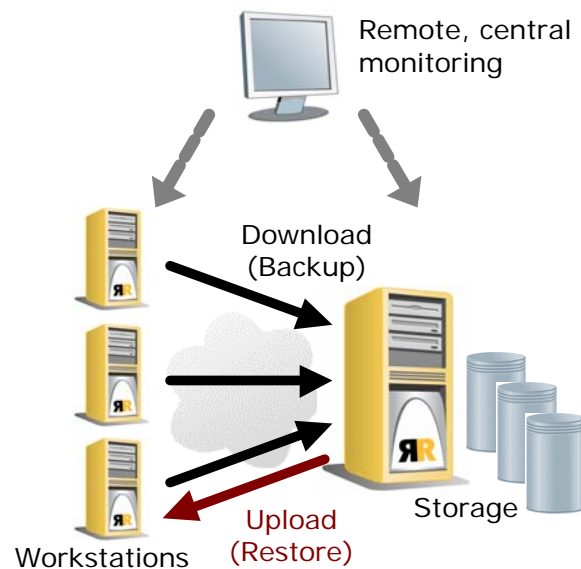


Centralized Disaster Recovery using RDS™

RDS™ is a cross-platform, scheduled replication application. Using RDS's replication and scheduling capabilities, a Centralized Disaster Recovery model may be used to handle backup and restore situations.

The solution is based on a scheduled replication to an offsite server, that can be monitored from remote locations. RDS provides asynchronous and unattended process, between heterogeneous operating systems (UNIX, Windows). Replication is guaranteed to be accurate and fast, supporting block level updates and preserving permissions, and at the same time allowing control over bandwidth usage.



1. The Nature of the Problem

Employees working on their individual workstations need a daily, solid backup solution assuring data integrity and recovery in case of need. Administration overhead can become unmanageable in a constantly changing environment.

The various ways to backup individual workstations content are lacking:

- **Manual Backup** - This relies on individuals running a script on their workstation everyday, which copies the files that have changed since it was last run onto a server. This server could then be saved to tape every night. The problems in that kind of solution are: people tend to forget to run the script, it is difficult to accurately check all workstations have been backed up, and inconsistency as to

what time of day the backup is run. Automatic scheduling of such a script still requires that each workstation be individually configured.

- **Tape Backup** - Specified areas of the workstations can be backed-up into a tape every night via the network. The problem with this approach is that each workstation is done in sequence. If one workstation runs very slowly it will delay the backup of the next in the list, so it would be quite possible not to have completed the backup by the time everyone starts work the following day. This solution cannot be used on daily backups where there is a small window in time and large amounts of data to backup. The number of backup tapes required becomes very large. Using incremental backups would significantly increase the work when restoring files.
- **Networked Drive** - Create an area on the server, one for each PC, and connect them to the network at high-speed (100mbs), to be used just as if it was a local hard drive. The server would then be backed up every night, providing centralized control, and speed of the PC wouldn't slow the backup down. The downside to this is a major reduction in performance, as the network is never as fast as working locally on the PC. The cost of upgrading the network to allow fast connection to all workstations is very high.

2. The Solution - Directory Replication

Using RDS, the working directories on each workstation can be mirrored to an area on the server. Work on the workstation takes place on the local drives, and is mirrored to the server at specified times of the day, after working hours. It allows users to restore files deleted by accident, without having to wait for IT.

The server can either be on the organization's LAN or anywhere else on the WAN.

The server can then be backed up to a backup-tape on a scheduled basis (weekly, monthly) as required, using standard tape backup tools.

RDS offers centralized configuration, control and detailed information about the number of files copied. It allows all workstations to replicate to the server at the same time, preventing slow workstations from affect the whole backup process. It allows control over bandwidth usage and replication of only the differences between the source and target machines, to reduce network resources.

Reduced administration offers an indirect reduction in cost, especially when extended to the remote sites, as RDS offers centralized monitoring from any place on the network.

As RDS can preserve permissions, individual employees will not be able to view or change other employees' content on the server, maintaining security within the organization.

