

# RMFT Java Command Line Reference Guide

Software Version 2.4.2

July 19, 2009

July 19, 2009  
Copyright © 2000-2008 by RepliWeb, Inc.

The information in this document has been compiled with care, but RepliWeb makes no warranties as to accurateness or completeness, as the software described herein may be changed or enhanced from time to time. This information does not constitute commitments or representations by RepliWeb, and is subject to change without notice. The software described in this document is furnished under license and may be used or copied only in accordance with the terms of this license.

No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written consent of RepliWeb, Inc.

Any trademarks, trade names, service marks, or service names owned or registered by any other company and used in this document are proprietary to that company.

Please direct correspondence or inquiries to:

RepliWeb, Inc.  
6441 Lyons Road  
Coconut Creek  
FL 33073

Phone: (954) 946-2274  
Fax: (954) 337-6424  
E-Mail: [info@repliweb.com](mailto:info@repliweb.com),  
Support: <http://support.repliweb.com>  
Web Site: <http://www.repliweb.com/>

# Table of Contents

|  |          |
|--|----------|
| <b>Introduction .....</b>                        | <b>1</b> |
| Abbreviating Qualifiers .....                    | 2        |
| <b>1. Sending Packages.....</b>                  | <b>3</b> |
| The 'send' Command .....                         | 3        |
| Basic Qualifiers for Sending a Package .....     | 4        |
| user .....                                       | 4        |
| password.....                                    | 4        |
| recipients.....                                  | 4        |
| server .....                                     | 5        |
| subject.....                                     | 5        |
| message .....                                    | 5        |
| distribution_list .....                          | 5        |
| files.....                                       | 6        |
| list_file .....                                  | 6        |
| subdirectory_files.....                          | 7        |
| Retry Qualifiers.....                            | 8        |
| auto_recovery .....                              | 8        |
| maximum_retries .....                            | 8        |
| retry_interval .....                             | 8        |
| Certificate-Based Login Qualifiers.....          | 9        |
| keystore_name .....                              | 9        |
| keystore_password.....                           | 9        |
| Qualifiers for Encrypting a Package .....        | 10       |
| encrypt .....                                    | 10       |
| public_keystore_name.....                        | 11       |
| public_keystore_password .....                   | 11       |
| public_certificate_subject.....                  | 11       |
| shared_secret .....                              | 12       |
| secure .....                                     | 12       |
| cipher .....                                     | 12       |
| ssl_trace.....                                   | 13       |
| Qualifiers for Digitally Signing a Package ..... | 14       |
| sign_files.....                                  | 14       |
| private_keystore_name .....                      | 14       |

|  |           |
|--|-----------|
| private_keystore_password .....                          | 14        |
| private_certificate_subject .....                        | 15        |
| private_certificate_password .....                       | 15        |
| Qualifiers for Sending Packages as Zip Files .....       | 16        |
| package_format .....                                     | 16        |
| zip_file_name .....                                      | 16        |
| delete_zip_file .....                                    | 16        |
| Example Command .....                                    | 17        |
| Other 'Send' Qualifiers .....                            | 18        |
| compare_files .....                                      | 18        |
| abort_on_transfer_error .....                            | 18        |
| notify .....   | 18        |
| priority .....   | 19        |
| expiration .....   | 19        |
| proxy_server .....                                       | 19        |
| proxy_port .....   | 19        |
| package_directory .....                                  | 20        |
| copy_files .....   | 20        |
| delete_package .....                                     | 20        |
| offline .....  | 21        |
| transport .....  | 21        |
| fastcopy_qualifiers .....                                | 22        |
| relative_server_url .....                                | 22        |
| hide_recipients .....                                    | 23        |
| allow_nonascii_filenames .....                           | 23        |
| unauthenticated_download .....                           | 23        |
| Submitting an Offline Package .....                      | 24        |
| Example of How to Submit an Offline Package .....        | 24        |
| Example of How to Submit an Offline Zipped Package ..... | 24        |
| package_directory .....                                  | 25        |
| <b>2. Searching for Packages .....</b>                   | <b>26</b> |
| server .....   | 27        |
| relative_server_url .....                                | 27        |
| user .....   | 27        |
| password .....   | 27        |
| downloaded .....   | 28        |
| secure .....   | 28        |
| proxy_server .....                                       | 28        |
| proxy_port .....   | 28        |
| files .....  | 29        |
| format .....   | 29        |

|  |           |
|--|-----------|
| xml .....                                | 29        |
| output .....                             | 29        |
| folder .....                             | 30        |
| after .....                              | 30        |
| before .....                             | 30        |
| subject .....                            | 30        |
| from .....                               | 30        |
| to .....                                 | 31        |
| message .....                            | 31        |
| read .....                               | 31        |
| encrypted .....                          | 31        |
| signed .....                             | 31        |
| expired .....                            | 31        |
| expired_since .....                      | 32        |
| expired_before .....                     | 32        |
| minimum_size .....                       | 32        |
| maximum_size .....                       | 32        |
| trace .....                              | 33        |
| Certificate-Based Login Qualifiers ..... | 34        |
| keystore_name .....                      | 34        |
| keystore_password .....                  | 34        |
| <b>3. Downloading Packages .....</b>     | <b>35</b> |
| server .....                             | 36        |
| relative_server_url .....                | 36        |
| user .....                               | 36        |
| password .....                           | 36        |
| secure .....                             | 37        |
| proxy_server .....                       | 37        |
| proxy_port .....                         | 37        |
| package_directory .....                  | 37        |
| package_id .....                         | 37        |
| files .....                              | 38        |
| compare_files .....                      | 38        |
| specific_files .....                     | 38        |
| after .....                              | 38        |
| before .....                             | 38        |
| subject .....                            | 39        |
| from .....                               | 39        |
| to .....                                 | 39        |
| message .....                            | 39        |
| read .....                               | 39        |

|  |           |
|--|-----------|
| encrypted .....                                | 39        |
| signed .....                                   | 40        |
| expired_since .....                            | 40        |
| expired_before .....                           | 40        |
| minimum_size .....                             | 40        |
| maximum_size .....                             | 40        |
| abort_on_transfer_error .....                  | 41        |
| trace .....                                    | 41        |
| post_command .....                             | 41        |
| post_command_parameters .....                  | 41        |
| server_open .....                              | 41        |
| check_inbox_interval .....                     | 42        |
| Certificate-Based Login Qualifiers .....       | 43        |
| keystore_name .....                            | 43        |
| keystore_password .....                        | 43        |
| <b>4. Scrambling Passwords .....</b>           | <b>44</b> |
| Scrambled Password Qualifiers .....            | 44        |
| scrambled_login_password .....                 | 44        |
| scrambled_keystore_password .....              | 44        |
| scrambled_public_keystore_password .....       | 44        |
| scrambled_public_certificate_password .....    | 45        |
| scrambled_private_keystore_password .....      | 45        |
| scrambled_private_certificate_password .....   | 45        |
| <b>5. Verifying and Decrypting Files .....</b> | <b>46</b> |
| Verifying Files .....                          | 47        |
| verify .....                                   | 48        |
| public_keystore_name .....                     | 48        |
| public_keystore_password .....                 | 48        |
| public_certificate_subject .....               | 48        |
| package_id .....                               | 49        |
| signature_file .....                           | 49        |
| signature_directory .....                      | 50        |
| Decrypting Files .....                         | 51        |
| decrypt .....                                  | 52        |
| key_id .....                                   | 52        |
| private_keystore_name .....                    | 52        |
| private_keystore_password .....                | 52        |
| private_certificate_subject .....              | 53        |
| private_certificate_password .....             | 53        |

|   |           |
|---|-----------|
| shared_secret .....                                 | 53        |
| cipher .....  | 53        |
| Shared Decryption and Verification Qualifiers ..... | 54        |
| file .....  | 54        |
| file_spec.....                                      | 54        |
| source_directory .....                              | 54        |
| target_file .....                                   | 55        |
| target_directory.....                               | 55        |
| <b>A. Time Expressions .....</b>                    | <b>56</b> |
| Absolute Time Expressions .....                     | 56        |
| Relative Time Expressions.....                      | 57        |

# Introduction

RMFT Java CLI is a platform-independent RMFT client that enables you to perform the following tasks, using a command prompt:

- Send files to other RMFT users (securely or non-securely, encrypted or unencrypted).
- Search for packages using numerous selection criteria.
- Download packages using numerous selection criteria.
- Decrypt files.
- Verify the signatures of digitally signed files.

Some RMFT Java CLI qualifiers (parameters) are mandatory such as the `-user` qualifier, while others can be used to implement optional features such as encryption. All files that you send are first processed by RMFT Server before being either automatically forwarded to the recipients or downloaded by the recipients (or a combination of both).

RMFT Java CLI commands can be issued through a UNIX shell, VMS DCL, or Windows MS-DOS command line interface.

To be able to invoke RMFT Java command line operations, the following elements are required:

- Java virtual machine v1.4 or above installed on your machine (excluding beta versions).
- RMFT client JAR file residing in the RMFT working directory.

