NASA/TM—2005–209997/VER200603.0/VOL. 5b

ICESat (GLAS) Science Computing Facility Document Series

Volume 5b
SCF Interface Software Installation Guide
Version 200603.0

Kristine Barbieri
Anita Brenner

National Aeronautics and
Space Administration

Goddard Space Flight Center
Greenbelt, Maryland 20771

March 2006

TABLE OF CONTENTS

TABLE OF CONTENTS ..... **II**

SECTION 1: INTRODUCTION **1**

SECTION 2: RELATED DOCUMENTATION **3**

SECTION 3: ENVIRONMENT **4**

3.1 USER ACCOUNT 4
3.2 SCRIPT LOCATION 4
3.3 SECURE SHELL ENVIRONMENT 4
3.4 KORN SHELL ENVIRONMENTAL VARIABLES 4
3.5 SITE NAMES 5

SECTION 4: LIST OF SCRIPTS **7**

SECTION 5: INSTALLATION PROCEDURE **9**

SECTION 6: RUNNING THE SCRIPTS FROM A CRON JOB **10**

APPENDIX A: CONFIGURATION OF UNIX USER ACCOUNT TO ACCOMMODATE NON-INTERACTIVE SSH BASED DATA TRANSFERS **11**

APPENDIX B: ABBREVIATIONS & ACRONYMS **14**

SECTION 1: INTRODUCTION

A series of Perl scripts have been written to move data between the ICESat Science Investigator-led Processing System (I-SIPS) and the Instrument Support Facility (ISF), the main Science Computing Facility (mSCF), and the University of Texas Center for Space Research (UTCSR) and from the main Science Computing Facility to the remote Science Computing Facilities (rSCFs). This document describes how to install these scripts.