RDS is a cross-platform solution, working seamlessly between Windows and Unix platforms, allowing the workstations and the server to run varying operating systems.

3. Solution Setup

Define scheduled jobs from each workstation to the central server. Every scheduled job defined will have a different target directory, corresponding to the source workstation. That way, at every moment each employee can restore working files, and there is no risk of one employee's content overriding another's.

Once a restore or a "roll-back" is required, define a job in the reverse direction, restoring data from the appropriate backup tree, for the specific workstation.

In our example, a backup job will be run at the end of each working day at 9:00pm, replicating data from each workstation to the centralized server.

1. Define a Download job, replicating the working directory from each workstation to a unique location on the server.

Depending on your organization conventions, it may be easier if all workstations use the same working directory.

Download Definition

Name: Workstation 001
Description: Daily Backup

Schedule when you want the job to run. Options include once, continuously, at regular intervals, trigger file, daily or weekly. **Schedule...**

Set the job properties including synchronization type, transfer engine type, performance, snapshot settings, pre/post commands and permissions. **Properties...**

Replication Source

Satellite: workstation001 Platform: Windows

User: james Password: ***** Domain: compdomain

Source Directory: c:\daily_content **Browse...**

Replication target on production_srv (Controller)

Target Directory: e:\workstation001 **Browse...**

Defaults **Template...** **Submit** **Cancel** **Help**

RDS Console GUI – Download Job Definition

2. Schedule the RDS jobs to backup the source directory tree from each workstation to the target directory on the centralized server.

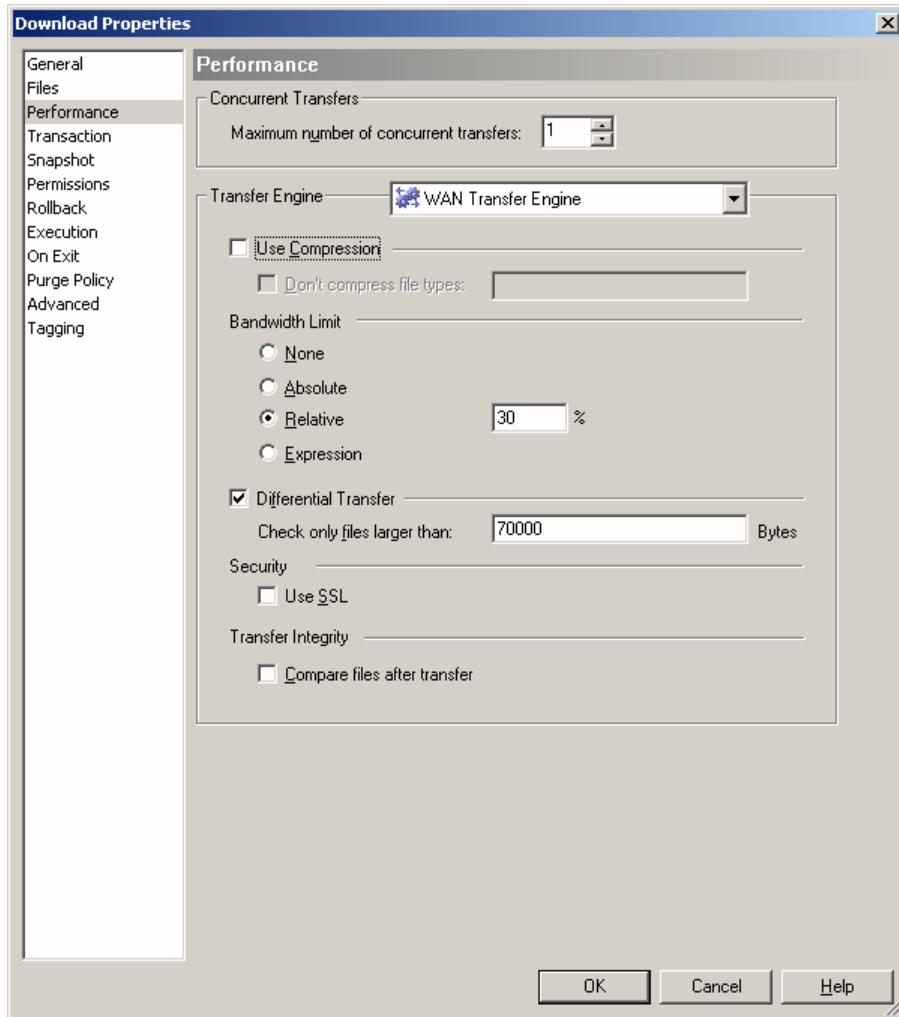
The scheduled jobs will be set to use the "Run daily at" option. Even though the replication runs daily (meaning on non-working days as well), since RDS only replicates the changed files from the last replication, if no work has been done and none of the files have changed, the RDS job will exit immediately after realizing there is no content to replicate.