When issuing RMFT Java CLI commands use the following syntax:

```
>java -jar "C:\Program Files\RepliWeb\RMFT\b-hub\bin\bhub_cli.jar"  
command mandatory_qualifiers <optional_qualifiers>
```

Example:

```
java -jar "C:\Program Files\RepliWeb\RMFT\b-hub\bin\bhub_cli.jar" send  
-server=192.247.160.148 -user=mike -password=1234 -recipients=jane  
-files=c:\lic.txt -subject="from java"
```

Qualifiers are case insensitive, meaning that they may be written in upper case or lower case or a combination of both.

As a general rule, if the value of a qualifier contains a space, the qualifier value must be enclosed in quotation marks. For instance, in the following example the directory `bhub files` contains a space; therefore the value (full path) of the `-files` qualifier must be enclosed within quotation marks:

```
-files="c:\bhub files\a.txt"
```

## Abbreviating Qualifiers

Qualifier names and values may be abbreviated, providing that the abbreviation is unique and does not conflict with another qualifier or abbreviated qualifier.

For example, the following are both valid ways of specifying the [expired\\_since](#) qualifier:

```
-expired_s
```

And:

```
-expired_since
```

However, the following abbreviation of the `-expired_since` qualifier is not permitted, as it conflicts with the `-expired_before` qualifier:

```
-expired
```

# 1. Sending Packages

This chapter describes the qualifiers that you can use to send packages. Some of the qualifiers are mandatory while others are optional.

When you issue the RMFT `send` command, a package is created and sent to RMFT Server. A package is a folder containing the files that you selected together with various file processing and routing metadata required by RMFT Server.

## The 'send' Command

The command for sending packages is `send`. The `send` command must precede any qualifiers that you want to use, as is illustrated in the following example:

```
>java -jar bhub_cli.jar send -server=192.247.160.148 -user=user1  
-password=bb -recipients=user2 -files=c:\lic.txt -subject="from java"
```

Note that the command in the above example is issued from the same directory as the `bhub_cli.jar` file. If you issue the command from a different directory, the full path of the `bhub_cli.jar` file must be specified.

**IMPORTANT** Paths or qualifier values that contains spaces must be enclosed with quotation marks, as shown below.

```
>java -jar "C:\Program Files\RepliWeb\RMFT\b-hub\bin\bhub_cli.jar" send  
-server=192.247.160.148 -user=bb -password=bb -recipients=aa  
-files=c:\lic.txt -subject="from java"
```

## Basic Qualifiers for Sending a Package

### user

Syntax: `-user=<user name>`

Example: `-user=dan3`

Use the `-user` qualifier to specify your RMFT user name (assigned by the RMFT administrator).

See also: [password](#) (4), [recipients](#) (4) and [server](#) (5).

### password

Syntax: `-password=password`

Use the `-password` qualifier to specify your RMFT password (assigned by the RMFT administrator).

See also: [server](#) (5), [recipients](#) (4) and [user](#) (4)

### recipients

Syntax: `-recipients=username,hostname,username`

Example: `-recipients=joe,bach:symphony,mary`

Use the `-recipients` qualifier to specify a list of package recipients. Recipient names must be separated by a comma with no spaces.

A recipient can be an RMFT user or a host. A host is a computer to which RMFT Server has been configured to transfer files. To send files to a host, you must specify the host using the following convention (see example above):

**Syntax:** `host_nickname:target_nickname`

**Example:** `bach:symphony`

The host nickname and target nickname are aliases for the real host name and target directory path. Consult with your RMFT administrator regarding need to send packages to hosts.

See also: [password](#) (4), [user](#) (4) and [server](#) (5).

## server

Syntax: `-server=machine name/IP address`

Example: `-server=123.123.123.23`

Use the `-server` qualifier to specify which RMFT Server to upload the package to.

See also: [password](#) (4), [recipients](#) (4) and [user](#) (4)

## subject

Syntax: `-subject=text_string`

Example: `-subject="financial reports"`

Use the `-subject` qualifier to provide a short description of the package. The subject will be displayed to the recipients when they access their RMFT client inbox.

See also: [message](#) (5)

## message

Syntax: `-message="message_text"`

Example: `-message="The files in this package contain sensitive information and should be handled accordingly"`

The `-message` qualifier enables you to send a short message to the package recipients.

## distribution\_list

Syntax: `-distribution_list=filename`

Default: `-distribution_list=[working_directory]filename`

Example: `-distribution_list=c:\temp\vendors.txt`

Use the `-distribution_list` qualifier to specify the name of a distribution list instead of specifying the names of individual recipients (using the `-recipients` qualifier). A distribution list can contain users and hosts.

**IMPORTANT** Each recipient must occupy a separate row in the distribution list file.

If you do not specify a path, the program assumes that the distribution list resides in your current working directory.

See also: [recipients](#) (5).

## files

Syntax 1: `-files=full path[,full path]`

Syntax 2: `-files=path`

Example 1: `-files="c:\upload\a.txt,c:\temp three\b.txt"`

Example 2: `-files=c:\upload`

---

**Note:** If you do not specify a path, the program assumes that the files are in your current working directory.

---

Use the `-files` qualifier to specify a list of files that you want to send to other RMFT users. If you specify several files, use a comma to separate the file paths from each other. If a directory or file name contains a space, you must enclose the qualifier value with quotation marks (see Example 1 above).

The `-files` qualifier also supports the `*` and `?` wildcard characters. You can use the asterisk as a substitute for zero or more characters. For example, to upload all `.txt` files from `C:\MyFiles\`, type the following:

**`-files=c:\myfiles\*.txt`**

You can use the question mark as a substitute for a single character in a name. For example, if you typed `r?d.doc`, the files `red.doc` and `rid.doc` would be uploaded to RMFT Server, but not `round.doc`.

See also: [list\\_file](#) (6) and [subdirectory\\_files](#) (7)

## list\_file

Syntax: `-list_file=path\filename`

Example: `-list_file=c:\data\files.txt`

The `-list_file` qualifier specifies the name of a text file that contains a list of files to be sent. You can use the `*` and `?` wildcard characters to match file specifications. For example, typing `c:\temp\*.txt` will transfer all text files in the `temp` directory. For more about using wildcards, see [files](#) (6).

If you do not specify a path, the program assumes that the list file is in your current working directory.

The following example shows the format of the text file specified by the `-list_file` qualifier.

```
c:\autoexec.bat
c:\config.sys
c:\data\a.b
c:\data\c.d
.
.
```

See also: [files](#) (6) and [subdirectory files](#) (7)

## subdirectory\_files

Syntax: `-subdirectory_files`

Example: `-subdirectory_files`

Use the `-subdirectory_files` qualifier to upload all subdirectory files that match the files specified with the `-files` qualifier and recreate the directory tree structure on the recipient machines. If only a directory name is specified with the `-files` qualifier, then all files in the specified directory's subdirectories will also be uploaded to RMFT Server.

**Note** The directory tree will only be recreated if RMFT Server is configured to automatically deliver the package to the recipients.

### Example 1:

The following command uploads all `.txt` files in `c:\upload` and its subdirectories to RMFT Server.

```
C:\Program Files\RepliWeb\RMFT\b-hub\bin>java -jar bhub_cli.jar send
-user=user1 -pass=33 -files=c:\upload\*.txt -server=localhost
-recipient=user1 -subdirectory_files
```

**Example 2:**

The following command uploads all files in **c:\upload** and its subdirectories to RMFT Server.

```
C:\Program Files\RepliWeb\RMFT\b-hub\bin>java -jar bhub_cli.jar send
-user=user1 -pass=33 -files=c:\upload -server=localhost -recipients=user1
-subdirectory_files
```

See also: [files](#) (6) and [list\\_file](#) (6).

## Retry Qualifiers

### **auto\_recovery**

Syntax: `-auto_recovery`

Use the `-auto_recovery` qualifier to automatically resend the package in the event of failure. If you include this qualifier in the command line, the Java CLI will attempt to resend the package from the precise point of failure, even if the failure occurred in the middle of a file.

### **maximum\_retries**

Syntax: `-maximum_retries=<number of retries>`

Use the `-maximum_retries` qualifier to specify the maximum number of times to retry the operation.

### **retry\_interval**

Syntax: `-retry_interval=<in seconds>`

Use the `-retry_interval` qualifier to set the interval between retry attempts. The interval is in seconds.

## Certificate-Based Login Qualifiers

Your RMFT administrator may require you to log in to RMFT Server using a certificate. If this is the case, use the qualifiers below to specify the name and password of your certificate keystore.

### **keystore\_name**

Syntax: `-keystore_name=<keystore_full_path>`

Example: `-keystore_name=c:\ph\certs\ph.jks`

Use the `-keystore_name` qualifier to specify the location of your Java keystore. The name of your Java keystore is required when using a certificate to authenticate your identity to RMFT Server.

The keystore that you specify may contain several different client certificates. Normally, Java will find a keystore certificate that is also accepted by the server.

If the IIS server is configured to **Require client certificates** and your keystore does not contain a valid certificate, your certificate will be rejected with the following error:

```
HTTP error 403 Access Forbidden
```

The recommended solution is to create a keystore containing one certificate and use it exclusively for sending files to RMFT Server.

