MICRO FOCUS

StarTeam 16.2

Command-line Tools Help

Micro Focus
The Lawn
22-30 Old Bath Road
Newbury, Berkshire RG14 1QN
UK
http://www.microfocus.com

2017-10-18

Contents

Introduction

The StarTeam Command Line Tools provides access into the StarTeam Server via the command line.

The StarTeam Command Line Tools are installed as part of the StarTeam SDK. The SDK is automatically installed with most StarTeam clients and the StarTeam Server.

To access the StarTeam Command Line Tools, open a command prompt and navigate to `C:\program files\Micro Focus\StarTeam SDK <version>\lib`.

Type `stcmd help` to get started.

Additional Resources

Additional information may also be found on the product community forum wiki pages:

http://community.microfocus.com/borland/managetrack/starteam/w/wiki/620.starteam-sdk-topics-releases.aspx

Check-out Trace Utility Command-line Operations

Below are descriptions of the command-line options for the **Check-out Trace** utility.

In general, you can run the utility from the command line with default options as follows: `CheckoutTraceDump.exe -go`. Valid options for **Check-out Trace** are described below.

-go	Specify this flag to run with default settings.
–path:<path>	Folder of the binary check-out trace (`.cotrc`) files. Defaults to the current folder.
–outpath:<path>	Folder for the output (`.csv`) files. Defaults to the same folder as the binary trace (`.cotrc`) files.
-file:<filespec>	Binary check-out trace files to be used as input. Supports standard file system wildcard values (`*,?`). Defaults to `"*.cotrc"` (all check-out trace files in the folder). You cannot use more than one path with the parameter, and you cannot specify this parameter more than once per command.
-ext:<extension>	The file extension used for check-out trace files. The extension is appended to the binary check-out trace (`.cotrc`) filename to create the output dump filename. Defaults to `.csv`.
-start:<start time>	Earliest date-time of interest. Only checkouts that occurred after this time will be output. By default, the utility does not filter by time.
-end:<end time>	Specifies the most recent date-time of interest. Only checkouts that occurred before this time will be output. By default, the utility does not filter by time.
–project:<project name>	Name of the project for which check-out information is to be filtered. Only checkouts from this project will be output. By default, the utility does not filter by project. All projects are included in the output. If both `-project` and `-projectid` are specified, `-projectid` takes precedence.
-projectid:<project ID>	ID of the project for which check-out information is to be filtered. Only checkouts from this project will be output. By default, the utility does not filter by project ID. All projects are included in the output. This option takes precedence over `-project` if both options are specified.
-separator:<separator>	String used to separate values in the output file. By default, the utility uses `","`.
–overwrite	Specify this flag to overwrite existing check-out trace files. If not specified, check-out trace binary files will be skipped if a trace dump file with the target name already exists.

Client Command-line Operations

stcmd.exe, stcmdEX.jar and the CommandProcessor Object

Use the `stcmd` executable from a command line, such as Microsoft Windows **Command Prompt**. `stcmd.exe` runs an in-process java program called the client. This client starts up or connects to an already running out-of-process java program called the server. The client and the server then communicate over TCP/IP. The level of indirection is necessary to support stateful commands (commands where a connection to StarTeam Server is created and retained across the scope of all subsequently issued `stcmd` commands). Using `stcmd` from a command line requires adding `stcmd` in front of each command on the command line. This will enable the `stcmd` executable to drive the **CommandProcessor** engine.

`Stcmd.exe` is a 32-bit executable that launches `stjava.exe` out-of-process. On 32 bit machines, it launches the 32-bit version. On 64-bit machines, it launches the 64-bit version.

The SDK runs in the (32/64 bit version) JRE spawned by `stjava.exe`, independent of `stcmd.exe`. `Stcmd.exe` is a pure java solution, the choice of class path determines the choice of JRE.

stcmd Path Specification

`stcmd` path specifications must use Java conventions (not Microsoft Windows). For example, the following will throw an `IndexOutOfBounds` exception:

```
stcmd co -rp "c:\temp" -p "Administrator:Administrator@localhost:49201/
StarDraw/StarDraw" *
```

The following will work correctly on all platforms that support the java virtual machine (Microsoft Windows, Unix, and Mac):

```
stcmd co -rp "c:/temp" -p "Administrator:Administrator@localhost:49201/
StarDraw/StarDraw" *
```

stcmdEx

When running the command line as a stateless batch process (using the `-p` or the `-s` parameters), an alternative approach is to use `stcmdEx`.

`stcmdEx.jar` runs a java program that takes the command and input parameters, and then loads and invokes the StarTeam SDK in process. There is no additional overhead of a separate out-of-process server, but as a consequence, there is no possibility of retaining state across invocations.

Important: `stcmdEx` will not work with stateful commands. That is, commands that follow the `connect`, `set`, ..., `disconnect` pattern.

`stcmdEx` will only work with stateless commands, or commands which rely upon `-p` to specify connectivity, credentials, project and view.

`stcmdEx` does not support the following commands: `status`, `statusAll`, `shutdown`, and `shutdownAll`. These are not commands, rather, they are tracking methods for the local client/server `stcmd` processes.

Additionally, `stcmdEx` does not support help queries. For example: `-?`.

`stcmdEx` is not meant to be used in an interactive environment. It is targeted at (concurrent) batch processes and services set up to run with no human intervention.

Since `stcmdEx` launches and runs the SDK in process, multiple parallel invocations of the script will each run in their own process spaces (JRE runtimes), thereby supporting batch parallelism. `stcmdEx` jobs can be run in parallel on the same physical workstation.

`stcmdEx` can be directly integrated into existing Microsoft Windows batch files (or Unix shell scripts) using the following syntax:

```
java.exe -jar stcmdEx.jar command-line-command-with-parameters
```

```
"C:\Program Files\Micro Focus\Java\Oracle1.8.0_91\bin\java.exe" -jar "C:\
Program Files\Micro Focus\StarTeam SDK <version>\lib\stcmdEx.jar" co /p
\"Administrator:Administrator@localhost:49201/Project Name/View Name\" /is
/rp \"c:\temp\" /filter \"GIMOU\" /o /vb >c:\temp\output.txt
```

Note: When passing command arguments within Microsoft Windows command prompts or bat files, `"` needs to be escaped with a `\` so that the `"` itself gets streamed as part of the argument.