Job Definition – Scheduling

3. In the **General** tab, define and configure each job to either **mirror** or **backup** the source directory to the target directories.

Use of the mirror logic will replicate file deletions made to the source, while use of the backup logic will only transfer updated files from the source to the target but will not delete files on the target.

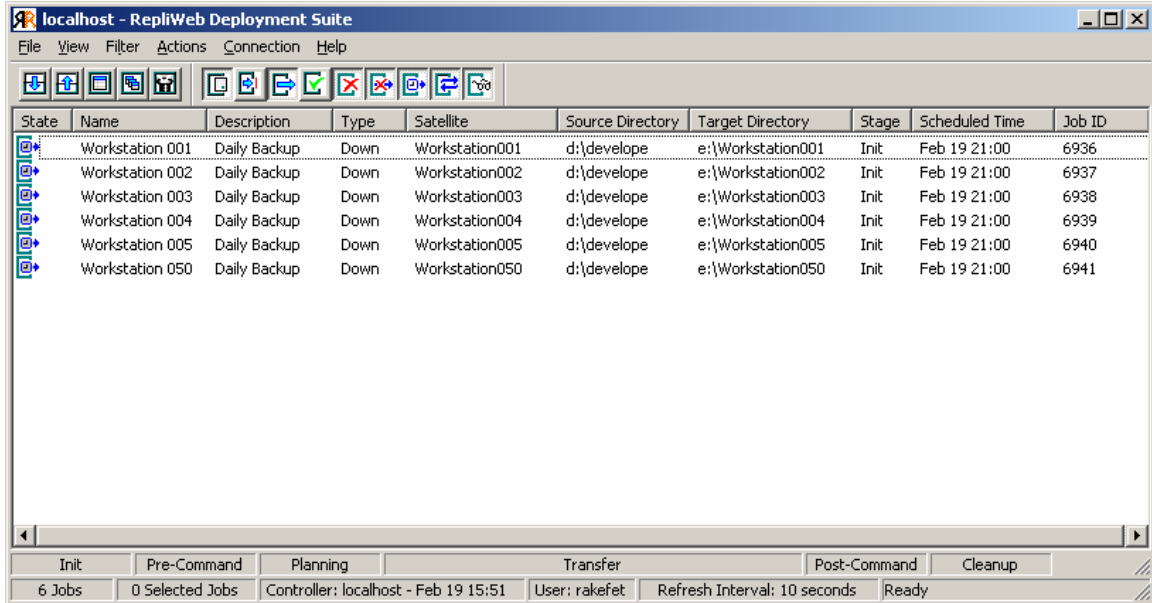
4. In the **Performance** tab, determine **Bandwidth Limit**, and to update large files at the block level, use the **Differential Transfer** option.



Job Properties – Performance Tab

5. Submit each job.

Ultimately there should be scheduled jobs displayed in the Job List Window, according to the number of workstations you are backing up.



The screenshot shows the 'localhost - RepliWeb Deployment Suite' window. The main area is a table with columns: State, Name, Description, Type, Satellite, Source Directory, Target Directory, Stage, Scheduled Time, and Job ID. There are five rows of jobs, all with a state of 'Down' and a scheduled time of 'Feb 19 21:00'. The status bar at the bottom shows '6 Jobs' and '0 Selected Jobs'.