See also: [keystore\\_pass](#) (9) and [secure](#) (12)

### **keystore\_password**

Syntax: `-keystore_password`

The `-keystore_password` qualifier specifies the password for accessing your keystore.

See also: [keystore\\_name](#) (9) and [secure](#) (12)

## Qualifiers for Encrypting a Package

You can encrypt packages by adding encryption qualifiers to the basic command line "send" qualifiers described in [Basic Qualifiers for Sending a Package](#) (4). The supported encryption methods can be implemented using the qualifiers described below.

### encrypt

The `-encrypt` qualifier accepts four possible values, each representing a different method of encryption.

#### Syntax:

```
-encrypt=shared_secret|pki_single_recipient|encrypt_to_server
```

#### Example:

```
-encrypt=encrypt_to_server
```

### shared\_secret

Use this method to encrypt files during transfer between your computer and the recipients' computers.

#### To encrypt files using a shared secret, specify:

- ◆ `-encrypt=shared_secret`

To implement this method, you must also specify a cipher using the `-cipher` qualifier and a shared secret using the `-shared_secret` qualifier. To be able to decrypt the files, the shared secret and the encryption cipher must also be known to the package recipients.

See also: [shared\\_secret](#) (12) and [cipher](#) (12).

### pki\_single\_recipient

Use this method to encrypt files during transfer between your computer and a single recipient's computer. To implement this method, the recipient's public key certificate should reside in one of your certificate keystores.

#### To encrypt files using a recipient's public key, specify:

- ◆ `-encrypt=pki_single_recipient`

## encrypt\_to\_server

This method uses RMFT Server's public key certificate to encrypt files during upload to RMFT Server. To implement this method, the RMFT Server public key certificate **does not** have to reside on your machine.

**To encrypt files during upload to RMFT Server using the RMFT Server certificate, specify:**

- ◆ `-encrypt=encrypted_to_server`

## public\_keystore\_name

Syntax: `-public_keystore_name=<keystore_full_path>`

Example: `-public_keystore_name=c:\ph\certs\ph.jks`

Use the `-public_keystore_name` qualifier to specify the full path of the Java keystore that contains the recipient's public key certificate. The recipient's public key certificate is required when encrypting files between your computer and a recipient's computer (using asymmetric encryption).

See also: [encrypt](#) (10), [public\\_keystore\\_pass](#) (11), [public\\_certificate\\_subject](#) (11)

## public\_keystore\_password

Syntax: `-public_keystore_password`

Use the `-public_keystore_password` qualifier to specify the password of the Java keystore that contains the recipient's public key certificate. The recipient's public key certificate is required when encrypting files between your computer and a recipient's computer (using asymmetric encryption).

See also: [encrypt](#) (10), [public\\_keystore\\_name](#) (11), [public\\_certificate\\_subject](#) (11)

## public\_certificate\_subject

Syntax: `-public_certificate_subject=<unique part of certificate subject>`

Example: `-public_certificate_subject=user@company.com`

Use the `-public_certificate_subject` qualifier to specify a unique part of the subject of the public key certificate with which you want to encrypt the files. The subject is required in order to identify the certificate that you want to use.

See also: [encrypt](#) (10), [public\\_keystore\\_name](#) (11), [public\\_keystore\\_pass](#) (11).

## shared\_secret

Syntax: `-shared_secret=password`

Example: `-shared_secret=4319834935785hf91`

The `-shared_secret` qualifier specifies a shared secret that must be known to the recipient(s) in order for them to decrypt the package contents.

The `-shared_secret` qualifier must be used together with the `-cipher` and `-encrypt` qualifiers.

## Example

```
C:\Documents and Settings>java -jar "C:\Program Files\RepliWeb\RMFT\b-  
hub\bin\bhub_cli.jar" send -user=User_1 -recipients=User_2  
-password=123 -server=localhost -files=c:\temp\a.txt  
-encrypt=shared_secret -shared_secret=hfiuq34743h -cipher=rc4
```

See also: [cipher](#) (12) [encrypt](#) (10) and [secure](#) (12)

## secure

Syntax: `-secure`

Use the `-secure` qualifier to establishes a secure connection (SSL) to RMFT Server.

To utilize this option, the appropriate certificate must be installed on the IIS server. Contact the RMFT administrator to verify whether or not a server-side certificate exists. In addition, you must import the CA (Certificate Authority) certificate to your Java `cacerts` file.

See also: [shared\\_secret](#) (12), [cipher](#) (12) and [encrypt](#) (10).

## cipher

Syntax: `-cipher=rc4`

The `-cipher` qualifier specifies the cipher that you want to use to encrypt the files. Currently, only `rc4` is supported.

See also: [shared\\_secret](#) (12), [encrypt](#) (10) and [secure](#) (12).

## **ssl\_trace**

Syntax: `-ssl_trace`

The `-ssl_trace` qualifier generates a detailed trace of files sent using the `-secure` qualifier.

## Qualifiers for Digitally Signing a Package

Use your certificate private key to digitally sign files. Only recipient's in possession of the corresponding public key will be able to verify the digital signature.

### **sign\_files**

Syntax: `-sign_files`

Use the `-sign_files` qualifier to digitally sign files using your private key certificate. The `-sign_files` qualifier must be used in conjunction with the `-private_certificate_subject`, `-private_certificate_password`, `-private_keystore_name` and `-private_keystore_password` qualifiers.

See also: [private\\_keystore\\_name](#) (14), [private\\_keystore\\_pass](#) (14), [private\\_certificate\\_subject](#) (15), [private\\_certificate\\_pass](#) (15).

### **private\_keystore\_name**

Syntax: `-private_keystore_name=<keystore_full_path>`

Example: `-private_keystore_name=c:\ph\certs\ph.jks`

Use the `-private_keystore_name` qualifier to specify full path of the Java keystore that contains your private key certificate. The recipients will require your public key certificate in order to verify your digital signature.

See also: [sign\\_files](#) (14), [private\\_keystore\\_pass](#) (14), [private\\_certificate\\_subject](#) (15), [private\\_certificate\\_pass](#) (15).

### **private\_keystore\_password**

Syntax: `-private_keystore_password`

Use the `-private_keystore_password` qualifier to specify the password of the Java keystore that contains your private key certificate.

See also: [sign\\_files](#) (14), [private\\_keystore\\_name](#) (14), [private\\_certificate\\_subject](#) (15), [private\\_certificate\\_pass](#) (15).

## **private\_certificate\_subject**

Syntax: `-private_certificate_subject=<unique part of certificate subject>`

Example: `-private_certificate_subject=user@company.com`

Use the `-private_certificate_subject` qualifier to specify a unique part of the subject of the private key certificate that you want to use to sign the files. The subject enables Java to identify the certificate that you want to use.

See also: [sign\\_files](#) (14), [private\\_keystore\\_name](#) (14), [private\\_keystore\\_pass](#) (14), [private\\_certificate\\_pass](#) (15).

## **private\_certificate\_password**

Syntax: `-private_certificate_password`

Use the `-private_certificate_password` qualifier to specify the password of your private key certificate. Your private key certificate is required to digitally sign files.

See also: [sign\\_files](#) (14), [private\\_keystore\\_name](#) (14), [private\\_keystore\\_pass](#) (14), [private\\_certificate\\_subject](#) (15).

## Qualifiers for Sending Packages as Zip Files

You can send packages as zip files using the qualifiers described below. These qualifiers should be used together with the other `send` qualifiers, described in the rest of this chapter.

---

**Note:** When sending the package as a zip file, you should replace the optional `-delete_package` and `-package_directory` qualifiers with the optional `-delete_zip_file` and `-zip_file_name=full_path` qualifiers described below.

---

### package\_format

Syntax: `-package_format=zip`

Use the `-package_format` qualifier to send a zip file (containing the package) instead of sending a package. If you omit this qualifier, the files will be sent as a package. This qualifier can be used together with the `-zip_file_name` qualifier described below.

### zip\_file\_name

Syntax: `-zip_file_name=[full_path]|file_name`

Example: `-zip_file_name=upload`

The `-zip_file_name` qualifier is an optional qualifier that you can use to specify the name or full path of the zip file (containing the package) that you want to send. If you do not specify a path (i.e. you only specify a name), the zip file will be created in your current working directory. If you omit the `-zip_file_name` qualifier, the zip file will be created in your current working directory and assigned a temporary system name.

---

**Note:** When using the `-zip_file_name` qualifier, it is not necessary to append the `.zip` extension to the file name.

---

### delete\_zip\_file

Syntax: `-delete_zip_file=always|never|success|failure`

Example: `-delete_zip_file=success`

Use the `-delete_zip_file` qualifier to specify when, if at all, to delete the zip file from the directory that you specified with the `-zip_file_name` qualifier. If you omit this qualifier, the zip file will be deleted after it is sent. If you did not specify a directory with the `-zip_file_name` qualifier (i.e. you only specified a name), the zip file will be created in your current working directory.