On Microsoft Windows platforms, `stcmdEx` can be incorporated into its own `.bat` file if necessary. Add the next two lines to a .bat file (`stcmdEx.bat`):

```
@echo ON
"C:\Program Files\Micro Focus\Java\Oracle1.8.0_91\bin\java.exe" -jar "C:
\Program Files\Micro Focus\StarTeam SDK <version>\lib\stcmdEx.jar" %*
```

This approach uses the standard MS-DOS command line execution and parameter passing technique. Then call this `.bat` file from a containing batch script:

```
CALL stcmdEx.bat co -p "Administrator:Administrator@localhost:49201/Project
Name/View Name" -is -rp "c:\temp" -filter "GIMOU" -o -vb >c:\temp\output.txt
```

Note: `stcmd` automatically delegates all stateless commands (for example, `-p` or `-s`) to `stcmdEx`. As a result, both `stcmd` and `stcmdEx` support platform return codes of `0` on success, `1` on failure for stateless commands alone. `stcmd` does not support return codes for stateful commands (where it is expected to be used interactively).

Using CommandProcessor Natively

Application developers who need to incorporate command line functionality natively into Ant, Hudson, or other scripts can directly load the SDK .jar and instantiate the **CommandProcessor** object without any of the `stcmd` shell overhead.

Each **CommandProcessor** object represents an instance of a different StarTeam Server connection. The actual object signature and usage pattern can be invoked from JavaScript, Jython, etc.

SDK Object Example

Below is an example of using the SDK object.

```
CommandProcessor cp = new CommandProcessor();
cp.execute(“connect localhost:49201@Administrator:Administrator”);
cp.execute(“set project = StarDraw view = StarDraw”);
cp.execute(“select name,status,dotnotation from changerequest into
queryoutput.txt where folder = \"Sales Material\" recurse order by name”);
cp.execute(“disconnect”);
```

The full **CommandProcessor** interface documentation is available as part of the SDK javadocs.

Special Characters

`*` Matches any string including an empty string. For example, `"x*z"` will match `"xyz"` and `"xz"`. `?` Matches any single character. For example, `"a?c"` will match `"abc"` but not `"ac"`.

[...] Matches any one of the characters enclosed by the left and right brackets.

A pair of characters separated by a hyphen (`-`) specifies a range of characters to be matched. If the first character following a left bracket (`[`) is an exclamation point (`!`) or a caret (^), the rest of the characters are not matched. Any character not enclosed in the brackets is matched. A hyphen (`-`) or right bracket (`]`) may be matched by including it as the first or last character in a bracketed set. For example, `"x[a - d]y"` matches `"xby"` `-`) or right bracket (`]`) may be matched by including it as the first or last character in a bracketed set. For example, `"x[a - d]y"` matches but not `"xey"` while `"x[!a - d]y"` matches `"xey"` but not `"xby"`. If you want to use an `*` or `?` or `[` in a pattern, you must precede it with the escape character (that is, a backslash `\`).

If you use `*` rather than `"*"` to indicate all files, a UNIX shell expands it into a series of items and passes this series as a group of options to the `stcmd` command. This can cause problems (for example, when you are checking out missing files) so it is best to use `"*"` and avoid unwanted complications. If you use a set of file patterns, each pattern should be enclosed in its own set of quotation marks. For example, you can use `"*.bat" "*.c"`, but you cannot use `"*.bat *.c"`.

These special characters also apply to the `files...` option available in some commands.

If `-p` is used without quotes (""), then use the following special characters. However, note that this is not a recommended practice.

If any of the variables used with this option contain characters that are used as delimiters, use the percent sign (%) followed by the hex code for each of those characters. For example, if “@” appears as a character in a password, you must replace it with `"%40"`. Replace the following:

For ":"	use `"%3a"`
For "/"	use `"%2f"`
For "@"	use `"%40"`
For "%"	use `"%25”`

In UNIX and other operating systems, some special characters must be preceded by a backslash `"\"` or another escape character. In the `-p` option, you can replace such characters with hex codes. For example, `"%3c"` could be used in UNIX instead of `"\<"`. Replace the following:

For a space	use `"%20"`
For "<"	use `"%3c"`
For ">"	use `"%3e”`

Add Enum: add-enum

Use the `add-enum` command to add an enumerated value to an existing enumerated property of a type on the server.

Important: Requires administrative permissions.

Syntax

```
stcmd{Ex} add-enum –s username[:password]@host:port [-epwdfile "filePath"]
[-cmp] [-encrypt encryptionType] –type typeName –property propertyName
–name enumInternalName [–displayName displayName] [-sortOrder position]
[-parentEnum parentEnumName] [-predecessorEnum predecessorEnumName]
```

Parameters

-s

Identifies the StarTeam Server. The full syntax is: `-s "userName:password@host:portNumber"`

For example: `-s "JMarsh:password@orion:49201"`

If the user name is omitted, the current user name is used. The user name in the example is "`JMarsh`".

If the password is omitted, the user is prompted to enter the password. The password in the example is "`password`". If the host name is omitted, the default is `localhost`. The host name in the example is "`orion`".

The port number is required. The default port number, `49201`, is used in the example.

-epwdfile

The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .

The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.

Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.

In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.

The following is the syntax of the commands that can be used to store an encrypted password.

Use the following syntax to be prompted for the password that will be encrypted and stored in a file.

```
stcmd store-password -epwdfile "filePath"
```

Use the following syntax to include the encrypted password in the command as clear text.