State	Name	Description	Type	Satellite	Source Directory	Target Directory	Stage	Scheduled Time	Job ID
Down	Workstation 001	Daily Backup	Down	Workstation001	d:\develope	e:\Workstation001	Init	Feb 19 21:00	6936
Down	Workstation 002	Daily Backup	Down	Workstation002	d:\develope	e:\Workstation002	Init	Feb 19 21:00	6937
Down	Workstation 003	Daily Backup	Down	Workstation003	d:\develope	e:\Workstation003	Init	Feb 19 21:00	6938
Down	Workstation 004	Daily Backup	Down	Workstation004	d:\develope	e:\Workstation004	Init	Feb 19 21:00	6939
Down	Workstation 005	Daily Backup	Down	Workstation005	d:\develope	e:\Workstation005	Init	Feb 19 21:00	6940
Down	Workstation 050	Daily Backup	Down	Workstation050	d:\develope	e:\Workstation050	Init	Feb 19 21:00	6941

Init Pre-Command Planning Transfer Post-Command Cleanup
6 Jobs 0 Selected Jobs Controller: localhost - Feb 19 15:51 User: rakefet Refresh Interval: 10 seconds Ready

Job List – Scheduled Jobs

Using the RDS CLI, this command will schedule the job to run every day at 21:00 (9:00PM). This command needs to be executed for each workstation.

⇒

```
>rds submit -download -logic=mirror -controller=backup_server  
-controller_user=CONTROLLER_USER -controller_password=***  
-satellite=workstation001 -satellite_user=SATELLITE_USER  
-satellite_password=***  
-source=d:\develope -target=d:\workstation001  
-run_option=daily -daily= "21:00"  
-differential_transfer -differential_min_size=70000  
-bandwidth=30%
```

You can use a container that includes replication jobs for all workstations. Once defined, the container may be submitted in one CLI command

⇒

```
>rds submit ... -input_container=backup_container  
-input_container_location=local
```

Restoring Data

Prepare to restore in the event of failure, or “roll-back” situation.

Define an RDS Job Template (so it is readily available when needed) for a job in the reverse direction of the scheduled jobs (if the scheduled jobs are download jobs then the restore job will be an upload, and vice-versa).

NOTE: This job is not scheduled – it runs immediately upon submission to restore the data.

NOTE: In order to restore specific files, you must use **Mirror** Logic and specify the names of the files you want transferred.

In case of a corrupted file, the corrupted file is newer than the backed up file, so using **Backup** logic will not result in any file transfer.

If you do not specify the name of the file to restore, all backed up files will be transferred (mirrored) back to the working directory, resulting in loss of updated and valid data.