The `-delete_zip_file` qualifier accepts the following values:

|                      |   |
|----------------------|---|
| <code>always</code>  | Always delete the zip file (the default).           |
| <code>never</code>   | Never delete the zip file.                          |
| <code>success</code> | Delete the zip file if it is successfully uploaded. |
| <code>failure</code> | Delete the zip file if the upload fails.            |

## Example Command

```
C:\Documents and Settings>java -jar "C:\Program Files\RepliWeb\RMFT
b-hub\bin\bhub_cli.jar" send -user=User2 -pass=22 -server=localhost
-recipients=user1 -files=c:\temp\a.txt -zip_file_name=rmftzip
-package_format=zip -delete_zip=never
```

## Other ‘Send’ Qualifiers

### compare\_files

The `-compare_files` qualifier verifies that the file has not be altered during transit by comparing the size and contents of the original source file with the transferred file.

### abort\_on\_transfer\_error

The `-abort_on_transfer_error` qualifier aborts the transfer if one of the files cannot be uploaded. If you use this qualifier and a transfer error occurs, no files will be uploaded.

### notify

**Syntax:** `-notify=[arrived_in_recipient_inboxes|delivered_to_hosts|opened_or_downloaded]`

**Example:** `-notify=arrived_in_recipient_inboxes,delivered_to_hosts`

Use the `-notify` qualifier if you want to be notified about the status of the package after it is sent. The notification will be sent to the email address specified in your RMFT account settings. The notification options are as follows:

| Value                                     | Description   |
|---|---|
| <code>arrived_in_recipient_inboxes</code> | You will be notified when the package has been distributed to all of the recipients' inboxes.   |
| <code>delivered_to_hosts</code>           | You will be notified when the package has been delivered to all of the hosts.   |
| <code>opened_or_downloaded</code>         | You will be notified each time any of the recipients open the package or downloads any of the files. If you specify this option, you will also receive a final status report before the package expires, informing you which recipients did not open the package, which recipients did not download any of the files, and which recipients only downloaded some of the files (and which files). |

**Note:** Because each qualifier value is unique, the example above can also be written: `-notify=ar,de`

---

## priority

Syntax: `-priority=low/normal/high`

Default: `-priority=normal`

Example: `-priority=low`

Use the `-priority` qualifier to indicate the level of package importance to RMFT Server and the package recipients. If you specify `-priority=high` RMFT Server will process your package before packages marked as low priority.

## expiration

Syntax: `-expiration=<minutes>`

Example: `-expiration=20000`

The `-expiration` qualifier specifies the date from which the package will cease to be available for downloaded.

## proxy\_server

**Syntax:** `-proxy_server=<address>`

**Example:** `-proxy_server=123.123.123.12`

Use this qualifier to connect to the RMFT Server machine via a proxy server. Replace `<address>` with the IP address or host name of your proxy server.

## proxy\_port

**Syntax:** `-proxy_port=<port_number>`

Use this qualifier to override the proxy server default port (8080) when connecting to the RMFT Server machine via a proxy server.

## package\_directory

Syntax: `-package_directory=full_path`

Use the `-package_directory` qualifier to specify the name of the directory in which you want to temporarily (i.e. before being sent) or permanently (i.e. after it is sent) save the package metadata. Use the `-delete_package` qualifier to determine whether the package metadata will be temporarily or permanently saved. If you do not specify the `-package_directory` qualifier, the package metadata will be temporarily/permanently saved to your current working directory.

This qualifier should be replaced with the `-zip_file_name` qualifier if you are sending the package as a zip file.

---

**Note:** The `-package_directory` qualifier only saves the package metadata. If you also want to temporarily/permanently save the source files, use the `-copy_files` qualifier as well. However, when the `-package_directory` qualifier is used with the `-offline` qualifier, the entire package is saved without needing to specify the `-copy_files` qualifier.

---

See also: [copy\\_files](#) (20), [delete\\_package](#) (20) and [zip\\_file](#) (16).

## copy\_files

Use the `-copy_files` qualifier to copy source files to the directory specified by the `-package_directory` qualifier. The files will be temporarily or permanently saved according to the value specified with the `-delete_package` qualifier.

---

**Note:** The package is a directory with a unique name beginning with your RMFT user name and followed by a unique ID, e.g., `user1_LSGNS58DN5ZFJK65KJIFXQOWQB`. The source files reside in the package's **Files** subdirectory.

---

See also: [package\\_directory](#) (20) and [delete\\_package](#) (20).

## delete\_package

Syntax: `-delete_package=always|never|success|failure`

Use the `-delete_package` qualifier to specify when, if at all, to delete the package from the temporary package directory. If you did not specify the `-package_directory` qualifier, the temporary package directory is your home directory appended with **Softlink\bhub\temp**.

This qualifier should be replaced with the `-delete_zip_file` qualifier if you are sending the package as a zip file.

|                      |   |
|----------------------|---|
| <code>always</code>  | Always delete the package.                                  |
| <code>never</code>   | Never delete the package.                                   |
| <code>success</code> | Delete the package if the package is successfully uploaded. |
| <code>failure</code> | Delete the package if the package upload fails.             |

See also: [copy\\_files](#) (20), [package\\_dir](#) (20) and [delete\\_zip\\_file](#) (16).

## offline

Syntax: `-offline`

Use this qualifier to create a package that you do not want to send immediately. The package will be saved to the directory specified by the `-package_directory` qualifier. You can send the package later using the [submit](#) command with the `-package_directory` qualifier.

## Example

```
C:\Documents and Settings>java -jar "C:\Program Files\RepliWeb\RMFT\b-  
hub\bin\bhub_cli.jar" send -server=bhub.server -user=charliebrow  
n  
-files="c:\aaa.xml,c:\temp\a.a" -recipients=yoel@company.com  
-offline -package_directory="c:\offline packages"
```

See also: [Submitting an Offline Package](#) (24), [package\\_dir](#) (20)

## transport

Syntax: `-transport=litecopy|fastcopy`

Example: `-transport=litecopy`

Use this qualifier to specify which transport protocol you want to use to upload files to RMFT Server: FASTCopy or LiteCopy. If you omit this qualifier, files will be uploaded using RMFT 's LiteCopy protocol.

**IMPORTANT** If you want to upload files using FASTCopy, you must first install FASTCopy on your computer. Currently, FASTCopy transport using the RMFT Java CLI is only supported on UNIX systems.

If you specify `-transport=fastcopy`, you can also specify FASTCopy qualifiers with the `-fcopy_qual`s qualifier described below. FASTCopy has over 250 qualifiers (features) for controlling and optimizing transfer.

For a complete explanation of all FASTCopy qualifiers, refer to the *FASTCopy Reference Guide*.

## fastcopy\_qualifiers

**Syntax:** `-fastcopy_qual="qualifier qualifier"`

**Example:** `-fastcopy_qual="-ack_interval=1000 -max_window=1000"`

Use this qualifier to specify any additional FASTCopy qualifiers that you want to use. Qualifiers must be separated by a space and enclosed within quotation marks. You can only use this qualifier together with the `-transport=fastcopy` qualifier described above.

**IMPORTANT** Currently, FASTCopy transport using the RMFT Java CLI is only supported on UNIX systems.

### Example:

The following command uploads files to RMFT Server using FASTCopy. It also includes the FASTCopy qualifiers `-ack_interval` and `-max_window` for optimization of transfer speed. For more information on these and other FASTCopy qualifiers, refer to the *FASTCopy Reference Guide*.

```
>java -jar bhub_cli.jar send -server=192.247.160.82 -user=a -pass=
a -recipient=a -transport=fastcopy -files=/opt/qa/r1_aix43.tar
-fastcopy_qualifiers="-ack_interval=1000 -max_window=1000"
```

## relative\_server\_url

**Note** This qualifier should only be used if instructed by your RMFT Administrator who will also provide you with the required qualifier value.

**Syntax:** `-relative_server_url=<virtual_directory>`

**Example:** `-relative_server_url=transfer`

Use the `-relative_server_url` qualifier to override the default path that follows the RMFT Server IP address or host name.

## hide\_recipients

**Syntax:** -hide\_recipients

When sending a package to multiple recipients, you can use the `-hide_recipients` qualifier to prevent the package recipients from seeing who else the package was sent to.

## allow\_nonascii\_filenames

**Syntax:** -allow\_nonascii\_filenames

Use the `-allow_nonascii_filenames` qualifier if the filenames of the source files contain non-ascii characters (e.g. Chinese). If you omit this qualifier and the filenames contain non-ascii characters, the transfer will fail.

## unauthenticated\_download

**Syntax:** -unauthenticated\_download

Use the `-unauthenticated_download` qualifier to allow recipients to download files without logging in.

**Note:** Selecting this option makes it easier for the recipients to download the files, but it will also allow anyone with access to the recipients' computers to download the package files (since login is not required). Therefore, it is advisable to only select this option if the files that you are sending do not contain any confidential information.

## Submitting an Offline Package

Use the *submit* command to send offline packages or zipped packages. Offline packages are packages that have been saved on your computer using the `-offline` qualifier.

The *submit* command must be followed by login qualifiers (required for logging into RMFT Server) and the `-package_directory` qualifier or `-` in the case of offline zip files - the `-package_format` and `-zip_file_name` qualifiers. Other qualifiers such as the `-recipients` qualifier are not required, since the package already contains this information.

---

**Note:** The qualifiers (i.e. credentials) that you need to use to log into RMFT Server are organization-specific. Some organizations will require you to log in using a user name and password, while others will require you to log in using a certificate. If you are unsure of how to log into RMFT Server, contact your RMFT administrator.

