

# RMFT Command Line Reference Guide

**For Windows 2000 and Higher**

Software Version 2.4.2

May 12, 2009

May 12, 2009 © 2000-2009 by RepliWeb, Inc.

The information in this manual 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 manual 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 manual 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>0</b> |
| Abbreviating Qualifiers .....                | 1        |
| <b>1. Sending Packages.....</b>              | <b>2</b> |
| The 'Send' Command.....                      | 2        |
| Basic Qualifiers for Sending a Package ..... | 3        |
| user .....                                   | 3        |
| password.....                                | 3        |
| scrambled_login_password .....               | 3        |
| server .....                                 | 3        |
| subject.....                                 | 4        |
| recipients.....                              | 4        |
| hide_recipients.....                         | 4        |
| files.....                                   | 5        |
| subdirectory_files .....                     | 6        |
| expire_after .....                           | 6        |
| Certificate-Based Login Qualifiers.....      | 8        |
| login_certificate_subject .....              | 8        |
| login_store_type.....                        | 8        |
| login_store_name .....                       | 8        |
| login_key_container .....                    | 9        |
| login_csp.....                               | 9        |
| login_provider_type.....                     | 9        |
| Qualifiers for Encrypting Packages .....     | 10       |
| encryption .....                             | 10       |
| shared_secret .....                          | 11       |
| public_certificate_subject.....              | 11       |
| public_store_type.....                       | 12       |
| public_store_name.....                       | 12       |
| public_key_container .....                   | 12       |
| public_csp.....                              | 12       |
| public_provider_type.....                    | 13       |
| secure .....                                 | 13       |

|  |           |
|--|-----------|
| cipher .....   | 13        |
| Encryption Examples.....   | 14        |
| Public Key Encryption to RMFT Server using the RMFT Server Certificate ..... | 14        |
| Shared Secret Encryption to Multiple Recipients .....                        | 14        |
| Public Key Encryption to a Single Recipient.....                             | 14        |
| Qualifiers for Signing Packages .....  | 15        |
| sign_files.....  | 15        |
| private_certificate_subject .....  | 15        |
| private_store_type .....   | 15        |
| private_store_name .....   | 15        |
| private_key_container.....   | 16        |
| private_csp.....   | 16        |
| private_provider_type .....  | 16        |
| Other 'Send' Qualifiers .....  | 17        |
| notify .....   | 17        |
| package_directory.....   | 17        |
| proxy_server .....   | 18        |
| proxy_port.....  | 18        |
| package_format .....   | 18        |
| offline .....  | 18        |
| compare_files.....   | 19        |
| abort_on_transfer_error.....   | 19        |
| trace .....  | 19        |
| relative_server_url .....  | 19        |
| allow_nonascii_filenames .....   | 20        |
| unauthenticated_download.....  | 20        |
| Submitting an Offline Package .....  | 21        |
| package_directory.....   | 21        |
| Other Submit Qualifiers .....  | 21        |
| <b>2. Searching for Packages.....</b>  | <b>22</b> |
| server .....   | 23        |
| relative_server_url .....  | 23        |
| user .....   | 23        |
| password.....  | 23        |
| secure .....   | 24        |
| proxy_server .....   | 24        |
| proxy_port.....  | 24        |
| files.....   | 24        |
| format.....  | 24        |
| xml .....  | 25        |

|                                      |           |
|--------------------------------------|-----------|
| output .....                         | 25        |
| folder .....                         | 25        |
| after .....                          | 25        |
| before.....                          | 25        |
| subject.....                         | 26        |
| from.....                            | 26        |
| to .....                             | 26        |
| message .....                        | 26        |
| read.....                            | 26        |
| encrypted .....                      | 26        |
| signed .....                         | 27        |
| expired .....                        | 27        |
| expired_since.....                   | 27        |
| expired_before.....                  | 27        |
| minimum_size.....                    | 27        |
| maximum_size.....                    | 28        |
| trace .....                          | 28        |
| Certificate Qualifiers .....         | 28        |
| <b>3. Downloading Packages .....</b> | <b>29</b> |
| server .....                         | 30        |
| relative_server_url .....            | 30        |
| user .....                           | 30        |
| password.....                        | 30        |
| secure .....                         | 31        |
| proxy_server .....                   | 31        |
| proxy_port.....                      | 31        |
| package_directory.....               | 31        |
| package_id.....                      | 31        |
| files.....                           | 32        |
| compare_files.....                   | 32        |
| after .....                          | 32        |
| before.....                          | 32        |
| subject.....                         | 32        |
| from.....                            | 33        |
| to .....                             | 33        |
| message .....                        | 33        |
| read.....                            | 33        |
| encrypted .....                      | 33        |
| signed .....                         | 33        |
| expired_since.....                   | 34        |
| expired_before.....                  | 34        |