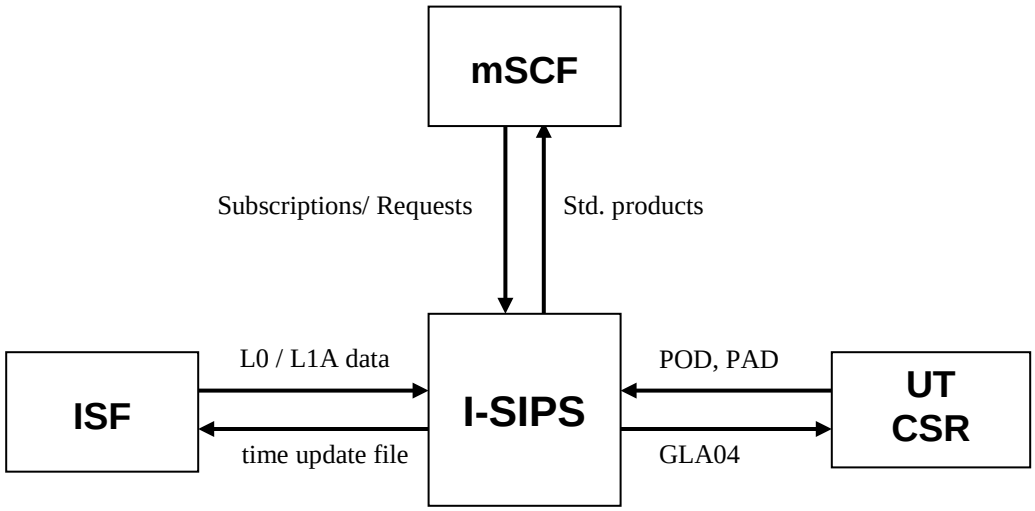

Figure 1-1: Data flow between the I-SIPS and the ISF, mSCF, and UTCSR

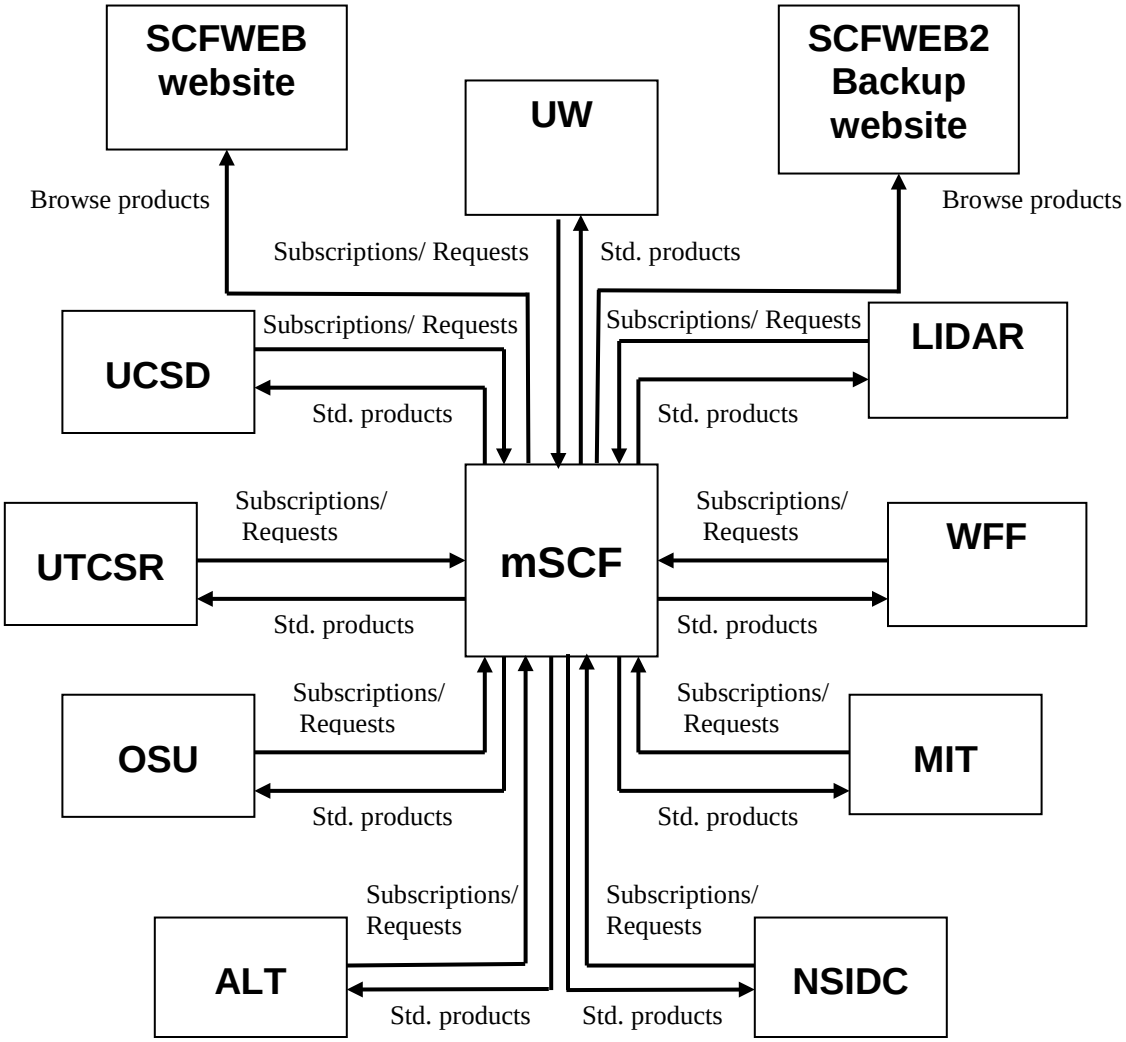

Figure 1-2: Data flow between the mSCF and the rSCFs

SECTION 2: RELATED DOCUMENTATION

- *SCF Architectural Design Document*
- *SCF Data Request Software Detailed Design Document*
- *SCF Data Request Software User's Guide*
- *SCF Data Visualization Software User's Guide*
- *SCF Operator's Guide*
- *SCF Interface Control Document*
- *SCF Interface Software Update Document*
- *SCF Interface Software Installation Guide*
- *SCF Interface Software Detailed Design Document*
- *SCF Interface Software Operator's Guide for rSCF*
- *SCF Interface Software Operator's Guide for mSCF*
- *I-SIPS Interface Software Operator's Guide for ISF*
- *I-SIPS Interface Software Operator's Guide for UTCSR*
- *Interface Control Document Between I-SIPS/ISF and CSR*

SECTION 3: ENVIRONMENT

3.1 User Account

It is expected that file movement between the local and remote hosts will be done under a specific user account. This account needs to be setup with the necessary files and appropriate permissions to enable the data to flow smoothly between the systems.

Data transfers between the I-SIPS and the mSCF and between the mSCF and the rSCFs are to be performed under the "scf" user account. Similarly, data transfers between the I-SIPS and the ISF are to be performed under the "isf" user account. Likewise, data transfers between the I-SIPS and the UTCSR are to be performed under the "utcsr" user account. Below is a table giving the setup parameters for these accounts:

User Name	User ID	Group	Group ID	Shell
scf	1005	external	401	Korn shell
isf	1006	external	401	Korn shell
utcsr	1007	exernal	401	Korn shell

3.2 Script Location

The perl scripts should be placed in a versioned subdirectory under the home directory (i.e. ~ /src/perl/200304.0). The scripts are then linked to the ~ /src/perl/ops directory and run from there. Korn shell scripts only reside in ~ /src/perl/ops. This path is one of the environmental variables used to run the scripts. The version number will be delivered by the mSCF with each software delivery. Any changes made by the rSCF must be documented and versioned in accordance with the mSCF versioning system.