---

### Example of How to Submit an Offline Package

```
~RepliWeb\RMFT\b-hub\bin>java submit -server=212.29.222.82 -  
user=charliebrown -pass=peanuts  
-package_directory=c:\bhub\temp\4\mike_8S6IPVK0Q8YO394U0N1J51L1L1
```

See also: [offline](#) (21) and [package\\_directory](#) (25)

### Example of How to Submit an Offline Zipped Package

```
~RepliWeb\RMFT\b-hub\bin>java -jar bhub_cli.jar submit -server=localhost  
-user=user1 -pass=22 -package_format=zip -zip_file_name=c:\temp\file.zip
```

See also: [Qualifiers for Sending Packages as Zip Files](#) (16).

## **package\_directory**

**Syntax:** `-package_directory=full_path`

**Example:**

```
-package_directory=c:\bhub\temp\4\mike_8S6IPVK0Q8YO394U0N1J51L1
```

Use the `-package_directory` qualifier to specify the full path of the offline package to be sent.

## 2. Searching for Packages

This chapter describes qualifiers that you can use to search for packages on RMFT Server. All search qualifiers must be preceded by the search command **find**.

The command syntax is as follows:

```
C:\Program Files\RepliWeb\RMFT\b-hub\bin>java -jar bhub_cli.jar find
<qualifiers>
```

### Example

The following command will display a detailed list of all packages in the user's inbox.

```
C:\Program Files\RepliWeb\RMFT\b-hub\bin>java -jar bhub_cli.jar find
-server=123.123.123.12 -user=user1 -password=g245t60
-format=details -folder=inbox
```

## server

**Syntax:** `-server=<machine name/IP address>`

**Example:** `-server=123.123.123.23`

Use the `-server` qualifier to specify the name or IP address of the RMFT Server to search.

See also: [user](#) (27), [password](#) (27).

## relative\_server\_url

**Note** This qualifier should only be used if instructed by your RMFT Administrator who will also provide you with the required qualifier value.

**Syntax:** `-relative_server_url=<virtual_directory>`

**Example:** `-relative_server_url=transfer`

Use the `-relative_server_url` qualifier to override the default path that follows the RMFT Server IP address or host name.

## user

**Syntax:** `-user=<b-hub_user>`

**Example:** `-user=dan3`

Use the `-user` qualifier to specify your RMFT user name (assigned by the RMFT administrator).

See also: [server](#) (27), [password](#) (27).

## password

**Syntax:** `-password=<password>`

Use the `-password` qualifier to specify your RMFT password (assigned by the RMFT administrator). Your RMFT administrator should provide you with instructions for logging into RMFT Server such as whether to log in using a password or certificate. If you are required to log in using a certificate (i.e. without a password), see [Certificate-Based Login](#) Qualifiers (34).

See also: [server](#) (27), [user](#) (27).

## downloaded

**Syntax:** `-downloaded=[true|false]`

**Example:** `-downloaded=false`

Specify `-downloaded=false` to exclude downloaded packages from your search results. To show a list of all packages including those that have already been downloaded (the default), omit this qualifier.

## secure

**Syntax:** `-secure`

Use the `-secure` qualifier to establish a secure connection (SSL) to RMFT Server.

To utilize this option, the appropriate certificate must be installed on the IIS server. Contact the RMFT administrator to verify whether or not a server-side certificate exists. In addition, you must import the CA (Certificate Authority) certificate to your Java `cacerts` file.

## proxy\_server

**Syntax:** `-proxy=<address>`

**Example:** `-proxy=<123.123.123.12>`

Use this qualifier to connect to the RMFT Server machine via a proxy server. Replace `<address>` with the IP address or host name of your proxy server.

## proxy\_port

**Syntax:** `-proxy=<port_number>`

Use this qualifier to override the proxy server default port (8080) when connecting to the RMFT Server machine via a proxy server.

## files

**Syntax:** `-files=<string>`

Use the `-files` qualifier to specify a string or the name of a specific file that you want to search for. All packages containing files that match the specified string or file name will be included in the search results. A string can contain a combination of standard and wildcard characters. For example, if you specify `-files=p*. *` all packages containing files that begin with the letter "p" will be included in the search results.

## format

**Syntax:** `-format=id|list|details|files`

**Example:** `-format=files`

Use the `-format` qualifier to specify the display format of the search results.

Select one the following values as appropriate:

- `id` - displays list of package IDs
- `list` (the default) - displays a list of packages (no file names).
- `details` - displays a detailed list of packages with messages and file names.
- `files` - displays a list of package IDs and file names.

## xml

**Syntax:** `-xml`

Use the `-xml` qualifier to display the search results in XML format.

## output

**Syntax:** `-output=<file name>`

**Example:** `-output=c:\search.txt`

Use the `-output` qualifier to write the search results to a file.

## folder

**Syntax:** `-folder=inbox|sent|deleted`

**Example:** `-folder=inbox`

Use the `-folder` qualifier to specify which folder you want to search.

Select one the following values as appropriate:

- `inbox`
- `sent`
- `deleted`

## after

**Syntax:** `-after=<since date>`

Use the `-after` qualifier to only search for packages with dates later than the specified date. You can use this qualifier together with the `-before` qualifier to search for packages between the two dates. The date and time format must conform to the syntax described in [Appendix A: Time Expressions](#) (55).

## before

**Syntax:** `-before=<before date>`

Use the `-before` qualifier to only search for packages with dates earlier than the specified date. You can use this qualifier together with the `-after` qualifier to search for packages between the two dates. The date and time format must conform to the syntax described in [Appendix A: Time Expressions](#) (55).

## subject

**Syntax:** `-subject=<string>`

Use the `-subject` qualifier to only search for packages whose subjects match the specified string. The string can contain part or all of the subject.

## from

**Syntax:** `-from=<user name>`

Use the `-from` qualifier to only search for packages from the specified sender.

## to

**Syntax:** `-to=<user name>`

Use the `-to` qualifier to only search for packages sent to the specified recipient.

## message

**Syntax:** `-message=<string>`

Use the `-message` qualifier to only download packages whose messages match the specified message string. The string can contain part or all of the message.

## read

**Syntax:** `-read=true|false`

Use the `-read` qualifier to only search for opened (the default) or unopened packages. Specify `-read=false` to search for unopened packages.

## encrypted

**Syntax:** `-encrypted=true|false`

Use the `-encrypted` qualifier to only search for encrypted (the default) or unencrypted packages. Specify `-encrypted=false` to search for unencrypted packages.

## signed

**Syntax:** `-signed=true|false`

Use the `-signed` qualifier to only search for signed (the default) or unsigned packages. Specify `-signed=false` to search for unsigned packages.

## expired

**Syntax:** `-expired=true|false`

Use the `-expired` qualifier to only search for expired (the default) or unexpired packages. Specify `-expired=false` to search for packages that have not reached their expiry date.

## expired\_since

**Syntax:** `-expired_since=<expiry date>`

Use the `-expired_since` qualifier to only search for packages with expiry dates after the specified date. You can use this qualifier together with the `-expired_before` qualifier to search for all packages with expiry dates between the two dates. The date and time format must conform to the syntax described in [A. Time Expressions](#) (56).

## expired\_before

**Syntax:** `-expired_before=<expiry date>`

Use the `-expired_before` qualifier to only search for packages with expiry dates before the specified date. You can use this qualifier together with the `-expired_since` qualifier to search for all packages with expiry dates between the two dates. The date and time format must conform to the syntax described in [A. Time Expressions](#) (56).

## minimum\_size

**Syntax:** `-minimum_size=<size in bytes>`

**Example:** `-minimum_size=245213`

Use the `-minimum_size` qualifier to only search for packages larger than the specified size. You can use this qualifier together with the `-maximum_size` qualifier to search for all packages between the specified minimum and maximum packages sizes.

## maximum\_size

**Syntax:** `-maximum_size =<size in bytes>`

**Example:** `-maximum_size =454232`

Use the `-maximum_size` qualifier to only search for packages smaller than the specified size. You can use this qualifier together with the `-minimum_size` qualifier to search for all packages between the specified minimum and maximum packages sizes.

## trace

**Syntax:** `-trace=<trace_value>`

**Example:** `-trace=all`

You can troubleshoot operations that repeatedly fail by adding the `-trace` qualifier to your command line. This qualifier accepts numerous values and should only be used after consulting with [RepliWeb Support](#).

## Certificate-Based Login Qualifiers

Your RMFT administrator may require you to log in to RMFT Server using a certificate. If this is the case, use the qualifiers below to specify the name and password of your certificate keystore.

### keystore\_name

Syntax: `-keystore_name=<keystore_full_path>`

Example: `-keystore_name=c:\ph\certs\ph.jks`

Use the `-keystore_name` qualifier to specify the location of your Java keystore. The name of your Java keystore is required when using a certificate to authenticate your identity to RMFT Server.

The keystore that you specify may contain several different client certificates. Normally, Java will find a keystore certificate that is also accepted by the server.