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password
"password"
```

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

This is an optional parameter. If not specified, then the platform default is not to compress.

-encrypt

Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks.

This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server.

The full syntax is: `-encrypt encryptionType`.

The types of encryption are:

RC4	RSA RC4 stream cipher (fast).
RC2_ECB	RSA RC2 block cipher (Electronic Codebook).
RC2_CBC	RSA RC2 block cipher (Cipher Block Chaining).

These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.

Note: For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.

-type

Specifies the type of item. The type is one of the stock type names, such as `changerequest`, `task`, `requirement`, `sprint`, `story`, `plan` or any custom type name that is applicable to the command.

-property

The name of the enumerated property.

-name

Specifies the name. Maximum of 254 characters.

-displayName

A name for a type that can be translated to the supported languages.

-sortOrder

A positional sort order may be specified, or a predecessor enum may be specified as an alternate. If neither sort order nor predecessor enum are specified, then the natural sort order is assumed.

-parentEnum

If the property is a hierarchical property, then a parent enum name may be optionally specified.

-predecessorEnum

The internal or display name of the enum that will be positional prior to this enum in a logical sort.

Place (position) this Enumerated Value immediately after the specified predecessor. An exception will be thrown if this Enumerated Value and the specified predecessor

belong to different hierarchies. If the context of the owning `EnumeratedProperty` is known to this and its predecessor, then the sort order of the entire list of values is realigned in accordance with this assignment. If not, then the sort order of this object alone is reassigned

Add Files: add

Use `add` to add files to a project from the command line.

You can simultaneously link the added files to a process item. All the files successfully added using this command will be linked and pinned to the tip revision of the process item. Use the `-active` option to specify the currently active process item (previously set using a StarTeam client on your workstation).

If no item is active or you prefer to use another item, use the option that indicates the type of the process item (`-cr`, `-req`, or `-task`), followed by the complete path from the root folder of the StarTeam project view to the item, using the forward slash (`/`) as a delimiter between folder names. For out-of-view process items, specify the project name and view name in front of the complete folder path. Separate the view path with a colon (`:`). For example, `-cr MyProject/RootView:ChildView/SourceCode/37` specifies `change request 37` in the `SourceCode` folder of the `ChildView` view in the `MyProject` project. During execution, the process first assumes that the process item is in the current view, and it checks the current view to determine whether the full path corresponds to a folder path within that view. If the process item is not found in the current view, it is treated as an out-of-process item, and the search for the process item begins from the project and view.

Use the `-mark` option to simultaneously mark the process item as fixed, finished, or complete, depending on its type. A StarTeam Server transaction processes the files selected to add. They succeed or fail together. Additionally, StarTeam creates a check in change package in the target view.

Syntax

The syntax for this command is:

```
stcmd{Ex} add  [[-p "projectSpecifier"] [-epwdfile "filePath"]
[-cmp] [-csf] [-encrypt encryptionType] ][-is] [-nivf] [-rp "folderPath"
| -fp "folderPath] [-l | -u | -nel] [-ro | -rw]] [-d "description"
| -rf "fileName"] [-vl "labelName] [[ -active |
[-cr | -req | -task] processItemPath] [-cp "name"] [-mark]]
[-q -pf "filterName"] [-ofp "resultsOutputFilePath"] -pi typename:"path"
[files...]
```

Parameter	Description
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands. **Note:** If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp`

Parameter	Description

	`\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.
	The full syntax is:
	`stcmd -p "userName:password@hostName:endpoint/projectName/ [viewName/][folderHierarchy/]"`
	For example:
	`stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/ SourceCode/”`
	• If the user name is omitted, the current user name is used.
	• If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
	• If the host name is omitted, the default is localhost.
	• Entering an endpoint (port number) is required. The default is 1024.
	• The project name is always required.
	• A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.
	• A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view’s root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/ Client`, use only "`SourceCode/Client`".

-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` . The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.  **Note:** When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter. In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`. The following is the syntax of the commands that can be used to store an encrypted password. Use the following syntax to be prompted for the password that will be encrypted and stored in a file. `stcmd store-password -epwdfile "filePath"` Use the following syntax to include the encrypted password in the command as clear text.

Parameter	Description

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password "password"
```

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

This is an optional parameter. If not specified, then the platform default is not to compress.

-csf

When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder.

The default is that StarTeam folders are not differentiated based on the case of letters in their names.

With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.

-encrypt

Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks.

This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server.

The full syntax is: `-encrypt encryptionType`.

The types of encryption are:

RC4	RSA RC4 stream cipher (fast).
RC2_ECB	RSA RC2 block cipher (Electronic Codebook).
RC2_CBC	RSA RC2 block cipher (Cipher Block Chaining).

These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.

Note: For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in

Parameter	Description
	the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.
-is	Applies the command to all child folders. Without this option, the command applies only to the specified folder. When this option is used with add-folder, you can add an entire branch of folders to the StarTeam folder hierarchy. When used with `add` or `ci`, the command recursively visits all modified files in all sub-folders and checks them in.
-nivf	If `-nivf` is included, then files in `Not in View` folders are also included in the action.

-rp	Overrides the working folder or working directory for the StarTeam view's root folder.
	While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways.
	While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project.
	The UNIX shell interprets a backslash (`\`) as an escape character when it precedes certain characters, such as quotation marks. As a result, an error occurs in the following example:
	`stcmd ci -p "xxx" -rp "C:\" "*"`
	which is interpreted as:
	`stcmd ci -p "xxx" -rp "C:" *"`
	To avoid a situation like this, escape the final character in "`C:\`" as follows:
	`stcmd ci -p "xxx" -rp "C:\\" "*"`
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in "`C:\orion \`":
	`stcmd ci -p "xxx" -rp "C:\orion" "*"`
	The full syntax is: `-rp "folderName" .`
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-fp	Overrides the specified StarTeam folder's working folder or working directory. This is equivalent to setting an alternate working path for the folder. While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways. While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the