|   |           |
|---|-----------|
| minimum_size .....  | 34        |
| maximum_size .....  | 34        |
| abort_on_transfer_error .....   | 34        |
| trace .....   | 35        |
| Certificate Qualifiers .....  | 35        |
| <b>4. Verifying and Decrypting Files.....</b>                               | <b>36</b> |
| Verifying Files .....   | 37        |
| verify .....  | 38        |
| public_certificate_subject.....   | 38        |
| public_store_type.....  | 38        |
| public_store_name.....  | 38        |
| public_key_container .....  | 39        |
| public_csp .....  | 39        |
| public_provider_type.....   | 39        |
| package_id.....   | 40        |
| signature_file.....   | 40        |
| signature_directory .....   | 41        |
| Decrypting Files.....   | 42        |
| decrypt .....   | 43        |
| key_id.....   | 43        |
| private_certificate_subject .....   | 43        |
| private_store_type .....  | 43        |
| private_store_name .....  | 44        |
| private_key_container.....  | 44        |
| private_csp.....  | 44        |
| private_provider_type .....   | 45        |
| shared_secret .....   | 45        |
| cipher .....  | 45        |
| temporary_file .....  | 45        |
| temporary_directory .....   | 46        |
| Decryption Examples.....  | 46        |
| Decrypting all Files in a Directory .....                                   | 46        |
| Decrypting all Files in a Directory that Match the File Specification ..... | 46        |
| Decrypting a Single File.....   | 46        |
| Shared Decryption and Verification Qualifiers .....                         | 47        |
| file .....  | 47        |
| file_spec.....  | 47        |
| files_directory.....  | 47        |
| target_file .....   | 48        |
| target_directory .....  | 48        |

|                                  |           |
|----------------------------------|-----------|
| <b>A. Time Expressions .....</b> | <b>49</b> |
| Absolute Time Expressions .....  | 49        |
| Relative Time Expressions .....  | 50        |

# Introduction

The RMFT native command line enables you to perform the following tasks, using a Windows MS-DOS prompt:

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

Some qualifiers are mandatory such as the `-server` qualifier, while others can be used to implement optional features such as encryption.

To be able to run RMFT command line operations, the file **bhub\_cli.exe** must reside on your machine. No other RepliWeb files are required.

Issue RMFT commands using the following **Syntax**:

```
full path>bhub_cli command qualifiers
```

## Basic Example

The following example shows the command for sending files to a single recipient.

```
C:\Program Files\RepliWeb\RMFT\b-hub\bin>bhub_cli send  
-server=12.12.12.12 -user=user1 -password=623184 -recipients=user2  
-files=c:\temp\buy3.txt
```

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

## Abbreviating Qualifiers

Qualifier names may be abbreviated, providing that the abbreviation is unique and does not conflict with another qualifier or abbreviated qualifier. The same is also true for predefined qualifier values (i.e. not user-defined). For example, the qualifier `-encryption=shared_secret` can also be written `-enc=share`

# 1. Sending Packages

This chapter provides a description of 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 uploaded to RMFT Server. A package is a folder containing the files that you selected together with various metadata required by RMFT Server for file processing and routing.

## The 'Send' Command

The command for sending packages is **send**. The **send** command must precede the qualifiers, as is illustrated in the following **Example**:

```
>bhub_cli send -server=localhost -user=user1 -password=623184  
-recipients=user2 -files=c:\temp\buy3.txt
```

---

**Note:** Commands must be issued from the same directory as the **bhub\_cli.exe** file. The default installation directory is:  
**~\RepliWeb\RMFT\b-hub\bin**

---

## Basic Qualifiers for Sending a Package

### 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: [password](#) (3), [server](#) (3), [recipients](#) (4), [files](#) (5).

### 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 or not a password is required. If you need to log in using a certificate (i.e. without a password), see [Certificate Qualifiers](#) (28). If you want to use a scrambled password, see [scrambled login password](#) (3).

See also: [server](#) (3), [recipients](#) (4), [files](#) (5), [user](#) (3).

### scrambled\_login\_password

Use the `-scrambled_login_password` qualifier instead of the `-password` qualifier to specify your scrambled RMFT password.

**To generate a scrambled password, open a DOS prompt and issue the following command:**

```
♦ >bhub_cli.exe scramble -password=your_server_password
```

### 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 files to.

See also: [password](#) (3), [recipients](#) (4), [files](#) (5), [user](#) (3).

## 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 Web Client inbox.

See also: [password](#) (3), [server](#) (3), [recipients](#) (4), [files](#) (5), [user](#) (3).

## recipients

**Syntax:** `-recipients=name`

**Syntax (multiple recipients):** `-recipients=\"name , name\"`

**Example (multiple recipients):** `-recipients=\"joe,bach:symphony,sam\"`

Use the `-recipients` qualifier to specify a list of package recipients. A recipient can be a RMFT user or a host. A host is a computer with which RMFT Server has been configured to communicate. To send files to a host, you must specify the recipient host using the following convention (see example above):

```
host_nickname:target_nickname
```

The host nickname and target nickname are aliases for the real host name and target directory path. The need to send packages to hosts should be determined after consulting with your RMFT administrator.

If you specify several recipients, the following syntax rules apply:

- Recipient names must be separated from each other by a comma with no spaces.
- The list of recipients must begin and end with a backslash followed by a quotation mark.

See also: [password](#) (3), [server](#) (3), [files](#) (5), [user](#) (3).

## 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.

## files

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

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

Use the `-files` qualifier to specify a list of files that you want to send to other RMFT users. For each file that you want to send, you must specify the full path. If you do not specify a path, the program assumes that the file is in your current working directory. If you are sending several files or if the path of a single file contains spaces, the following syntax rules apply:

- Directory names and file names must be preceded by a double-backslash.
- If you are sending several files, the paths must be separated from each other by a comma (no spaces).
- Lists of files or a single path with spaces must begin and end with a backslash followed by a quotation mark.

