliseconds
 location An RCE location. May be wildcarded

Broadcast 'update yourself' packet to a network. All running
 RCEs matching the wildcarded location on that network will
 update target filesystem by executing ip:src/file.
```

Checking the Update Status (V1.10.0 and up)
==========

The dsl_update command will monitor the status of the update using the shelf manager.

To enable this, add the --sm <shelf_manager_ip> argument to the dsl_update command.

Checking the Update Status (V1.0.0 and up)
==========

The time for an RCE to complete an update can vary depending on the SDK content
You can check for completion of the update using dsl_identify.
The dsl_identify command will query RCEs for their current operating status.

While the update is in progress, a response should be received showing the
current version information.

While the RCE is rebooting, no response will be received.

Once the update is complete, the received response will have the updated
version information.

RCE single status:

```
dsl_identify egbert/1/0/0 --ifname em2

Slot 1 : Had Responses
 RCE 0/0 : OS: RTEMS IP: 192.168.204.137 ID: 0x00000000 V1.0.0
```

RCE COB status:

This will dump the stats for all RCEs in the COB.

```
dsl_identify egbert/1 --ifname em2
```

Reboot Commands
==========
To make your custom bitfile the default at boot, execute the appropriate
reboot command with the -b flag set to 1. All subsequent boots will now
load your fpga.bit.

RTEMS:

```
reboot -t rtems -b 1
```

LINUX:

```
reboot_rce -t linux -b 1
```