Parameter	Description

	working path if your platform does not understand the path specified in the StarTeam project.
	A backslash (\) is interpreted as an escape character when it precedes quotation marks. As a result, an error occurs in the following example:
	`stcmd ci -p "xxx" -fp "C:\" "*"`
	which is interpreted as:
	`stcmd ci -p "xxx" -fp "C:" *"`
	To avoid a situation like this, escape the final character in "C:\" as follows:
	`stcmd ci -p "xxx" -fp "C:\\" "*"`
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in `C:\orion\`:
	`stcmd ci -p "xxx" -fp "C:\orion" "*"`
	The full syntax is: `-rp "folderName"`.
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-l | -u | -nel	Locks the item(s). `-l` is exclusive lock, `-u` is unlocked, and `-nel` is non exclusive lock. These items are mutually exclusive and an optional parameter.
-ro	Makes the working file read-only after this operation. Without this option, the file remains as it was prior to the operation. Usually, you use `-ro` to prevent yourself from editing a file that is not locked by you. `-ro` must be used with `-l` or`-u` or `-nel`. If you use `-ro`, you cannot use `-rw`.
-rw	Makes the working file read-write after this operation. Without this option, the file remains as it was prior to the operation. `-rw` must be used with `-l` or `-u` or `-nel`. If you use `-rw`, you cannot use `-ro`.
-d	A user specified Description. However, we continue to support `-r` as an alternate to `-d` for the description, but strictly for backward compatibility
-rf	Specifies the file name that contains the check-in reason. This is useful if the same reason should be applied to all check-ins across multiple command line runs.
-vl	Specifies a label (created using `stcmd label`) to be applied to the checked-in files. The label is enclosed in double quotation marks. This option can appear in the command more than once. The label can be either a view or revision label, but it must already exist in the application.
-active	The active process item.
-cr, -req, -task	Complete path from the project view's root folder to the change request, requirement, or task number to be used as a process item. Use the forward slash (/) as a delimiter between folder names. For out-of-view process items, specify the project name and view name in front of the complete folder path. For example: `-cr MyProject/RootView/RootFolder/SourceCode/37` This specifies change request 37 in the `SourceCode` folder (under the root folder) of the `ChildView` view in the `MyProject` project. **Note:** For in-view process items, as long as the change request, requirement, or task numbers are the unique primary descriptors of their types (true by default), it is

Parameter	Description
	sufficient simply to specify the number, with no path. The project and view names are assumed from `-p`. If a process item is specified, then the files being checked in are attached to the process item and follow the project process rules. `-cr`, `-req` or `-task` are mutually exclusive. If any one of them is specified, `-filter`/`-f` are ignored.
-cp	Name of the code page used for localization and internationalization of the content, file and folder names, keyword expansion, etc. Supported code page names are US-ASCII (the default), UTF-8, UTF-16, windows-1252, ISO-8859-1, ISO-8859-9, ISO-8859-15, windows-31j, EUC-JP, Shift_JIS, ISO-2022-JP, x-mswin-936, GB18030, x-EUC-CN, GBK.
-mark	Indicates that, if all the files are successfully added, the process item's status will be changed to fixed (for a change request), finished (for a task), or complete (for a requirement). The files are pinned to the revision with the new status. The item is not marked as fixed, finished, or complete unless all the files are successfully added.
-q	Enables quiet mode. The `-q` option is retained for backward compatibility with the old command line. If `-q` is specified, then `-pf` cannot be specified. The command will return no results.
-pf	Specifies the filter name whose associated filter properties produce the columns in the output matrix. Each command returns a result matrix. `-pf` determines the matrix columns. See `-ofp` for more information. If not specified, the primary descriptor property of the Type is returned as the command output. `-pf` does not apply to the select query command.
-ofp	Provides a file name with a fully qualified path into which to write the command output. By default, a "`|`" character separates each column in the output. A new line separates each row. The first row is the command name. The second row has the property names. All subsequent rows contain the data. If the file already exists, the output is appended to the end of the file. It is possible to override the "`|`" character separator by specifying `separator = fieldSeparator` as a parameter to the connect command. For example, `separator = ;;` specifies two adjacent semicolons (`;`) as the column separator.
files...	Specifies the files to be used in the command by name or by file name-pattern specification, such as `"*.c"`. All options are interpreted using the semantic conventions of UNIX instead of Windows because UNIX conventions are more specific. This means that `"*"`, rather than `"*.*"` means "all files." The pattern `"*.*"` means "all files with file name extensions." For example, "`star*.*`" finds `starteam.doc` and `starteam.cpp`, but not `starteam`. To find all of these, you could use `"star*"`. Without this option, the default is `"*"`. When used, this option must always be the last option. Any options after it are ignored. If you use *, rather than `"*"` to indicate all files, a UNIX shell expands it into a series of items and passes this series as a group of options to the `stcmd` command. This can cause problems, for example, when you are checking out missing files, so it is best to use `"*"` to avoid unwanted complications. If you use a set of file patterns, each pattern should be enclosed in its own set of quotation marks. For example, you can use `"*.bat" "*.c"`, but you cannot use `"*.bat *.c"`.

Parameter	Description
	Note: Always enclose this option in quotation marks. Failure to do so can result in unpredictable consequences on all supported platforms. Several special characters can be used in the file specification: ***** Matches any string including the empty string. For example, * matches any file name, with or without an extension. "xyz*" will match "xyz" and "xyz.cpp" and "xyzutyfj". **?** Matches any single character. For example, "a?c" will match "abc" but NOT "ac". **[...]** Matches any one of the characters enclosed by the left and right brackets. **-** A pair of characters separated by a hyphen (-) specifies a range of characters to be matched. If the first character following the right bracket ([ ) is an exclamation point (!) or a caret (^), the rest of the characters are not matched. Any character not enclosed in the brackets is matched. For example, "x[a-d]y" matches "xby" but not "xey". "x[!a-d]y" matches "xey" but not "xby". A hyphen (-) or right bracket ( ]) may be matched by including it as the first or last character in the bracketed set. To use an asterisk (*), question mark (?), or left bracket ([ ) in a pattern, you must precede it with the escape character (which is the backslash (\).
-pi	The type name is the internal name of a component that supports being a process item type. for example, change request, task, requirement, story, or custom component name. For a type to be a process item type, it must have a status property for check in. The syntax to use is -pi typename:"path".