In the example, below `user1` sends two files to `user3`. The transfer report lists the name, size and transfer status of each file. Note that the files with the extensions `.control` and `.content` contain various processing metadata.

```
C:\Program Files\RepliWeb\RMFT\b-hub\bin>bhub_cli
send -server=localhost -user=user1 -pass=22 -recipients=user3
-file="c:\\temp\\about.doc,c:\\temp\\77mb.txt"
connecting to server
sending Hello message
logging in
creating package
total package size: 81325893 bytes
start sending files

File : c:\temp\about.doc
Size :22016
Status : Uploaded

File : c:\temp\77mb.txt
Size :81303136
Status : Uploaded

File :
C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\softlink\server\temp\user1_LGVUQH2UDM6
B5
70PZTCV90IJRI\control\SERVER_user1_LGVUQH2UDM6B570PZTCV90IJRI.control
Size :554
Status : Uploaded

File :
C:\DOCUME~1\ADMINI~1\LOCALS~1\Temp\softlink\server\temp\user1_LGVUQH2UDM6
B5
70PZTCV90IJRI\SERVER_user1_LGVUQH2UDM6B570PZTCV90IJRI.content
Size :187
Status : Uploaded
```

```
disconnecting
```

See also: [password](#) (3), [server](#) (3), [recipients](#) (4), [user](#) (3).

## subdirectory\_files

Syntax: `-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 (hosts). 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>bhub_cli send
-user=user1 -pass=33 -files=c:\upload\*.txt -server=localhost
-recipients=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>bhub_cli send
-user=user1 -pass=33 -files=c:\upload -server=localhost -recipients=user1
-subdirectory_files
```

## expire\_after

Syntax: `-expire_after=time period`

**Example:** `-expire_after=\"1 day\"`

Use the `-expire_after` qualifier to specify a time-period during which recipients will be able to download your package. At the end of the specified time-period, the package will become unavailable for download.

See [Relative Time Expressions](#) (50 ) for more examples of time-frame expressions.

## Certificate-Based Login Qualifiers

The following section describes qualifiers for authenticating your identity to RMFT Server. Note that these qualifiers should only be used after consulting with your RMFT administrator, who may require you to log in to RMFT Server using a certificate.

### login\_certificate\_subject

**Syntax:** `-login_certificate_subject=certificate_subject`

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

Use the `-login_certificate_subject` qualifier to specify a unique part of your certificate's subject.

See also: [login\\_store\\_type](#) (8), [login\\_store\\_name](#) (8) [login\\_key\\_container](#) (9), [login\\_csp](#) (9), [login\\_provider\\_type](#) (9).

### login\_store\_type

**Syntax:** `-login_store_type=user|machine|enterprise`

**Example:** `-login_store_type=user`

Use the `-login_store_type` qualifier to specify your certificate's store type. [login\\_certificate\\_subject](#) (8), [login\\_store\\_name](#) (8) [login\\_key\\_container](#) (9), [login\\_csp](#) (9), [login\\_provider\\_type](#) (9).

### login\_store\_name

**Syntax:** `-login_store_name=store_location`

**Example:** `-login_store_name=My`

Use the `-login_store_name` qualifier to specify the name of the store containing your certificate.

See also: [login\\_certificate\\_subject](#) (8), [login\\_store\\_type](#) (8), [login\\_key\\_container](#) (9), [login\\_csp](#) (9), [login\\_provider\\_type](#) (9).

## login\_key\_container

**Syntax:** `-login_key_container=key_container_name`

Use the `-login_key_container` qualifier to specify your certificate's key container.

Useful information on the role played by Cryptographic Service Provider Types and Key Containers in the cryptographic process can be found on [Microsoft's](#) corporate Web site

See also: [login\\_certificate\\_subject](#) (8), [login\\_store\\_type](#) (8), [login\\_store\\_name](#) (8), [login\\_csp](#) (9), [login\\_provider\\_type](#) (9).

## login\_csp

**Syntax:** `-login_csp=Cryptographic Service Provider`

**Example:** `-login_csp=Microsoft Base DSS Cryptographic Provider`

Use the `-login_csp` qualifier to specify your certificate's Cryptographic Service Provider.

See also: [login\\_certificate\\_subject](#) (8), [login\\_store\\_type](#) (8), [login\\_store\\_name](#) (8), [login\\_key\\_container](#) (9), [login\\_provider\\_type](#) (9).

## login\_provider\_type

**Syntax:** `-login_provider_type=Cryptographic Service Provider Type`

Use the `-login_provider_type` qualifier to specify your certificate's Cryptographic Service Provider Type. The *Cryptographic Service Provider Type* should be represented by a numeric value.

Useful information on the role played by Cryptographic Service Provider Types and Key Containers in the cryptographic process can be found on [Microsoft's](#) corporate Web site

See also: [login\\_certificate\\_subject](#) (8), [login\\_store\\_type](#) (8), [login\\_store\\_name](#) (8), [login\\_key\\_container](#) (9), [login\\_csp](#) (9).