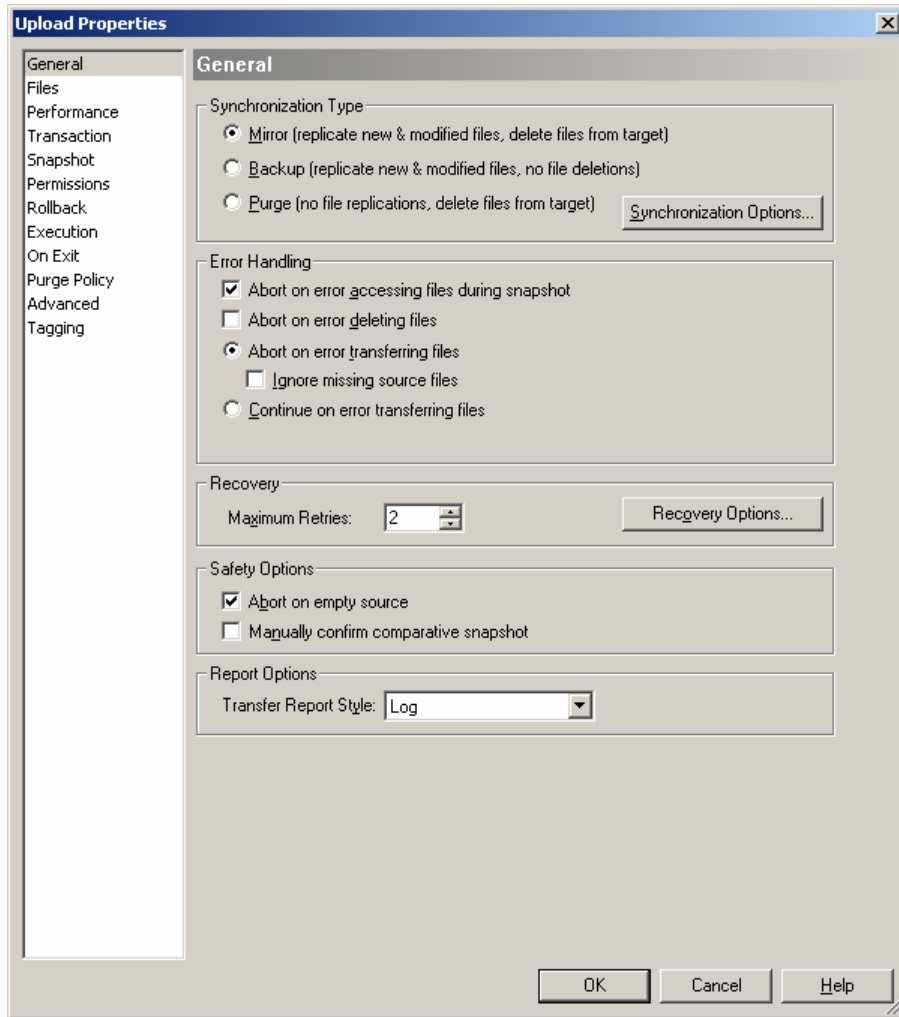
1. Define a Download job, replicating the working directory from each workstation to a unique location on the server.

The screenshot shows the 'Upload Definition' dialog box with the following fields and values:

- Name: Restore
- Description: select appropriate source directory...
- Schedule when you want the job to run. Options include once, continuously, at regular intervals, trigger file, daily or weekly. (Schedule... button)
- Set the job properties including synchronization type, transfer engine type, performance, snapshot settings, pre/post commands and permissions. (Properties... button)
- Replication source on production_srv (Controller)
 - Source Directory: c:\workstation020 (Browse... button)
- Replication Target
 - Satellite: localhost
 - Platform: Windows
 - User: administrator
 - Password: *****
 - Domain: compdomain
 - Target Directory: c:\daily_content (Browse... button)
- Buttons: Defaults, Template..., Submit, Cancel, Help