3.3 Secure Shell Environment

Openssh is used for performing actions across hosts in a secure environment. It uses public key authentication, therefore keys need to be established in unencrypted, no-passphrase key files on each system. Once this is done and connections have been made between systems at least once, the scripts will run non-interactively. Refer to Appendix A for information on creating the key files.

3.4 Korn Shell Environmental Variables

Three files must be present in the home directory of the local host to initialize the environmental variables in the Korn shell: .dtprofile, .profile, and .kshrc. In the .dtprofile file, the following line must be present:

```
DTSOURCEPROFILE=true
```

Environmental variables need to be defined in the .kshrc file in the home directory of the local host running the scripts in order to perform certain functions.

For a local host pulling data from a remote host (i.e. for the ISF, mSCF, and UTCSR pulling data from the I-SIPS):

- export USERNAME=*user name*
- export ISIPS_HOST=*I-SIPS host name*
- export ISIPS_OUTPUT=*I-SIPS output cache*
- export ISIPS_INPUT=*I-SIPS input cache*
- export LOCAL_INPUT=*local input cache*
- export SRC_DIR=*directory containing Perl scripts*
- export SSH_DIR=*directory containing secure shell (ssh)*

For a local host pushing data to a remote host (i.e. for the mSCF pushing data to the rSCFs):

- export USERNAME=*username*
- export SRC_DIR=*directory containing Perl scripts*
- export SSH_DIR=*directory containing secure shell (ssh)*

For a local host moving data from the input cache to other subdirectories after data has been pulled from a remote host (i.e. for the ISF, mSCF, and UTCSR after pulling data from the I-SIPS):

- export LOCAL_NAME=*local site name*
- export LOCAL_HOST=*local host name*
- export export SRC_DIR=*directory containing Perl scripts*

Refer to the section "Site Names" for local site name information

For a local host moving data from the input cache to other subdirectories after data has been pushed by a remote host (i.e. for the rSCFs after data has been pushed by the mSCF):