## Qualifiers for Encrypting Packages

You can encrypt packages by adding encryption qualifiers to the basic `send` command line described in [Basic Qualifiers for Sending a Package](#) (3). The supported encryption methods can be implemented using the qualifiers described below.

### encryption

The `-encryption` qualifier accepts three possible values, each enabling a different method of encryption.

**Syntax:** `-encryption=shared_secret|pki_single_recipient|encrypt_to_server`

**Example:** `-encryption=shared_secret`

See also: [Encryption Examples](#) (14).

### 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:**

- ◆ `-encryption=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](#) (11) and [cipher](#) (13).

### 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 stores.

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

- ◆ `-encryption=pki_single_recipient`

See also: [public\\_certificate\\_subject](#) (11), [public\\_store\\_type](#) (12), [public\\_store\\_name](#) (12), [public\\_key\\_container](#) (12), [public\\_csp](#) (12), [public\\_provider\\_type](#) (13).

### **encrypt\_to\_server**

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

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

- ◆ `-encrypt=encrypt_to_server`

### **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.

See also: [cipher](#) (13) [encryption](#) (10) and [secure](#) (13).

### **public\_certificate\_subject**

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

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

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

See also: [public\\_store\\_type](#) (12), [public\\_store\\_name](#) (12), [public\\_key\\_container](#) (12), [public\\_csp](#) (12), [public\\_provider\\_type](#) (13).

## public\_store\_type

**Syntax:** `-public_store_type=user|machine|enterprise`

**Example:** `-public_store_type=user`

Use the `-public_store_type` qualifier to specify the store type of the recipient's public key certificate.

See also: [public\\_certificate\\_subject](#) (11), [public\\_store\\_name](#) (12), [public\\_key\\_container](#) (12), [public\\_csp](#) (12), [public\\_provider\\_type](#) (13).

## public\_store\_name

**Syntax:** `-public_store_name=store_location`

**Example:** `-public_store_name=My`

Use the `-public_store_name` qualifier to specify the store name of the recipient's public key certificate.

See also: [public\\_certificate\\_subject](#) (11), [public\\_store\\_type](#) (12), [public\\_key\\_container](#) (12), [public\\_csp](#) (12), [public\\_provider\\_type](#) (13).

## public\_key\_container

**Syntax:** `-public_key_container=key_container_name`

Use the `-public_key_container` qualifier to specify the key container of the recipient's public certificate.

Useful information on the role played by Cryptographic Service Provider Types and Key Containers in the cryptographic process can be found on [Microsoft's](#) corporate Web site.

See also: [public\\_certificate\\_subject](#) (11), [public\\_store\\_type](#) (12), [public\\_store\\_name](#) (12), [public\\_csp](#) (12), [public\\_provider\\_type](#) (13).

## public\_csp

**Syntax:** `-public_csp=Cryptographic Service Provider`

**Example:** `-public_csp=Microsoft Base DSS Cryptographic Provider`

Use the `-public_csp` qualifier to specify the Cryptographic Service Provider of the recipient's public certificate.

See also: [public\\_certificate\\_subject](#) (11), [public\\_store\\_type](#) (12), [public\\_store\\_name](#) (12), [public\\_key\\_container](#) (12), [public\\_provider\\_type](#) (13).

## public\_provider\_type

**Syntax:** `-public_provider_type=Cryptographic Service Provider Type`

Use the `-public_provider_type` qualifier to specify the Cryptographic Service Provider Type of the recipient's public certificate. The *Cryptographic Service Provider Type* should be represented by a numeric value.

Useful information on the role played by Cryptographic Service Provider Types and Key Containers in the cryptographic process can be found on [Microsoft's](#) corporate Web site.

See also: [public\\_certificate\\_subject](#) (11), [public\\_store\\_type](#) (12), [public\\_store\\_name](#) (12), [public\\_key\\_container](#) (12), [public\\_csp](#) (12).

## 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 RMFT Server's IIS server. Contact your RMFT administrator to verify whether or not a server-side certificate exists. In addition, you must import the CA (Certificate Authority) certificate to your CA certificate store.

## 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: [cipher](#) (13) [encryption](#) (10) and [secure](#) (13).

## Encryption Examples

### Public Key Encryption to RMFT Server using the RMFT Server Certificate