Example

The following example uses add to add all .doc files with the status Not In View to User Manual, a child of the root folder StarDraw (in the StarDraw view of the StarDraw project). It locks the files and gives them the description "First draft of chapter".

```
stcmd add -rp "1024/StarDraw/StarDraw/User Manual" -l -d "First draft of
chapter" "*.doc"
```

Add Folder: add-folder

Use add-folder to add StarTeam folders to a view from the command line. You can add the folder to the root folder or any other folder in that view. The working folder for your new StarTeam folder is created by default within StarTeam, not on your workstation. The working folder has the same name as the StarTeam folder. It is a child folder of the working folder for the StarTeam folder's parent.

For example, suppose you create a StarTeam folder named Wizard. Wizard is a child of a StarTeam folder whose working folder is C:\StarDraw. Therefore, Wizard's working folder becomes C:\StarDraw\Wizard.

Using the -is option allows you to add a branch of folders to the project view's folder hierarchy. When you use -is, use either -rp or -fp to specify the folder on your workstation whose child folders will become the new StarTeam folder's child folders. Using -fp is recommended, as it specifies the path directly to the parent of those child folders. In contrast, -rp, which specifies the path to the working folder used for the

view's root folder, appends StarTeam folder names in the hierarchy from the root folder to the new folder to the path you specify. Only when you use the `-is` option do `-rp` and `-fp` have any effect on this command.

Syntax

The syntax for this command is:

```
stcmd{Ex} add-folder [[-p "projectSpecifier"] [-epwdfile "filePath"]
[-cmp] [-csf] [-encrypt encryptionType] ][-is] [-rp "folderPath" |
-fp "folderPath"] -name "folderName" [-d | -r "description"| -rf "fileName"]
[-ex "excludeType"] [-q|-pf "filterName"] [-ofp "resultsOutputFilePath"]
[-exlist "fileMask" | -exfile "fileName"]
```

Parameter	Description
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.

Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.
- A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.
- A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view’s root folder. For example, if the root folder of

Parameter	Description
	the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .
	
	The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.
	
	Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.
	
	In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.
	
	The following is the syntax of the commands that can be used to store an encrypted password.
	
	Use the following syntax to be prompted for the password that will be encrypted and stored in a file.
	`stcmd store-password -epwdfile "filePath"`
	
	Use the following syntax to include the encrypted password in the command as clear text.
	
	Note: This action does not access the network with the clear value.
	
	`stcmd store-password -epwdfile "filePath" -password "password"`
	
	After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:
	`stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"`
	
	Important:
	
	If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:
	`-p user@hostname:port/projectName.viewName -epwdfile "pathToPasswordFile"`

-cmp	Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place. Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data. This is an optional parameter. If not specified, then the platform default is not to compress.
-csf	When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder.

Parameter	Description
	The default is that StarTeam folders are not differentiated based on the case of letters in their names. With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.
-encrypt	Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks. This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server. The full syntax is: `-encrypt encryptionType`. The types of encryption are: **RC4** RSA RC4 stream cipher (fast). **RC2_ECB** RSA RC2 block cipher (Electronic Codebook). **RC2_CBC** RSA RC2 block cipher (Cipher Block Chaining). These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it. **Note:** For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.
-is	Applies the command to all child folders. Without this option, the command applies only to the specified folder. When this option is used with add-folder, you can add an entire branch of folders to the StarTeam folder hierarchy. When used with `add` or `ci`, the command recursively visits all modified files in all sub-folders and checks them in.
-rp	Overrides the working folder or working directory for the StarTeam view's root folder. While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways. While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project. The UNIX shell interprets a backslash (\) as an escape character when it precedes certain characters, such as quotation marks. As a result, an error occurs in the following example: `stcmd ci -p "xxx" -rp "C:\" "*"`

Parameter	Description

	which is interpreted as:
	`stcmd ci -p "xxx" -rp "C:" *"`
	To avoid a situation like this, escape the final character in "`C:\`" as follows:
	`stcmd ci -p "xxx" -rp "C:\\" "*"`
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in "`C:\orion\`":
	`stcmd ci -p "xxx" -rp "C:\orion" "*"`
	The full syntax is: `-rp "folderName"` .
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-fp	Overrides the specified StarTeam folder's working folder or working directory. This is equivalent to setting an alternate working path for the folder.
	While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways.
	While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project.
	A backslash (\) is interpreted as an escape character when it precedes quotation marks. As a result, an error occurs in the following example:
	`stcmd ci -p "xxx" -fp "C:\" "*"`
	which is interpreted as:
	`stcmd ci -p "xxx" -fp "C:" *"`
	To avoid a situation like this, escape the final character in "C:\" as follows:
	`stcmd ci -p "xxx" -fp "C:\\" "*"`
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in `C:\orion\`:
	`stcmd ci -p "xxx" -fp "C:\orion" "*"`
	The full syntax is: `-rp "folderName"`.
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-name	Specifies the name. Maximum of 254 characters.
-d	A user specified Description. However, we continue to support `-r` as an alternate to `-d` for the description, but strictly for backward compatibility
-rf	Specifies the file name that contains the check-in reason. This is useful if the same reason should be applied to all check-ins across multiple command line runs.
-ex	Indicates the exclude lists to be used by this new folder. Exclude lists exclude certain files or types of files from visibility. If a working file in this folder's working folder would have the status Not In View but it matches a file specification in one of the exclude lists, the application does not display it at all. It is as though the file did not exist. For example, suppose you are creating files in an application that makes automatic backup copies of each file (with the extension `.bak`) every time you save a file. Your working folder

Parameter	Description

might contain several `.bak` files, but you have no reason to add them to the project view. From the application, it is annoying to see these .bak files as possible candidates, so you exclude them. Excluding files is done on a per-folder basis. However, exclude lists can be inherited from parent folders.