If the IIS server is configured to **Require client certificates** and your keystore does not contain a valid certificate, your certificate will be rejected with the following error:

```
HTTP error 403 Access Forbidden
```

The recommended solution is to create a keystore containing one certificate and use it exclusively for authenticating to RMFT Server.

See also: [keystore\\_password](#) (34), [secure](#) (28)

### keystore\_password

Syntax: `-keystore_password`

The `-keystore_password` qualifier specifies the password for accessing your keystore.

See also: [keystore\\_name](#) (34), [secure](#) (28).

# 3. Downloading Packages

The following chapter describes qualifiers that you can use to download packages from RMFT Server. All download qualifiers must be preceded by the **download** command.

---

**Note:** Currently, you can only download packages from your inbox. In future versions, support will be added to enable download from any folder.

---

The syntax is as follows:

```
C:\Program Files\RepliWeb\RMFT\b-hub\bin>java -jar bhub_cli.jar
download <qualifiers>
```

## Example

The following command will download all packages from the user's inbox to the local directory **c:\target**.

```
C:\Program Files\RepliWeb\RMFT\b-hub\bin>java -jar bhub_cli.jar
download
-server=<123.123.123.12> -user=user1 -password=g245t60
-package_directory=c:\target
```

Mandatory download qualifiers are: [server](#) (36), [user](#) (36), [password](#) (36) and [package\\_directory](#) (37).

## server

**Syntax:** `-server=<machine name/IP address>`

**Example:** `-server=123.123.123.23`

Use the `-server` qualifier to specify the name/IP address of the RMFT Server from which you want to download packages.

See also: [user](#) (36), [password](#) (36), [package\\_directory](#) (37)

## relative\_server\_url

**Note** This qualifier should only be used if instructed by your RMFT Administrator who will also provide you with the required qualifier value.

**Syntax:** `-relative_server_url=<virtual_directory>`

**Example:** `-relative_server_url=transfer`

Use the `-relative_server_url` qualifier to override the default path that follows the RMFT Server IP address or host name.

## user

**Syntax:** `-user=<RMFT _user>`

**Example:** `-user=dan3`

Use the `-user` qualifier to specify your RMFT user name (assigned by the RMFT administrator).

See also: [server](#) (36), [password](#) (36), [package\\_directory](#) (37).

## password

**Syntax:** `-password=<password>`

Use the `-password` qualifier to specify your RMFT password (assigned by the RMFT administrator). Your RMFT administrator should provide you with instructions for logging into RMFT Server such as whether to log in with a password or with a certificate. If you are required to log in using a certificate (i.e. without a password), see [Certificate-Based Login Qualifiers](#) (43).

See also: [server](#) (36), [user](#) (36), [package\\_directory](#) (37).

## secure

**Syntax:** `-secure`

Use the `-secure` qualifier to establish a secure connection (SSL) to RMFT Server.

To utilize this option, the appropriate certificate must be installed on the IIS server. Contact the RMFT administrator to verify whether or not a server-side certificate exists. In addition, you must import the CA (Certificate Authority) certificate to your certificate store.

## proxy\_server

**Syntax:** `-proxy=<address>`

**Example:** `-proxy=<123.123.123.12>`

Use this qualifier to connect to the RMFT Server machine via a proxy server. Replace `<address>` with the IP address or host name of your proxy server.

## proxy\_port

**Syntax:** `-proxy=<port_number>`

Use this qualifier to override the proxy server default port (8080) when connecting to the RMFT Server machine via a proxy server.

## package\_directory

**Syntax:** `-package_directory=<target_directory>`

**Example:** `-package_directory=c:\target`

Use the `-package_directory` qualifier to specify the name of the directory to which you want to download the package(s).

See also: [server](#) (36), [user](#) (36), [password](#) (36), [package\\_directory](#) (37).

## package\_id

**Syntax:** `-package_id=<package_id>`

**Example:** `-package_id=user_100F40ARUNQJW6QY32UUFKQVOT`

Use the `-package_id` qualifier to specify the ID of the package that you want to download or of the package containing specific files that you want to download.

See also: [specific\\_files](#) (38).

## files

**Syntax:** `-files=<string>`

Use the `files` qualifier to specify a string or the name of a specific file that you want to download. All packages containing files that match the specified string or file name will be downloaded. Note that any other files in the package will also be downloaded with the package. A string can contain a combination of standard and wildcard characters. For example, if you specify `-files=p*. *` all packages containing files that begin with the letter "p" will be downloaded. You must also specify a target directory using the `-package_directory` qualifier.

## compare\_files

The `-compare_files` qualifier verifies that the file has not be altered during transit by comparing the size and contents of the original source file with the transferred file.

## specific\_files

**Syntax:** `-specific_files=document.pdf`

Use the `-specific_files` qualifier to specify the name of files that you want to download from a specific package. You can also specify wildcards to download any matching files, but file names and wildcards cannot be specified together.

The `-specific_files` qualifier must be used with the `-package_id` qualifier.

See also: [package\\_id](#) (37).

## after

**Syntax:** `-after=<date>`

Use the `-after` qualifier to only download packages with dates later than the specified date. You can use this qualifier together with the `-before` qualifier to download packages dated between the two dates. The date and time format must conform to the syntax described in [A. Time Expressions](#) (56).

## before

**Syntax:** `-before=<before date>`

Use the `-before` qualifier to only download packages with dates earlier than the specified date. You can use this qualifier together with the `-after` qualifier to download packages dated between the two dates. The date and time format must conform to the syntax described in [A. Time Expressions](#) (56).

## subject

**Syntax:** `-subject=<string>`

Use the `-subject` qualifier to only download packages whose subjects match the specified string. The string can contain part or all of the subject.

## from

**Syntax:** `-from=<user name>`

Use the `-from` qualifier to only download packages from the specified sender.

## to

**Syntax:** `-to=<user name>`

Use the `-to` qualifier to only download packages sent to the specified recipient.

## message

**Syntax:** `-message=<string>`

Use the `-message` qualifier to only download packages whose messages match the specified message string. The string can contain part or all of the message.

## read

**Syntax:** `-read=true|false`

Use the `-read` qualifier to only download opened (the default if you specify the qualifier without a value) or unopened packages. Specify `-read=false` to download unopened packages.

## encrypted

**Syntax:** `-encrypted=true|false`

Use the `-encrypted` qualifier to only download encrypted (the default if you specify the qualifier without a value) or unencrypted packages. Specify `-encrypted=false` to download unencrypted packages.

## signed

**Syntax:** `-signed=true|false`

Use the `-signed` qualifier to only download signed (the default if you specify the qualifier without a value) or unsigned packages. Specify `-signed=false` to download unsigned packages.

## expired\_since

**Syntax:** `-expired_since=<expiry date>`

Use the `-expired_since` qualifier to only download packages with expiry dates after the specified date. You can use this qualifier together with the `-expired_before` qualifier to download all packages with expiry dates between the two dates. The date and time format must conform to the syntax described in [A. Time Expressions](#) (56).

## expired\_before

**Syntax:** `-expired_before=<expiry date>`

Use the `-expired_before` qualifier to only download packages with expiry dates before the specified date. You can use this qualifier together with the `-expired_since` qualifier to download all packages with expiry dates between the two dates. The date and time format must conform to the syntax described in [A. Time Expressions](#) (56).

## minimum\_size

**Syntax:** `-minimum_size=<size in bytes>`

**Example:** `-minimum_size=245213`

Use the `-minimum_size` qualifier to only download packages larger than the specified size. You can use this qualifier together with the `-maximum_size` qualifier to download all packages between the specified maximum and minimum package sizes.

## maximum\_size

**Syntax:** `-maximum_size =<size in bytes>`

**Example:** `-maximum_size =454232`

Use the `-maximum_size` qualifier to only download packages smaller than the specified size. You can use this qualifier together with the `-minimum_size` qualifier to download all packages between the specified maximum and minimum package sizes.

### **abort\_on\_transfer\_error**

The `-abort_on_transfer_error` qualifier aborts the transfer if one of the files cannot be downloaded. If you use this qualifier and a transfer error occurs, no files will be downloaded.

### **trace**

**Syntax:** `-trace`

You can troubleshoot operations that repeatedly fail by adding the `-trace` qualifier to your command line. This qualifier should only be used after consulting with [RepliWeb Support](#).

### **post\_command**

**Syntax:** `-post_command=Process.bat/Process.exe/Process.lua`

**Example:** `-post_command=rename.bat`

Use the `-post_command` qualifier to process downloaded files according to your needs (e.g. rename, move to another directory, etc.).

### **post\_command\_parameters**

**Syntax:** `-post_command_parameters="param1 param2"`

**Example:** `-post_command_parameters="$(PACKAGE_ID) downloaded_$(PACKAGE_ID)"`

Use the `-post_command_parameters` qualifier to specify any parameters required by the process specified with the `-post_command` qualifier. You can also specify the RMFT variable `$(PACKAGE_ID)` as one of the parameters. The variable is replaced with the actual package ID during runtime.

### **server\_open**

**Syntax:** `-server_open`

Use the `-server_open` qualifier to establish an open-ended session with RMFT Server. You must use this qualifier with the `-check_inbox_interval` qualifier described below.