```
C:\Program Files\RepliWeb\RMFT\b-hub\bin>bhub_cli send -user=mor6 -pass=6
-server=10.0.20.236 -recipients=mike -files=C:\Mor\decrypt\4.txt
-encrypt=encrypt_to_server
```

### Shared Secret Encryption to Multiple Recipients

```
C:\Program Files\RepliWeb\RMFT\b-hub\bin>bhub_cli send -user=mor6 -pass=6
-server=10.0.20.236 -recipients="joe,jane,tim\"
-files=C:\Mor\decrypt\4.txt -encrypt=shared_secret -shared_secret=123
-public_certificate_subject=mor_s.rwint.com -public_store_type=machine
-public_store_name=my -cipher=rc4
```

### Public Key Encryption to a Single Recipient

```
C:\Program Files\RepliWeb\RMFT\b-hub\bin>bhub_cli send -user=mor6 -pass=6
-server=10.0.20.236 -recipients=mike -files=C:\Mor\decrypt\4.txt
-encrypt=pki_single_recipient -public_certificate_subject=mor_s.rwint.com
-public_store_type=user -public_store_name=my -cipher=rc4
```

## Qualifiers for Signing Packages

Use your private key certificate 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. The `-sign_files` qualifier should be used together with the qualifiers described below.

See also: [private\\_certificate\\_subject](#) (15), [private\\_store\\_type](#) (15), [private\\_store\\_name](#) (15), [private\\_key\\_container](#) (16), [private\\_csp](#) (16), [private\\_provider\\_type](#) (16).

### private\_certificate\_subject

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

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

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

See also: [private\\_store\\_type](#) (15), [private\\_store\\_name](#) (15), [private\\_key\\_container](#) (16), [private\\_csp](#) (16), [private\\_provider\\_type](#) (16).

### private\_store\_type

**Syntax:** `-private_store_type=user|machine|enterprise`

**Example:** `-private_store_type=user`

Use the `-private_store_type` qualifier to specify the store type of your private key certificate.

See also: [private\\_certificate\\_subject](#) (15), [private\\_store\\_name](#) (15), [private\\_key\\_container](#) (16), [private\\_csp](#) (16), [private\\_provider\\_type](#) (16).

### private\_store\_name

**Syntax:** `-private_store_name=store_location`

**Example:** `-private_store_name=My`

Use the `-private_store_name` qualifier to specify the store name of your private key certificate.

## private\_key\_container

**Syntax:** `-private_key_container=key_container_name`

Use the `-private_key_container` qualifier to specify the key container of your private key certificate.

Useful information on the role played by Cryptographic Service Provider Types and Key Containers in the cryptographic process can be found on [Microsoft's](#) corporate Web site.

See also: [private\\_certificate\\_subject](#) (15), [private\\_store\\_type](#) (15), [private\\_store\\_name](#) (15), [private\\_csp](#) (16), [private\\_provider\\_type](#) (16).

## private\_csp

**Syntax:** `-private_csp=Cryptographic Service Provider`

**Example:** `-private_csp=Microsoft Base DSS Cryptographic Provider`

Use the `-private_csp` qualifier to specify the Cryptographic Service Provider of your private key certificate.

See also: [private\\_certificate\\_subject](#) (15), [private\\_store\\_type](#) (15), [private\\_store\\_name](#) (15), [private\\_key\\_container](#) (16), [private\\_provider\\_type](#) (16).

## private\_provider\_type

**Syntax:** `-private_provider_type=Cryptographic Service Provider Type`

Use the `-private_provider_type` qualifier to specify the Cryptographic Service Provider Type of your private key certificate. The *Cryptographic Service Provider Type* should be represented by a numeric value.

Useful information on the role played by Cryptographic Service Provider Types and Key Containers in the cryptographic process can be found on [Microsoft's](#) corporate Web site.

See also: [private\\_certificate\\_subject](#) (15), [private\\_store\\_type](#) (15), [private\\_store\\_name](#) (15), [private\\_key\\_container](#) (16), [private\\_csp](#) (16).

## Other ‘Send’ Qualifiers

### 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> | Notifies you when the package has been distributed to all of the recipients' inboxes.   |
| <code>delivered_to_hosts</code>           | Notifies you when the package has been delivered to all of the hosts.   |
| <code>opened_or_downloaded</code>         | Notifies you 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`

### package\_directory

**Syntax:** `-package_directory=<directory full path>`

Use the `-package_directory` qualifier to save the package metadata to a temporary directory before the package is sent. The package metadata consists of

routing information included in the **SERVER\_<package ID>.control** and **SERVER\_<package ID>.content** files. If you omit the `-package_directory` qualifier, the package metadata will be temporarily saved to your current working directory. To permanently save the entire package (and not just the package metadata), use this qualifier together with the `-offline` qualifier. You can send a saved package whenever you need, using the [submit](#) command.

See also: [Submitting an Offline Package](#) and [offline](#) (18).

## 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\_format

**Syntax:** `-package_format=[directory|zip]`

**Default:** `-package_format=directory`

**Example:** `-package_format=zip`

Use this qualifier to determine the package format. If you omit this qualifier, the package will be sent as a directory.

### To zip the package, specify:

- ◆ `-package_format=zip`

## 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_directoryqualifier`. You can send the package later using the [submit](#) command with the `-package_directoryqualifier`.

## Example

```
bhub_cli.exe send -server=server.server -user=charliebrowne  
-files=\"c:\\aaa.xml,c:\\temp\\a.a\" -recipients=yoel@company.com  
-offline -package_directory=c:\\server\\temp\\4
```

See also: [Submitting an Offline Package](#) (21).

## 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.

## 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](#).

## 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.

## 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. Offline packages are packages that have been saved on your computer using the `-offline` qualifier.

The *submit* command should be followed by login qualifiers (required for logging into RMFT Server) and the `-package_directory` qualifier. Other qualifiers such as the `-recipients` and `-files` qualifier are not required, since the package already contains this information.

---

**Note:** The qualifiers that you require 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 your certificate subject. If you are unsure of how to log into RMFT Server, contact your RMFT administrator.

---

### Example

```
>bhub_cli.exe submit -server=212.29.222.82 -user=charliebrown  
-pass=1ft2g -package_dir=c:\server\temp\4\mike_8S6IPVK0Q8YO394U0N1J51L1L1
```

See also: [offline](#) (18).

### package\_directory

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

**Example:**