- export USERNAME=*username*
- export REMOTE_HOST=*remote host name*
- export REMOTE_INPUT=*remote input cache*
- export LOCAL_NAME=*local site name*
- export LOCAL_HOST=*local host name*
- export LOCAL_INPUT=*local input cache*
- export SRC_DIR=*directory containing Perl scripts*
- export SSH_DIR=directory containing secure shell (ssh)

Refer to the section "Site Names" for local site name information

3.5 Site Names

Acronyms have been chosen to represent the various GLAS sites. These site names will be used in the Product Delivery Record and Product Acceptance Notice file names, the input and output cache names on the I-SIPS and mSCF, and in documentation. The table below gives the site name for each institution:

Institution	Site Name
GSFC ICESat Science Investigator-led Processing System	I-SIPS
GSFC Main Science Computing Facility	mSCF
GSFC Instrument Support Facility	ISF
GSFC Altimetry	ALT
GSFC Lidar	LIDAR

Massachusetts Institute of Technology	MIT
National Snow and Ice Data Center	NSIDC
Ohio State University	OSU
University of California, San Diego, Scripps Institute of Oceanography	UCSD
University of Texas, Austin, Center for Space Research	UTCSR
University of Washington	UW
Wallops Flight Facility	WFF
Remote Science Computing Facility (ALT, LIDAR, MIT, NSIDC, OSU, UCSD, UTCSR, UW, WFF collectively or generally)	rSCF
SCF web server (glas-scfweb.gsfc.nasa.gov)	SCFWEB
Backup SCF web server (glas-scfweb2.gsfc.nasa.gov)	SCFWEB2

SECTION 4: LIST OF SCRIPTS

Listed below are the files that need to be installed on the local host to serve the following purposes:

For a local host pulling data from a remote host and moving that data into subdirectories (i.e. for the ISF, MSCF, and UTCSR pulling data from the I-SIPS):

- pull_from_remote.pl
- pull_pan_from_remote.pl
- files_to_subdir_from_pull.pl
- read_pdr_files.pm
- verify_pdr.pm
- create_pan.pm
- get_subdir.pm
- read_keyword.pm
- get_cksum.pm
- send_mail_local.pm
- send_mail_isips.pm
- run_files_to_subdir_from_pull.ksh
- run_scripts.ksh
- run_scripts_pull_isips.ksh (mscf only)

For a local host creating data sets and pushing them to a remote host (i.e. for the ISF and UTCSR pushing data to the I-SIPS or for the mSCF pushing data to the rSCF's):

- create_pdr.pl
- push_to_remote.pl
- check_for_pan.pl
- read_pdr_files.pm
- verify_pdr.pm
- read_keyword.pm
- get_cksum.pm
- send_mail_local.pm
- run_create_pdr.ksh
- run_check_for_pan.ksh
- run_push_to_remote.ksh
- run_scripts<*_rscf*>.ksh
- run_scripts_push_isips.ksh (mscf only)

Additional files needed by the mSCF to fill subscriptions and special requests with data received from the I-SIPS:

- mscf_files_to_subdir_from_pull.pl
- pull_anc06_from_remote.pl
- read_fn_file.pl
- create_pdr_input.pm
- update_error_table.pm
- run_scripts_mscf.ksh
- run_mscf_files_to_subdir_from_pull.ksh
- run_ pull_anc06_from_remote.ksh
- run_read_fn_file.ksh
- run_cleanup.ksh

For a local host moving data from the ingest cache to other subdirectories after data has been pushed by a remote host (i.e. for the rSCFs after data has been pushed by the mSCF):

- files_to_subdir_from_push.pl
- read_pdr_files.pm
- verify_pdr.pm
- create_pan.pm
- get_subdir.pm
- read_keyword.pm
- get_cksum.pm
- send_mail_local.pm
- run_scripts.ksh

Additional files needed by SCFWEB/SCFWEB2 to untar files and convert files from HDF to PNG format:

- untar.ksh
- hdf_to_png.ksh

SECTION 5: INSTALLATION PROCEDURE

Once the appropriate user account, directories, and secure shell environment have been established, the following steps need to be performed:

- Transfer Perl and ksh scripts to dedicated directory
- Modify the Perl path in all scripts (first line)
- Modify the environmental variables in the .kshrc to reflect the correct paths and hosts
- Modify the environmental variables in the Perl scripts, if necessary
- Modify the paths in the ksh scripts, if necessary
- Modify the subdirectory paths in get_subdir.pm, if necessary
- Transfer test data to the appropriate ingest or output caches
- Run the scripts to test that they work

If the scripts will be run automatically from a cron job, perform the following additional steps:

- Copy the environmental variables in the .kshrc to run_scripts.ksh
- Copy PATH statements from .profile to run_scripts.ksh
- Modify the scripts to be executed in the run_scripts.ksh
- Set the cron job

SECTION 6: RUNNING THE SCRIPTS FROM A CRON JOB

Setting a cron job will enable the scripts to be invoked automatically at set time intervals. Before a cron job may be setup, the user has to have permission to perform a cron job. Refer to the crontab man page on how to authorize and setup a cron job. Below is an example of how a cron job can be setup:

- To vi the cron file: *crontab -e*
- Type the following line in the file to invoke the scripts at 0 and 30 minutes after the hour, every hour, everyday. All output will be redirected to the file cron.log.
  0,30 * * * * * path/run_scripts.ksh >> path/cron.log 2>&1
- Write and quit out of the file to invoke
- To view the cron file: *crontab -l*
- To remove the cron file: *crontab -r*

APPENDIX A: CONFIGURATION OF UNIX USER ACCOUNT TO ACCOMMODATE NON-INTERACTIVE SSH BASED DATA TRANSFERS

Purpose

The configuration of a user account on a Unix system to permit non-interactive SSH based data transfers between two Unix systems. This approach will utilize public-key authentication with plain text key files (no passphrase).

Prerequisites

OPENSSH 3.4p1 with OPENSSL v0.96 or newer

Installation on a Local System

Perform the following operations when logged in as the account owner:

These examples assume the OPENSSH installation system directory is "/opt/openssh".

1) Create the OPENSSH configuration directory

   - mkdir $HOME/.ssh
   - chmod 700 $HOME/.ssh
   - chmod go-w $HOME (do not allow home directory to be group/world writable)

2) Generate the RSA authentication key pair for SSH protocol version 1:

   /opt/openssh/bin/ssh-keygen -t rsa1 -f $HOME/.ssh/identity -N ""

This operation will generate the private key ($HOME/.ssh/identity) and the corresponding public key ($HOME/.ssh/identity.pub).

3) Generate the DSA authentication key pair for SSH protocol version 2:

   /opt/openssh/bin/ssh-keygen -t dsa -f $HOME/.ssh/id_dsa -N ""

This operation will generate the private key ($HOME/.ssh/id_dsa) and the corresponding public key ($HOME/.ssh/id_dsa.pub).

4) Generate the RSA authentication key pair for SSH protocol version 2:

   /opt/openssh/bin/ssh-keygen -t rsa -f $HOME/.ssh/id_rsa -N ""

This operation will generate the private key ($HOME/.ssh/id_rsa) and the corresponding public key ($HOME/.ssh/id_rsa.pub).

5) Set apropriate file permissions for OPENSSH configuration files. File permissions for keys will PROBABLY be set correctly when generated. If not, execute the following commands:

```
chmod 700 $HOME/.ssh
chmod 600 $HOME/.ssh/identity $HOME/.ssh/id_dsa $HOME/.ssh/id_rsa
chmod 644 $HOME/.ssh/identity.pub $HOME/.ssh/id_dsa.pub
chmod 644 $HOME/.ssh/id_rsa.pub
```

Installation on a Remote System

Use the public keys on a remote system where you wish to login to.