## check\_inbox\_interval

**Syntax:** `-check_inbox_interval=interval_in_seconds`

**Example:** `-check_inbox_interval=300`

Use the `-check_inbox_interval` qualifier to specify how often you want to check for new packages in your RMFT Server inbox. New packages will be downloaded automatically. You must use the `-check_inbox_interval` qualifier together with the `-server_open` qualifier described above.

### Example:

The following command checks mike's inbox every 5 minutes for new packages. It downloads any new packages that it finds to `c:\target`.

```
>java -jar bhub_cli.jar download
-server=<123.123.123.12> -user=mike -password=g245t60
-package_directory=c:\target -server_open -check_inbox_interval=300
```

## Certificate-Based Login Qualifiers

Your RMFT administrator may require you to log in to RMFT Server using a certificate. If this is the case, use the qualifiers below to specify the name and password of your certificate keystore.

### **keystore\_name**

Syntax: `-keystore_name=<keystore_full_path>`

Example: `-keystore_name=c:\ph\certs\ph.jks`

Use the `-keystore_name` qualifier to specify the location of your Java keystore. The name of your Java keystore is required when using a certificate to authenticate your identity to RMFT Server.

The keystore that you specify may contain several different client certificates. Normally, Java will find a keystore certificate that is also accepted by the server.

If the IIS server is configured to **Require client certificates** and your keystore does not contain a valid certificate, your certificate will be rejected with the following error:

```
HTTP error 403 Access Forbidden
```

The recommended solution is to create a keystore containing one certificate and use it exclusively for sending files to RMFT Server.

See also: [secure](#) (37).

### **keystore\_password**

Syntax: `-keystore_password`

The `-keystore_password` qualifier specifies the password for accessing your keystore.

See also: [secure](#) (37).

# 4. Scrambling Passwords

You can scramble both public keystore passwords and private keystore passwords as well as any passwords for certificates residing in these keystores. You can also scramble your RMFT login password. A password scrambled using the RMFT Java CLI **scramble** command can only be used with the RMFT Java CLI.

The command for scrambling any type of password (login, keystore, and so on) is **scramble** followed by the `-password` qualifier.

## Example:

```
java -jar bhuh_cli.jar scramble -password=mypass
```

After scrambling the password, you can use it according to your needs.

## Scrambled Password Qualifiers

### scrambled\_login\_password

Use the `-scrambled_login_password` qualifier to specify your scrambled password for logging into RMFT Server.

### scrambled\_keystore\_password

Use the `-scrambled_keystore_password` qualifier to specify your scrambled keystore password for authenticating your identity to RMFT Server.

### scrambled\_public\_keystore\_password

Use the `-scrambled_public_keystore_password` qualifier to specify your scrambled public keystore password (for encrypting and verifying files).

### **scrambled\_public\_certificate\_password**

Use the `-scrambled_public_certificate_password` qualifier to specify your scrambled public key password (for encrypting and verifying files).

### **scrambled\_private\_keystore\_password**

Use the `-scrambled_private_keystore_password` qualifier to specify your scrambled private keystore password (for signing and decrypting files).

### **scrambled\_private\_certificate\_password**

Use the `-scrambled_private_certificate_password` qualifier to specify your scrambled private key password (for signing and decrypting files).

See also: [Verifying and Decrypting Files](#) (46).

# 5. Verifying and Decrypting Files

This chapter describes the `process` command which you can use to decrypt files and verify the signatures of digitally signed files. The `process` command can be followed by various decryption and verification qualifiers, depending on whether you want to decrypt or verify files (or both). A single `process` command can include qualifiers that will both decrypt the files and verify their digital signatures. However, you may find it more convenient to submit two separate commands, one for decryption and one for signature verification.

## Example Command:

The command below will decrypt and verify the digital signatures of all files in the directory `c:\received`.

```
C:\Program Files\RepliWeb\RMFT\b-hub\bin>java -jar bhub_cli.jar process
-decrypt -public_certificate_subject=charlie@company.com
-private_certificate_password=34f5478 -private_keystore_name=c:\certs\ph.jks
-private_keystore_password=u3ee23r -verify
-public_keystore_name=c:\cb\certs\cb.jks -public_keystore_password=34ad345
-public_certificate_subject=john@corp.com -source_directory=c:\received
```

This chapter is divided into the following sections:

- [Verifying Files](#) (47)
- [Decrypting Files](#) (51)
- [Shared Decryption and Verification Qualifiers](#) (54)

## Verifying Files

This section describes qualifiers that you can use to verify the signatures of digitally signed files.

Qualifiers for verifying digital signatures are as follows:

- [verify](#) (48)
- [public keystore name](#) (48)
- [public keystore pass](#) (48)
- [public certificate subject](#) (48)
- [package id](#) (49)
- [signature file](#) (49)
- [signature directory](#) (50)

Signature verification qualifiers must follow the **process** command and be used together with the file location qualifiers described in [Shared Encryption and Verification Qualifiers](#) (54).

### Example Command:

The command below will verify the digital signatures of all files in the directory **c:\received**.

```
C:\Program Files\RepliWeb\RMFT\b-hub\bin>java -jar bhub_cli.jar process
-verify -private_certificate_subject=jn@corp.com -
private_keystore_name=c:\certs\cb.jks -private_keystore_password=34ad345
-source_directory=c:\received
```

## verify

**Syntax:** `-verify`

Use the mandatory `-verify` qualifier to verify digital signatures. You can position the `-verify` qualifier anywhere in the command line following the **process** command.

See also: [public\\_keystore\\_name](#) (48), [public\\_keystore\\_password](#) (48), [public\\_certificate\\_subject](#) (48).

## public\_keystore\_name

**Syntax:** `-public_keystore_name=<keystore_full_path>`

**Example:** `-public_keystore_name=c:\ph\certs\ph.jks`

Use the mandatory `-public_keystore_name` qualifier to specify the full path of the Java keystore containing the signer's public key certificate.

See also: [verify](#) (48), [public\\_keystore\\_password](#) (48), [public\\_certificate\\_subject](#) (48).

## public\_keystore\_password

**Syntax:** `-public_keystore_password=<keystore_password>`

Use the mandatory `-public_keystore_password` qualifier to specify the password of the Java keystore containing the signer's public key certificate.

See also: [verify](#) (48), [public\\_keystore\\_name](#) (48), [public\\_certificate\\_subject](#) (48).

## public\_certificate\_subject

**Syntax:** `-public_certificate_subject=<certificate_subject>`

**Example:** `-public_certificate_subject=info@company.com`

Use the mandatory `-public_certificate_subject` qualifier to specify a unique part of the subject of the signer's public key certificate.

See also: [verify](#) (48), [public\\_keystore\\_password](#) (48), [public\\_keystore\\_name](#) (48).

## package\_id

**Syntax:** `-package_id=<package_id>`

**Example:** `-package_id=charliebrowne_8S6IPVK0Q8YO394U0N1J51L1L1`

The `-package_id` qualifier is an optional qualifier that can be used to verify the package ID of the package that contains (if the files were delivered as an RMFT package) or originally contained the signed files (before they were downloaded/forwarded from RMFT Server to your computer).

Since the package ID forms part of the file signature, omitting the `-package_id` qualifier will result in a partial verification of the digital signature. Thus, if you omit this qualifier, you will be asked whether you want to continue the verification process.

The `-package_id` qualifier must be used with the `-public_keystore_name`, `-public_keystore_password` and `-public_certificate_subject` qualifiers. You must also specify the full path of the signed files using the `-file_spec`, `-source_directory`, or `-files` qualifiers described in [Shared Decryption and Verification Qualifiers](#) (54).

---

**Note:** If you download signed files from RMFT Server (using RMFT Web Client), remember to make a note of the package ID.

---

See also: [verify](#) (48), [public keystore password](#) (48), [public keystore name](#) (48), [public certificate subject](#) (48).

## signature\_file

**Syntax:** `-signature_file=full_path[,full_path]`

**Example:**

```
-signature_file="c:\new tax\accounts.doc.sig"
```

Usually, this qualifier is not required as the digital signature is embedded in the source file. However, in coordination with the RMFT system administrator, you can opt to receive the digital signatures as separate files. The `-signature_file` or `-signature_directory` qualifier should be used in these cases.

Use the `-signature_file` qualifier to specify the name of a signature file (signature files are appended with the **.sig** extension). The `-signature_files` qualifier should be used with the `-public_keystore_name`, `-signature_keystore_password`, `-public_certificate_subject` and `-file` qualifiers.

See also: [verify](#) (48), [public keystore pass](#) (48), [public keystore name](#) (48) [public certificate subject](#) (48).

## signature\_directory

**Syntax:** `-signature_directory=<directory_containing_signature_files>`

Usually, this qualifier is not required as the digital signature is part of the source file. However, in coordination with the RMFT system administrator, you can opt to receive the digital signatures as separate files. The `-signature_directory` or `-signature_files` qualifier should be used in these cases.

Use the `-signature_directory` qualifier to specify the name of the directory containing the signatures files. This *cannot* be the same directory as the digitally signed files.

---

**IMPORTANT:** The specified directory should only contain signature files (i.e. files with a **.sig** extension). If other files are present, the verification operation will fail.