```
-package_dir=c:\server\temp\4\mike_8S6IPVK0Q8YO394U0N1J51L1L1
```

Use the `-package_directory` qualifier to specify the full path of the package to be sent (i.e. including the package name).

### Other Submit Qualifiers

Other qualifiers that can be used with the submit command include: [secure](#) (13), [login\\_certificate\\_subject](#) (8), [login\\_store\\_type](#) (8), [login\\_store\\_name](#) (8), [login\\_key\\_container](#) (9), [login\\_csp](#) (9), [login\\_provider\\_type](#) (9).

## 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>bhub_cli 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>bhub_cli 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/IP address of the RMFT Server to search.

See also: [user](#) (23), [password](#) (23).

## 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](#) (23), [password](#) (23).

## 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 or not a password is required. If you log in using a certificate (i.e. without a password), see [Certificate Qualifiers](#) (28).

See also: [server](#) (23), [user](#) (23).

## 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>`

Use this qualifier to specify 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).

## 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=id|list|details|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

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. The date and time format must conform to the syntax described in [A. Time Expressions](#) (49).

## before

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

Use the `-before` qualifier to only search for packages with dates earlier than the specified date. The date and time format must conform to the syntax described in [A. Time Expressions](#) (49).

## 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 `-encrypted` 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](#) (49).

## 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](#) (49).

## 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.

## 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.

## 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 Qualifiers

See [Certificate-Based Login Qualifiers](#) (8) for qualifiers that you can use to authenticate your identity to RMFT Server. Note that these qualifiers should only be used after consulting with your RMFT administrator, who may require you to log in using a certificate

# 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 command `download`.

---

**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>bhub_cli 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>bhub_cli download  
-server=<123.123.123.12> -user=user1 -password=g245t60  
-package_directory=c:\target
```

Mandatory download qualifiers are: [server](#) (30), [user](#) (30), [password](#) (30) and [package\\_directory](#) (31).

## 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](#) (30), [password](#) (30), [package\\_directory](#) (31)

## 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](#) (30), [password](#) (30), [package\\_directory](#) (31).

## 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 or not a password is required. If you log in using a certificate (i.e. without a password), see [Certificate Qualifiers](#) (28).

See also: [server](#) (30), [user](#) (30), [package\\_directory](#) (31).

## 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>`

Use this qualifier to specify 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).

## 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](#) (30), [user](#) (30), [password](#) (30), [package\\_directory](#) (31).

## 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.

## files

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

**Example:** `-files=*.txt`

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. 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.

## after

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

Use the `-after` qualifier to only download packages with dates later than the specified date. The date and time format must conform to the syntax described in [A. Time Expressions](#) (49).

## before

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

Use the `-before` qualifier to only download packages with dates earlier than the specified date. The date and time format must conform to the syntax described in [A. Time Expressions](#) (49).

## 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](#) (49).

## 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](#) (49).

## mimimum\_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.

## 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.

## abort\_on\_transfer\_error

The `-abort_on_transfer_error` qualifier aborts the download 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=<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 Qualifiers

See [Certificate-Based Login Qualifiers](#) (8) for qualifiers that you can use to authenticate your identity to RMFT Server. Note that these qualifiers should only be used after consulting with your RMFT administrator, who may require you to log in using a certificate.

# 4. 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>bhub_cli process -verify
-public_certificate_subject=charlie@company.com -public_store_type=user
-public_store_name=my -decrypt -private_certificate_subject=john@corp.com
-private_store_type=user -private_store_name=my -
files_directory=c:\received
```

This chapter is divided into the following sections:

- [Verifying Files](#) (37)
- [Decrypting Files](#) (42)
- [Shared Decryption and Verification Qualifiers](#) (47)

## 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](#) (38)
- [public\\_certificate\\_subject](#) (38)
- [public\\_store\\_type](#) (38)
- [public\\_store\\_name](#) (38)
- [public\\_key\\_container](#) (39)
- [public\\_csp](#) (39)
- [public\\_provider\\_type](#) (39)
- [package\\_id](#) (40)
- [signature\\_file](#) (40)
- [signature\\_directory](#) (41)

Signature verification qualifiers must follow the `process` command and be used together with the file location qualifiers described in [Shared Decryption and Verification Qualifiers](#) (47).

### Example:

```
C:\Program Files\RepliWeb\RMFT\b-hub\bin>bhub_cli process -verify
-public_certificate_subject=charlie@company.com -public_store_type=user
-public_store_name=my -file_spec=c:\temp\*.doc
```

## 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\\_certificate\\_subject](#) (38), [public\\_store\\_type](#) (38), [public\\_key\\_container](#) (39), [public\\_store\\_name](#) (38), `public_csp` (39), [public\\_provider\\_type](#) (39).

## 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 contained in the signer's public key certificate. You can also use the `-public_store_type` and `-public_store_name` qualifiers to override the default store type (**User**) and store name (**My**).

See also: [public\\_store\\_type](#) (38), [public\\_key\\_container](#) (39), [public\\_store\\_name](#) (38), `public_csp` (39), [public\\_provider\\_type](#) (39).

## public\_store\_type

**Syntax:** `-public_store_type=user|machine|enterprise`

**Default:** `-public_store_type=user`

Use the `-public_store_type` qualifier to override the default store type (`user`) of the signer's public key certificate.

| Permitted Values        | Description            |
|-------------------------|------------------------|
| <code>user</code>       | Current user store     |
| <code>machine</code>    | Local machine store    |
| <code>enterprise</code> | Local enterprise store |

See also: [public\\_certificate\\_subject](#) (38), [public\\_key\\_container](#) (39), [public\\_store\\_name](#) (38), [public\\_csp](#) (39), [public\\_provider\\_type](#) (39).

## public\_store\_name

**Syntax:** `-public_store_name=<store_location>`

**Default:** `-public_store_name=My`

Use the `-public_store_name` qualifier to override the default name (**My**) of the store containing the signer's public key certificate.

See also: [public\\_certificate\\_subject](#) (38), [public\\_store\\_type](#) (38), [public\\_key\\_container](#) (39), [public\\_csp](#) (39), [public\\_provider\\_type](#) (39).

## public\_key\_container

**Syntax:** `-public_key_container=<key_container_name>`

Use the `-public_key_container` qualifier to specify the signer's public key certificate container.

See also: [public\\_certificate\\_subject](#) (38), [public\\_store\\_type](#) (38), [public\\_store\\_name](#) (38), [public\\_csp](#) (39), [public\\_provider\\_type](#) (39).

## public\_csp

**Syntax:** `-public_csp=<Cryptographic Service Provider>`

**Example:** `-public_csp=Microsoft Base DSS Cryptographic Provider`

Use the `-public_csp` qualifier to specify the Cryptographic Service Provider of the signer's public key certificate.

See also: [public\\_certificate\\_subject](#) (38), [public\\_store\\_type](#) (38), [public\\_key\\_container](#) (39), [public\\_store\\_name](#) (38), [public\\_provider\\_type](#) (39).

## public\_provider\_type

**Syntax:** `-public_provider_type=<Cryptographic Service Provider Type>`

Use the `-public_provider_type` qualifier to specify a numeric value representing the Cryptographic Service Provider Type of the signer's public key certificate.

See also: [public\\_certificate\\_subject](#) (38), [public\\_store\\_type](#) (38), [public\\_key\\_container](#) (39), [public\\_store\\_name](#) (38), [public\\_csp](#) (39).

## 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_certificate_subject` qualifier. You must also specify the location of the signed files using the `-file_spec`, `-files_directory` or `-files` qualifiers described in [Shared Decryption and Verification Qualifiers](#) (47).

---

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

---

## signature\_file

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

**Example:**

```
-signature_file="c:\\new tax\\accounts.doc.sig,d:\\profit\\sales.doc.sig"
```

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_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). You can specify several signature files, as long as they are all from the same package and their paths are separated from each other by a comma (no spaces). If one or more of the paths contains a space, all files and directory names must be preceded by a double backslash and the entire list enclosed with quotation marks, escaped with backslashes (see example above). The `-signature_file` qualifier should be used with the `-files`, and `-public_certificate_subject` qualifiers. You can

also use the `-public_store_type` and `-public_store_name` qualifiers to override the default store type (**User**) and store name (**My**).

---

**IMPORTANT:** When specifying several signature files, the files must be specified in the same order as their equivalent signed files (which are specified using the `-files` qualifier). For

**Example:**

```
-file=c:\file1.doc,c:\file2.doc,c:\file3.doc  
-signature_files=c:\file1.doc.sig,c:\file2.doc.sig,  
c:\file3.doc.sig
```

---

## signature\_directory

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

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_file` 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_dir` qualifier should be used with the `-files_dir`, and `-public_certificate_subject` qualifiers. You can also use the `-public_store_type` and `-public_store_name` qualifiers to override the default store type (**User**) and store name (**My**).

## 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](#) (43)
- [key\\_id](#) (43)
- [private\\_certificate\\_subject](#) (43)
- [private\\_store\\_type](#) (43)
- [private\\_store\\_name](#) (44)
- [private\\_key\\_container](#) (44)
- [private\\_csp](#) (44)
- [private\\_provider\\_type](#) (45)
- [shared\\_secret](#) (45)
- [cipher](#) (45)
- [temporary\\_file](#) (45)
- [temporary\\_directory](#) (46)
- [target\\_file](#) (48)
- [target\\_directory](#) (48)

Decryption qualifiers must follow the `process` command and be used together with the file location qualifiers described in [Shared Decryption and Verification Qualifiers](#) (47).

### Example:

```
C:\Program Files\RepliWeb\RMFT\b-hub\bin>bhub_cli process -decrypt
-private_certificate_subject=charlie@company.com -private_store_type=user
-private_store_name=my -files_directory=c:\temp
-target_directory=c:\target
```

## 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: [private\\_certificate\\_subject](#) (43), [private\\_store\\_type](#) (43), [private\\_store\\_name](#) (44), [private\\_key\\_container](#) (44), [private\\_csp](#) (44), [private\\_provider\\_type](#) (45), [shared\\_secret](#) (45) [cipher](#) (45).

## 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: [private\\_store\\_type](#) (43), [private\\_certificate\\_subject](#) (43), [private\\_store\\_name](#) (44), [private\\_key\\_container](#) (44), [private\\_csp](#) (44), [private\\_provider\\_type](#) (45).

## private\_certificate\_subject

The `-private_certificate_subject` qualifier is mandatory when decrypting files that were encrypted using your public key certificate.

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

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

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

See also: [private\\_store\\_type](#) (43), [private\\_store\\_name](#) (44), [private\\_key\\_container](#) (44), [private\\_csp](#) (44), [private\\_provider\\_type](#) (45).

## private\_store\_type

Use the `-private_store_type` qualifier to override the default certificate store type when decrypting files that were encrypted using your public key certificate.

**Syntax:** `-private_store_type=user|machine|enterprise`

**Default:** `-private_store_type=user`

| Permitted Values | Description            |
|------------------|------------------------|
| user             | Current user store     |
| machine          | Local machine store    |
| enterprise       | Local enterprise store |

See also: [private\\_certificate\\_subject](#) (43), [private\\_store\\_name](#) (44), [private\\_key\\_container](#) (44), [private\\_csp](#) (44), [private\\_provider\\_type](#) (45), [shared\\_secret](#) (45) [cipher](#) (45).

### private\_store\_name

Use the `-private_store_name` qualifier to override the default store name when decrypting files that were encrypted using your public key certificate.

**Syntax:** `-private_store_name=<store_name>`

**Default:** `-private_store_name=My`

See also: [private\\_certificate\\_subject](#) (43), [private\\_store\\_type](#) (43), [private\\_store\\_name](#) (44), [private\\_key\\_container](#) (44), [private\\_csp](#) (44), [private\\_provider\\_type](#) (45), [shared\\_secret](#) (45) [cipher](#) (45).

### private\_key\_container

**Syntax:** `-private_key_container=key_container_name`

Use the `-private_key_container` qualifier to specify the name of your private key container.

See also: [private\\_certificate\\_subject](#) (43), [private\\_store\\_type](#) (43), [private\\_store\\_name](#) (44), [private\\_key\\_container](#) (44), [private\\_csp](#) (44), [private\\_provider\\_type](#) (45), [shared\\_secret](#) (45) [cipher](#) (45).

### private\_csp

**Syntax:** `-private_csp=Cryptographic Service Provider`

**Example:** `-private_csp=Microsoft Base DSS Cryptographic Provider`

Use the `-private_csp` qualifier to specify the Cryptographic Service Provider of your private key certificate.

See also: [private\\_certificate\\_subject](#) (43), [private\\_store\\_type](#) (43), [private\\_store\\_name](#) (44), [private\\_key\\_container](#) (44), [private\\_provider\\_type](#) (45), [shared\\_secret](#) (45) [cipher](#) (45).

## private\_provider\_type

**Syntax:** `-private_provider_type=Cryptographic Service Provider Type`

Use the `-private_provider_type` qualifier to specify the Cryptographic Service Provider Type of your private key certificate. The *Cryptographic Service Provider Type* should be represented by a numeric value.

See also: [private\\_certificate\\_subject](#) (43), [private\\_store\\_type](#) (43), [private\\_store\\_name](#) (44), [private\\_key\\_container](#) (44), [private\\_csp](#) (44), [shared\\_secret](#) (45) [cipher](#) (45).

## 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](#) (45)

## 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](#) (45)

## temporary\_file

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

**Example:** `-files=c:\temp\old.doc -temporary_file=c:\temp\temp.doc`

Use the `-temporary_file` qualifier to provide a temporary name for a decrypted file before it is written to the final target directory. This is useful if an application is waiting to pick up a file with a specific file name from the final target directory. After being written to the final target directory, the file will either be renamed to its original name or to the name specified with the `-target_file` qualifier.

## temporary\_directory

**Syntax:** `-temporary_directory=<directory name>`

**Example:** `-temporary_directory=c:\temp`

Use the `-temporary_directory` qualifier to specify the name of a temporary directory in which to decrypt the files. After decryption the files will be moved to the final target directory (specified with the `-target_directory` qualifier). This is useful if an application is waiting to pick up files from the final target directory.

## Decryption Examples

### Decrypting all Files in a Directory

```
C:\Program Files\RepliWeb\RMFT\b-hub\bin>bhub_cli process -decrypt
-private_certificate_subject=mor_s.rwint.com -private_store_type=machine
-private_store_name=my -files_directory=C:\Mor\encrypt
-target_directory=C:\temp\decrypt
```

### Decrypting all Files in a Directory that Match the File Specification

```
C:\Program Files\RepliWeb\RMFT\b-hub\bin>bhub_cli process -decrypt
-private_certificate_subject=mor_s.rwint.com -private_store_type=machine
-private_store_name=my -file_spec=C:\Mor\encrypt\*.txt
-target_directory=C:\temp\decrypt
```

### Decrypting a Single File

```
C:\Program Files\RepliWeb\RMFT\b-hub\bin>bhub_cli process -decrypt
-private_certificate_subject=mor_s.rwint.com -private_store_type=machine
-private_store_name=my -file=C:\Mor\encrypt\11.txt
-target_file=C:\temp\decrypt\111.txt
```

## 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](#) (37) and [Decrypting Files](#) (42).

### file

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

**Example:** `-file="c:\\new tax\\accounts.pdf,d:\\profit\\sales.doc"`

Use the `-file` qualifier to specify the full path of digitally signed/encrypted files. You can specify several file names, as long as they are *all from the same package* and their paths are separated from each other by a comma (no spaces). If one or more of the paths contains a space, all files and directory names must be preceded by a double backslash and the entire list enclosed with quotation marks, escaped with backslashes (see example above).

### 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.

### files\_directory

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

Use the `-files_directory` qualifier to specify the name of a directory that only contains digitally signed/encrypted files. If the directory contains files that have not been digitally signed/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:** `-files_directory=c:\temp -target_directory=c:\target`

Use the mandatory `-target_directory` qualifier to specify a target directory for the verified/decrypted files. This qualifier must be used with the `-files_directory` qualifier.

# 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”