The full syntax is: `-ex excludeType`

The types include:

inherit This folder will inherit any exclude lists used by its parent folder and use the exclude list specified with either `-exfile` or `-exlist` (if one is created). This is the default.

local This folder will use only the exclude list specified with either`-exfile` or `-exlist`.

none This folder will use no exclude lists, regardless of what you specify with either `-exfile` or `-exlist`.

-q Enables quiet mode. The `-q` option is retained for backward compatibility with the old command line. If `-q` is specified, then `-pf` cannot be specified. The command will return no results.

-pf Specifies the filter name whose associated filter properties produce the columns in the output matrix. Each command returns a result matrix. `-pf` determines the matrix columns. See `-ofp` for more information. If not specified, the primary descriptor property of the Type is returned as the command output. `-pf` does not apply to the select query command.

-ofp Provides a file name with a fully qualified path into which to write the command output. By default, a `"|"` character separates each column in the output. A new line separates each row. The first row is the command name. The second row has the property names. All subsequent rows contain the data. If the file already exists, the output is appended to the end of the file.

It is possible to override the `"|"` character separator by specifying `separator = fieldSeparator` as a parameter to the connect command.

For example, `separator = ;;` specifies two adjacent semicolons (`;`) as the column separator.

-exlist Specifies the local exclude list for this folder. Use a maximum of 254 characters. Enter one or more file specifications (using the standard * and ? wild cards), separated by commas, spaces, or semicolons. To include a comma, space, or semicolon as part of the specification, enclose the specification in double quotation marks: `*.exe,*.dll p*z.doc;*.t?t "test *.*"`

If you are using double-quotation marks in your exclude list or have a lengthy exclude list, we recommend that you use the `-exfile` option. With `-exlist`, each quotation mark in the exclude list needs to be preceded by the escape character for your system or shell. For example, the caret (^) works on NT systems. With`-exfile` , you do not need to use escape characters.

-exfile Specifies the path to the file that contains the local exclude list for this folder. See `-exlist` for a description of the exclude list's contents.

Example

The following example uses `add-folder` to create a folder named `Wizard` as a child of the `StarDraw` folder, the root folder of the `StarDraw` project view. In addition, it sets a local exclude list for Wizard. By default, Wizard inherits its parent folder's exclude lists and use the local one as well.

Use the `set` command to set the context of the project/view/parent folder.

```
stcmd add-folder -name "Wizard" -d "StarDraw setup wizard" -exlist "*.bak"
```

The next example creates the same folder as in the previous example. However, it includes child folders. In this case, the folder with the path `C:\Wizard` has child folders (`Source`, `Spec`, and `Doc`), all of which are added as StarTeam folders in addition to Wizard. All of the new folders (`Wizard`, `Source`, `Spec`, and `Doc`) will have the default working folders assigned to them automatically by the StarTeam Server, regardless of the setting for `-fp`. `Wizard` will be the parent of `Source`, `Spec`, and `Doc`. `StarDraw` is the parent of `Wizard`.

```
stcmd add-folder -name "Wizard" -d "StarDraw setup wizard" -is -fp "C:
\Wizard" exlist "*.bak"
```

Add Group: add-group

Use the `add-group` command to add a group to the server.

Syntax

```
stcmd{Ex} add-group –s username[:password]@host:port [-epwdfile "filePath"]
[-cmp] [-encrypt encryptionType] –name newGroupName
[–description newGroupDescription] [-parentgroup parentGroupName]
[-grouptype newGroupType]
```

Parameter	Description
-s	Identifies the StarTeam Server. The full syntax is: `-s "userName:password@host:portNumber"` For example: `-s "JMarsh:password@orion:49201"` If the user name is omitted, the current user name is used. The user name in the example is “`JMarsh`”. If the password is omitted, the user is prompted to enter the password. The password in the example is “`password`”. If the host name is omitted, the default is `localhost`. The host name in the example is “`orion`”. The port number is required. The default port number, `49201`, is used in the example.
-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` . The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore. **Note:** When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter. In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`. The following is the syntax of the commands that can be used to store an encrypted password.

Parameter	Description

Use the following syntax to be prompted for the password that will be encrypted and stored in a file.

```
stcmd store-password -epwdfile "filePath"
```

Use the following syntax to include the encrypted password in the command as clear text.

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password "password"
```

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

This is an optional parameter. If not specified, then the platform default is not to compress.

-encrypt

Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks.

This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server.

The full syntax is: `-encrypt encryptionType`.

The types of encryption are:

RC4	RSA RC4 stream cipher (fast).
RC2_ECB	RSA RC2 block cipher (Electronic Codebook).
RC2_CBC	RSA RC2 block cipher (Cipher Block Chaining).

These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.

Note: For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates

Parameter	Description
	the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.
-name	Specifies the name. Maximum of 254 characters.
-description	A user specified Description.
-parentGroup	Name of the parent group. If `-parentgroup` is not specified, the newly created group is assigned to the `All Users` group.
-groupType	If `-grouptype` is not specified, the new group type is marked `UNKNOWN`. Legal values for `-grouptype` are "`team`" or "`user`".

Add Project: add-project

Use `add-project` to add a project to a StarTeam Server configuration from the command line. When a project is created, its root view and the root folder for the root view are also created. In this command, the `-rp` option specifies the working folder for that root folder.

Using `-is` allows you to use the working folder's child folders as the root folder's child folders in the StarTeam folder hierarchy.

Syntax

The syntax for this command is:

```
stcmd{Ex} add-project [-epwdfile "filePath"] [-cmp]
[-encrypt encryptionType] [-is] [-q] -s "serverName" -name "projectName"
-rp "folderPath" [-d "description"] [-kw "fileMask" |-kwfile "fileName"]
[-ex "excludeType"] [-exlist "fileMask" |-exfile "fileName"]
```

Parameter	Description
-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` . The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore. **Note:** When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter. In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`. The following is the syntax of the commands that can be used to store an encrypted password. Use the following syntax to be prompted for the password that will be encrypted and stored in a file. `stcmd store-password -epwdfile "filePath"` Use the following syntax to include the encrypted password in the command as clear text.

Parameter	Description

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password "password"
```

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

This is an optional parameter. If not specified, then the platform default is not to compress.