---

The `-signature_directory` qualifier should be used with the `-public_keystore_name`, `-public_keystore_password`, `-public_certificate_subject` and `-source_directory` qualifiers.

See also: [verify](#) (48), [public keystore pass](#) (48), [public keystore name](#) (48) [public certificate subject](#) (48).

## Decrypting Files

This section describes qualifiers that you can use to decrypt files that have been encrypted using your public key or a shared secret.

Decryption qualifiers are as follows:

- [decrypt](#) (52)
- [key\\_id](#) (52)
- [private\\_keystore\\_pass](#) (52)
- [private\\_certificate\\_subject](#) (53)
- [private\\_certificate\\_pass](#) (53)
- [shared\\_secret](#) (53)
- [cipher](#) (53)

Decryption qualifiers must follow the **process** command and be used together with the file location qualifiers described in [Shared Decryption and Verification Qualifiers](#) (54).

### Example Command:

The command below will decrypt all files in the directory **c:\received**. The files will be decrypted and saved in the directory **c:\decrypted**.

```
C:\Program Files\RepliWeb\RMFT\b-hub\bin>java -jar bhub_cli.jar process
-decrypt -private_certificate_subject=bob@company.com
-private_certificate_pass=fw45 -private_keystore_name=c:\certs\ph.jks
-private_keystore_password=u3ee23r -source_directory=c:\received
-target_directory=c:\decrypted
```

## decrypt

**Syntax:** `-decrypt`

Use the mandatory `-decrypt` qualifier to decrypt encrypted files. You can position the `-decrypt` qualifier anywhere in the command line following the `process` command.

See also: [decrypt](#) (52), [key\\_id](#) (52), [private\\_keystore\\_name](#) (52), [private\\_keystore\\_pass](#) (52), [private\\_certificate\\_subject](#) (53), [private\\_certificate\\_pass](#) (53).

## key\_id

**Syntax:** `-key_id=<your RMFT user name>`

Use the `-key_id` qualifier to specify your RMFT user name when you receive a package that has been encrypted by RMFT Server using your public key certificate. If you are not sure whether the package was encrypted by the sender or by RMFT Server, contact your RMFT Server administrator.

See also: [decrypt](#) (52), [key\\_id](#) (52), [private\\_keystore\\_name](#) (52), [private\\_keystore\\_pass](#) (52), [private\\_certificate\\_subject](#) (53), [private\\_certificate\\_pass](#) (53).

## private\_keystore\_name

**Syntax:** `-private_keystore_name=<keystore_full_path>`

**Example:** `-private_keystore_name=c:\cb\certs\ph.jks`

Use the mandatory `-private_keystore_name` qualifier to specify the full path of the Java keystore containing your private key certificate.

See also: [decrypt](#) (52), [key\\_id](#) (52), [private\\_keystore\\_pass](#) (52), [private\\_certificate\\_subject](#) (53), [private\\_certificate\\_pass](#) (53).

## private\_keystore\_password

**Syntax:** `-private_keystore_password=<keystore password>`

Use the mandatory `-private_keystore_password` qualifier to specify the password of the Java keystore containing your private key certificate.

See also: [decrypt](#) (52), [key\\_id](#) (52), [private\\_keystore\\_name](#) (52), [private\\_keystore\\_pass](#) (52), [private\\_certificate\\_subject](#) (53), [private\\_certificate\\_pass](#) (53).

## private\_certificate\_subject

**Syntax:** `-private_certificate_subject=<certificate_subject>`

**Example:** `-private_certificate_subject=info@company.com`

Use the mandatory `-private_certificate_subject` qualifier to specify a unique part of the subject of your private key certificate.

See also: [decrypt](#) (52), [key\\_id](#) (52), [private\\_keystore\\_name](#) (52), [private\\_keystore\\_pass](#) (52), [private\\_certificate\\_pass](#) (53).

## private\_certificate\_password

**Syntax:** `-private_certificate_password=<certificate_password>`

**Example:** `-private_certificate_password=45YWS5Y6`

Use the mandatory `-private_certificate_password` qualifier to specify a unique part of the subject of your private key certificate.

See also: [decrypt](#) (52), [key\\_id](#) (52), [private\\_keystore\\_name](#) (52), [private\\_keystore\\_pass](#) (52), [private\\_certificate\\_subject](#) (53), [private\\_certificate\\_pass](#) (53).

## shared\_secret

**Syntax:** `-shared_secret=<shared_secret>`

**Example:** `-shared_secret=nw54ti9ve45v`

Use the `-shared_secret` qualifier to specify the password used to encrypt the files. For successful decryption, you must also specify the cipher used to encrypt the files using the `-cipher` qualifier.

See also: [cipher](#) (53)

## cipher

**Syntax:** `-cipher=<cipher>`

**Example:** `-cipher=rc4`

Use the `-cipher` qualifier to specify the cipher used to encrypt the files. For successful decryption, you must also specify the sender's encryption password (secret) using the `-shared_secret` qualifier.

See also: [shared\\_secret](#) (53)

## Shared Decryption and Verification Qualifiers

The qualifiers described in this section should be used when verifying digital signatures and/or decrypting files. You can use these qualifiers together with the `-verify` and `-decrypt` qualifiers *in a single command* to decrypt and verify files. Use of a single command to decrypt and verify files assumes that the received files are both encrypted and digitally signed.

These qualifiers must be used together with the relevant verification and/or decryption qualifiers described in [Verifying Files](#) (47) and [Decrypting Files](#) (51).

### file

**Syntax:** `-file=full_path[ ,full_path]`

**Example:** `-file="c:\new tax\accounts.pdf"`

Use the `-file` qualifier to specify the full path of a digitally signed/encrypted file.

### file\_spec

**Syntax:** `-file_spec=[full_path]file_spec`

**Example:** `-file_spec="c:\new tax\*.txt"`

Use the `-file_spec` qualifier to specify the extension of digitally signed/encrypted files of the same type. This qualifier eliminates the need to specify individual file names when verifying/decrypting several files with a common file type.

### source\_directory

**Syntax:** `-source_directory=<full_path>`

Use the `-source_directory` qualifier to specify the name of a directory that only contains digitally signed and/or encrypted files. If the directory contains files that have not been digitally signed and/or encrypted, the processing operation will fail.

Only files corresponding to the public key certificate that you specified with the `-public_certificate_subject` qualifier will be verified. Likewise, only files corresponding to the private key certificate that you specified with the `-private_certificate_subject` qualifier will be decrypted.

## target\_file

**Syntax:** `-target_file=<file_name>`

**Example:** `-file=old.doc -target_file=new.doc`

Use the `-target_file` qualifier to rename a single file after it is decrypted/verified. This qualifier must be used with the `-file` qualifier and can only be used when decrypting/verifying a single file.

## target\_directory

**Syntax:** `-target_directory=<directory_name>`

**Example:** `-source_directory=c:\temp -target_directory=c:\target`

Use the `-target_directory` qualifier to specify a target directory for the verified/decrypted files. The files will be verified/decrypted and saved in the specified target directory.

This qualifier must be used with the `-source_directory`, `-file_spec` or `-file` qualifiers described above.

# A. Time Expressions

The qualifiers `-before`, `-after`, `-expired_since` and `-expired_before` accept time expressions as values. Time expressions can be used to narrow the search for packages or to download packages received, sent or deleted during a specific time period.

## Absolute Time Expressions

On all supported systems, you can specify a date or time in the following formats:

### Time Format

| Format | Description   |
|--------|---|
| hh:mm  | Hours (two digits) and minutes (two digits). The current date is assumed. |

### Date Format

| Format | Description  |
|--------|--|
| ddmmyy | Day of the month (two digits), month (two digits) and year (two digits). The time is assumed to be midnight on the specified date. |

### Time and Date Format

| Format     | Description   |
|------------|---|
| ddmmyyhhmm | A combination of the two expressions. You can also use the "today" keyword. If no time is specified with a qualifier that requires a time expression, the time is assumed to be midnight and the date is assumed to be today. |

## Relative Time Expressions

Sometimes it is more useful to specify a time expression as relative to the current time rather than in absolute terms. For example, if you want to select all files that are over a month old, or to specify that a particular operation will terminate three hours after it is initiated.

Relative time expressions are specified as values that will be added or subtracted from the current time to obtain the final value. They consist of one or more terms that are composed of a plus (+) or minus (-) sign, a numeric value, and a keyword indicating the unit.

The following units can be used:

- Years
- Months
- Weeks
- Days
- Minutes
- Hours
- Seconds

You must include a space between the value and the unit and always enclose the relative time expression in quotation marks.

If no sign is included in a term, it is considered to have a sign of +. Each term in an expression is added or subtracted (according to its sign) from the current time when the expression is evaluated (during execution).

### Examples

Anything over six months old:

“-6 months”

Anything that is dated 3 weeks, 4 days and 7 hours before the current time:

“-3 weeks -4 days -7 hours”

Note that if a sign is not used specifically, the term has a sign of +, so that:

“-5 weeks 4 days”

Means 5 weeks less 4 days ago and not 5 weeks and 4 days ago. It is the same as

the expression:

“-5 weeks +4 days”

or the expression:

“-4 weeks -3 days”