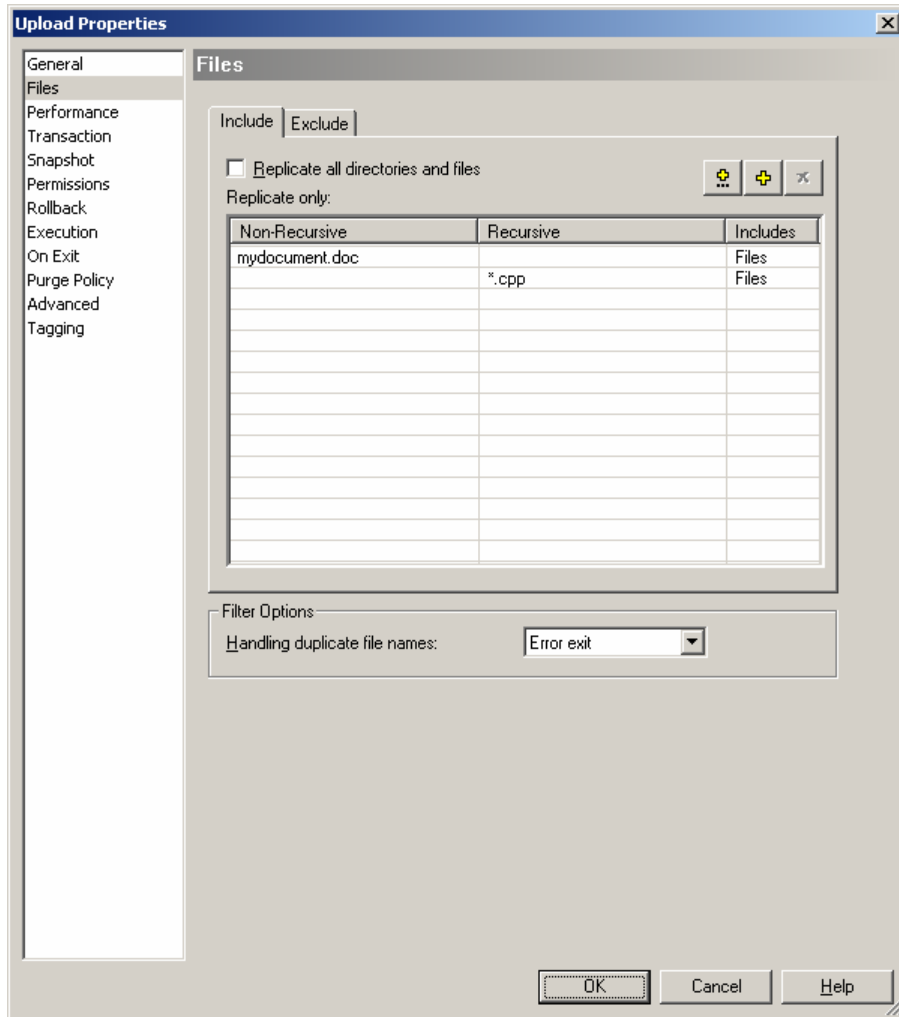
Upload Job Definition – Restoring Data

2. In the **General** tab, specify **Mirror** logic.



General Tab – Restore Using Mirror Logic

3. In the **Job Properties / Files tab / Include**, specify the names of the files you want to restore.



Files Tab – Specify Files to Restore

Using the CLI, to upload and restore one of the backup directories, issue an upload job from the appropriate directory to the relevant workstation.



```
>rds submit -upload -logic=mirror -controller=backup_server  
-controller_user=CONTROLLER_USER -controller_password=***  
-satellite=workstation001 -satellite_user=SATELLITE_USER  
-satellite_password=***  
-source=d:\workstation001 -target=d:\develop  
-file_specs=\"MyDocument.doc\"
```

4. The Actual Backup

Now, after all data has been collected to a centralized location, it's time to backup data to a tape drive. This step can be seamlessly integrated into the collection process.

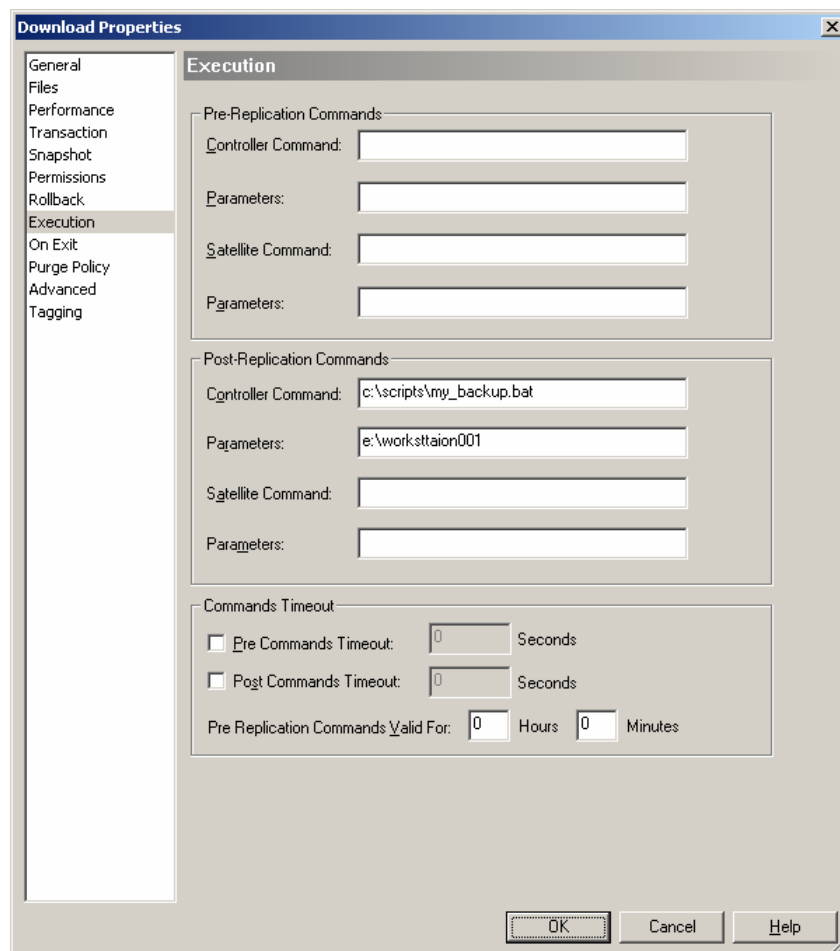
The manual way would be to use a batch file on a scheduled basis, which copies all content from the centralized location to the back-up drive.