-encrypt

Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks.

This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server.

The full syntax is: `-encrypt encryptionType`.

The types of encryption are:

RC4	RSA RC4 stream cipher (fast).
RC2_ECB	RSA RC2 block cipher (Electronic Codebook).
RC2_CBC	RSA RC2 block cipher (Cipher Block Chaining).

These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.

Note: For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.

-is

Applies the command to all child folders. Without this option, the command applies only to the specified folder. When this option is used with add-folder, you can add an entire branch of folders to the StarTeam folder hierarchy.

When used with `add` or `ci`, the command recursively visits all modified files in all sub-folders and checks them in.

Parameter	Description
-q	Enables quiet mode. The `-q` option is retained for backward compatibility with the old command line. If `-q` is specified, then `-pf` cannot be specified. The command will return no results.
-s	Identifies the StarTeam Server. The full syntax is: `-s "userName:password@host:portNumber"` For example: `-s "JMarsh:password@orion:49201"` If the user name is omitted, the current user name is used. The user name in the example is "`JMarsh`". If the password is omitted, the user is prompted to enter the password. The password in the example is "`password`". If the host name is omitted, the default is `localhost`. The host name in the example is "`orion`". The port number is required. The default port number, `49201`, is used in the example.
-name	Specifies the name. Maximum of 254 characters.
-nivf	If `-nivf` is included, then files in `Not in View` folders are also included in the action.

-rp	Overrides the working folder or working directory for the StarTeam view's root folder.
	
	While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways.
	
	While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project.
	
	The UNIX shell interprets a backslash (\) as an escape character when it precedes certain characters, such as quotation marks. As a result, an error occurs in the following example:
	`stcmd ci -p "xxx" -rp "C:\" "*"`
	
	which is interpreted as:
	`stcmd ci -p "xxx" -rp "C:" *"`
	
	To avoid a situation like this, escape the final character in "`C:\`" as follows:
	`stcmd ci -p "xxx" -rp "C:\\" "*"`
	
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in "`C:\orion\`":
	`stcmd ci -p "xxx" -rp "C:\orion" "*"`
	
	The full syntax is: `-rp "folderName"` .
	
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-d	A user specified Description. However, we continue to support `-r` as an alternate to `-d` for the description, but strictly for backward compatibility
-kw	Specifies the file extensions with which you want to use keywords. Use a maximum of 254 characters. Enter one or more file specifications (using the standard * and ? wild cards), separated by commas, spaces, or semicolons.

Parameter	Description
	To include a comma, space, or semicolon as part of the specification, enclose the specification in double quotation marks. For example: `*.cpp,*.h p*z.doc;*.t?t "test *.*"`
	If you are using double-quotation marks in your keyword list or have a lengthy list, we recommend that you use the `-kwfile` option. With `-kwlist`, each quotation mark in the keyword list needs to be preceded by the escape character for your system or shell. For example, the caret (^) works on NT systems. With `-kwfile`, you do not need to use escape characters.
-kwfile	Specifies the path to the file containing the file extensions with which you want to use keywords. If you use `-kwfile`, you cannot use `-kw`.
-ex	Indicates the exclude lists to be used by this new folder. Exclude lists exclude certain files or types of files from visibility. If a working file in this folder's working folder would have the status Not In View but it matches a file specification in one of the exclude lists, the application does not display it at all. It is as though the file did not exist. For example, suppose you are creating files in an application that makes automatic backup copies of each file (with the extension `.bak`) every time you save a file. Your working folder might contain several `.bak` files, but you have no reason to add them to the project view. From the application, it is annoying to see these .bak files as possible candidates, so you exclude them. Excluding files is done on a per-folder basis. However, exclude lists can be inherited from parent folders. The full syntax is: `-ex excludeType` The types include: **inherit** This folder will inherit any exclude lists used by its parent folder and use the exclude list specified with either `-exfile` or `-exlist` (if one is created). This is the default. **local** This folder will use only the exclude list specified with either`-exfile` or `-exlist`. **none** This folder will use no exclude lists, regardless of what you specify with either `-exfile` or `-exlist`.
-exlist	Specifies the local exclude list for this folder. Use a maximum of 254 characters. Enter one or more file specifications (using the standard * and ? wild cards), separated by commas, spaces, or semicolons. To include a comma, space, or semicolon as part of the specification, enclose the specification in double quotation marks: `*.exe,*.dll p*z.doc;*.t?t "test *.*"` If you are using double-quotation marks in your exclude list or have a lengthy exclude list, we recommend that you use the `-exfile` option. With `-exlist`, each quotation mark in the exclude list needs to be preceded by the escape character for your system or shell. For example, the caret (^) works on NT systems. With`-exfile` , you do not need to use escape characters.
-exfile	Specifies the path to the file that contains the local exclude list for this folder. See `-exlist` for a description of the exclude list's contents.

Example

The following example uses `add-project` to create a project named `Integrations` on the computer named `Orion`. (Orion is running an instance of the StarTeam Server with a server configuration that uses

port 1024.) This command creates the project, specifies that the data sent between workstations and the server should be compressed and encrypted, and gives the project a description.

```
stcmd add-project -s "JMarsh:password@Orion:1024" -cmp -encrypt "RC4"
-name "Integrations" -rp "C:\integrations" -d "integrations between our
products and our partner’s products"
```

Add Property: add-property

Use the `add-property` command to add a property to an existing component the server.

Important: Requires administrative permissions.

Syntax

```
stcmd{Ex} add-property [-epwdfile "filePath"] [-e] [-cmp] [-encrypt
encryptionType]
–s username[:password]@host:port –type typeName –xml xmlDefinition [–xmlfile
pathToXMLDefiniton]
```

The component type must be specified via the `-type typeName` parameter.

Refer to the Server documentation for the XML schema definition of component properties.

Parameter	Description

-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .
	
	The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.
	