By first transferring the aforementioned public keys to an account on a remote system, they can be used to perform non interactive SSH based data transfers from the local system to the remote system. The expression "REMOTEHOST:/$HOME" refers to the home directory of an account on a remote system.

1) The contents of $HOME/.ssh/identity.pub can be concatenated to

   REMOTEHOST:/$HOME/.ssh/authorized_keys for SSH protocol version 1.

   I) transfer $HOME/.ssh/identity.pub to a remote system (REMOTEHOST:/$HOME)

   II) cat REMOTEHOST:/$HOME/identity.pub >>

   REMOTEHOST:/$HOME/.ssh/authorized_keys

2) The contents of $HOME/.ssh/id_dsa.pub can be concatenated to

   REMOTEHOST:/$HOME/.ssh/authorized_keys for SSH protocol version 2.

   I) transfer $HOME/.ssh/id_dsa.pub to a remote system (REMOTEHOST:/$HOME)

   II) cat REMOTEHOST:/$HOME/id_dsa.pub >>

   REMOTEHOST:/$HOME/.ssh/authorized_keys

3) The contents of $HOME/.ssh/id_rsa.pub can be concatenated to

   REMOTEHOST:/$HOME/.ssh/authorized_keys for SSH protocol version 2.

   I) transfer $HOME/.ssh/id_rsa.pub to a remote system (REMOTEHOST:/$HOME)

   II) cat REMOTEHOST:/$HOME/id_rsa.pub >>

   REMOTEHOST:/$HOME/.ssh/authorized_keys

4) Set file permissions for authorized_keys on remote host.

   When logged in to remote system:

   chmod 600 $HOME/.ssh/authorized_keys

5) A sample data transfer form a local system to a remote system using accounts with the same name (UID)

   when logged into local system:

   /opt/openssh/bin/scp $HOME/sample_data_file REMOTEHOST:$HOME

Shortcut for Creating SSH Keys on a Remote Site

For SSH protocol version 2

1) On the remote site, generate the DSA authentication key pair in /SCF/.ssh:
   - /opt/openssh/bin/ssh-keygen –t dsa –N “”
2) Copy /SCF/.ssh/id_dsa.pub from the remote site to the main site. Make sure to rename id_dsa.pub to id_dsa.pub_<rscf> on the main site or you will overwrite the main site’s id_dsa.pub file.
   - i.e. scp /SCF/.ssh/id_dsa.pub icesat0:/SCF/.ssh/id_dsa.pub_glo
3) Copy /SCF/.ssh/id_dsa.pub from the main site to the remote site. Make sure to rename id_dsa.pub to id_dsa.pub_<mscf> on the remote site or you will overwrite the remote site’s id_dsa.pub file.
   - i.e. scp /SCF/.ssh/id_dsa.pub glo:/SCF/.ssh/id_dsa.pub_icesat0
4) On the remote site, concatenate /SCF/.ssh/id_dsa.pub_<mscf> to the end of the authorized_keys file. You can either cut and paste the files or cat them.
5) On the main site, concatenate /SCF/.ssh/id_dsa.pub_<rscf> to the end of the authorized_keys file. You can either cut and paste the files or cat them.

APPENDIX B: ABBREVIATIONS & ACRONYMS

GLAS	Geoscience Laser Altimeter System
GSFC	Goddard Space Flight Center
ICESat	Ice, Cloud, and land Elevation Satellite
ISF	Instrument Support Facility
I-SIPS	ICESat Science Investigator-led Processing System
LIDAR	LIght Detection And Ranging
MIT	Massachusetts Institute of Technology
mSCF	Main Science Computing Facility
NASA	National Aeronautics and Space Administration
OSU	Ohio State University
rSCF	Remote Science Computing Facility
UCSD	University of California at San Diego
UTCSR	University of Texas Center for Space Research
UW	University of Washington
WFF	Wallops Flight Facility