To automate the process, RDS' **Post Command** mechanism can be used. Post processing is an integral part of the RDS replication process, and may be used to complete the backup process. The post command is actually a script that will be executed upon successful completion of the job.

1. The post command is specified in the Execution Tab.

Assuming our collection process is performed using Download jobs, all content is collected to the Controller. We would then specify a script to run on the Controller, transferring data from the Controller to the backup device.

The batch file is executed only if the transfer stage is successfully completed.



Execution Tab – Using Post Commands

2. The batch file called `my_backup.bat`, receives the directory to backup as a parameter. The batch file will look like this:

```
rem This script backs-up content from a centralized location to a
rem back-up drive
rem Parameters: $1 - location to backup - e:\Workstation001
rem $1 is the target directory of the replication job, which is
rem actually the source of this script to backup.
rem
rem Put your backup command here:
backup $1 into tape
exit 1
```

Using the RDS CLI, the submit command will now also include the post command script.



```
>rds submit -download -logic=mirror -controller=backup_server
-controller_user=CONTROLLER_USER -controller_password=***
-satellite=workstation001 -satellite_user=SATELLITE_USER
-satellite_password=***
-source=d:\develope -target=d:\workstation001
-run_option=daily -daily= "21:00"
-differential_transfer -differential_min_size=70000
-bandwidth=30%
-post_controller_command="c:\scripts\my_backup.bat"
-post_controller_parameters="e:\Workstation001"
```

5. Advanced

The process of backing using RDS to another machine, can be further refined.

1. **Snapshot-in-time** – RDS jobs may be scheduled in multiple ways: daily, weekly, every x hours, using a trigger file, etc. This range of options enables defining complex backup schemes, according to your needs. You may have partial updates during the week, and a complete update at the end of the week; some backup jobs may be put on hold on vacation times, etc.

Every scheduled job defined will have a different target directory, so that at each given moment, you'll have multiple backup directories to restore from.

Once a restore or a “roll-back” is required, define a job in the reverse direction, restoring data from the appropriate backup tree.

2. **Replicate changes within each backup file** – Using RDS' **Differential Transfer** option, RDS will attempt to replicate only the changed blocks within a file. That way, if any of the backup files are only appended, only these changes will be transferred, reducing network usage and transfer time.

3. **Automation** – Using RDS’ Scheduling options, the whole process can be automated and performed on a scheduled basis (weekly, daily, etc.).
4. **Templates and Containers** – A template file can be used for easy generation of similar jobs, or simply to have a backup of regularly scheduled jobs for future resubmission.

A **Container** is a method used to group multiple jobs in an organized fashion. A Container may be created, removed, renamed, or copied. It can include any kind of jobs and templates.

Pressing **Submit** will immediately submit all jobs in the Container.

5. **Post Command** – Using RDS’s Post Command mechanism allows performing the whole collection & backup operation automatically and cohesively.

###

For any additional information, please contact us at support.repliweb.com