	  **Note:** When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.
	
	In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.
	
	The following is the syntax of the commands that can be used to store an encrypted password.
	
	Use the following syntax to be prompted for the password that will be encrypted and stored in a file.
	`stcmd store-password -epwdfile "filePath"`
	
	Use the following syntax to include the encrypted password in the command as clear text.
	
	  **Note:** This action does not access the network with the clear value.
	
	`stcmd store-password -epwdfile "filePath" -password "password"`
	
	After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:
	`stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/ SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"`

Parameter	Description

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

This is an optional parameter. If not specified, then the platform default is not to compress.

-encrypt

Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks.

This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server.

The full syntax is: `-encrypt encryptionType`.

The types of encryption are:

RC4	RSA RC4 stream cipher (fast).
RC2_ECB	RSA RC2 block cipher (Electronic Codebook).
RC2_CBC	RSA RC2 block cipher (Cipher Block Chaining).

These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.

Note: For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.

-s

Identifies the StarTeam Server. The full syntax is: `-s "userName:password@host:portNumber"`

For example: `-s "JMarsh:password@orion:49201"`

If the user name is omitted, the current user name is used. The user name in the example is "`JMarsh`".

If the password is omitted, the user is prompted to enter the password. The password in the example is "`password`". If the host name is omitted, the default is `localhost`. The host name in the example is "`orion`".

The port number is required. The default port number, `49201`, is used in the example.

Parameter	Description
-type	Specifies the type of item. The type is one of the stock type names, such as `changerequest`, `task`, `requirement`, `sprint`, `story`, `plan` or any custom type name that is applicable to the command.
–xml	The XML definition must be specified. It can be inline via the `-xml` parameter, or provided via a file (the `-xmlfile` parameter).
–xmlfile	The XML definition must be specified. It can be inline via the `-xml` parameter, or provided via a file (the `-xmlfile` parameter).

Add Type: add-type

Use the `add-type` command to add a type to the server.

Important: Requires administrative permissions.

Syntax

```
stcmd{Ex} add-type [-epwdfile "filePath"] [-cmp] [-encrypt encryptionType]
–s username[:password]@host:port –xml xmlDefinition [–xmlfile
pathToXMLDefiniton] [-verify]
```

Refer to the server documentation for the XML schema definition of custom components.

Parameter	Description

-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .
	
	The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.
	
	  **Note:** When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.
	
	In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.
	
	The following is the syntax of the commands that can be used to store an encrypted password.
	
	Use the following syntax to be prompted for the password that will be encrypted and stored in a file.
	`stcmd store-password -epwdfile "filePath"`
	
	Use the following syntax to include the encrypted password in the command as clear text.
	
	  **Note:** This action does not access the network with the clear value.
	
	`stcmd store-password -epwdfile "filePath" -password "password"`

Parameter	Description

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

This is an optional parameter. If not specified, then the platform default is not to compress.

-encrypt

Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks.

This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server.

The full syntax is: `-encrypt encryptionType`.

The types of encryption are:

RC4	RSA RC4 stream cipher (fast).
RC2_ECB	RSA RC2 block cipher (Electronic Codebook).
RC2_CBC	RSA RC2 block cipher (Cipher Block Chaining).

These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.

Note: For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.

-s

Identifies the StarTeam Server. The full syntax is: `-s "userName:password@host:portNumber"`

For example: `-s "JMarsh:password@orion:49201"`

If the user name is omitted, the current user name is used. The user name in the example is "`JMarsh`".

Parameter	Description
	If the password is omitted, the user is prompted to enter the password. The password in the example is “password”. If the host name is omitted, the default is localhost. The host name in the example is “orion”. The port number is required. The default port number, 49201, is used in the example.
–xml	The XML definition must be specified. It can be inline via the -xml parameter, or provided via a file (the -xmlfile parameter).
–xmlfile	The XML definition must be specified. It can be inline via the -xml parameter, or provided via a file (the -xmlfile parameter).
-verify	Does a test that the XML is well formed, and that the type will be able to be created, without actually creating the type on the server. If the schema is incorrect, a server exception will be thrown.

Add Views: add-view

Use add-view to add a view to a StarTeam Server configuration from the command line. When the view is created, its parent view is the view specified with the -p option and its root folder is the folder specified with the -p option or it is the directory used in a previous session when not specifying a directory using the -p option. In this command, the -rp option specifies the working folder for the root folder. Use the following options to create the following types of views:

Syntax

The syntax for this command is:

```
stcmd{Ex} add-view  [[-p "projectSpecifier"] [-epwdfile "filePath"]
[-cmp] [-encrypt encryptionType] ][-rp "folderPath"] [-d "description"]
[-dr [-ro | -ba | -bn [-cst | -cfgl "labelName" | -cfgp "stateName" |
-cfgd "asOfDate" [-pattern "date-pattern"]]]]
```

Note: The specification of a parameter such as -ba requires the server configuration setting **Disable Advanced Views** to be turned off. It is on by default. You must specify -cfgd, -cfgp or -cfgl so that the server does not treat the view as an advanced view.

For example:

```
add-view -p "user:password@host:port/project" -name "viewname"
-rp "working folder" -dr -ba -cfgd "10/17/2014"  -pattern "M/d/y"
```

If you use the syntax above, then you don't need the option DisableAdvancedViews to have been set in the starteam-configs.xml file.

Note that if you use -cfgd, then the date time specified must be after the parent view was created.

Parameter	Description
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. -p is retained for backward compatibility. Commands using -p continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See connect and set for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.

Parameter	Description

Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.
- A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.
- A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view’s root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile

The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .

The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.

Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.

In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.

Parameter	Description

The following is the syntax of the commands that can be used to store an encrypted password.

Use the following syntax to be prompted for the password that will be encrypted and stored in a file.

```
stcmd store-password -epwdfile "filePath"
```

Use the following syntax to include the encrypted password in the command as clear text.

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password "password"
```

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

This is an optional parameter. If not specified, then the platform default is not to compress.

-encrypt

Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks.

This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server.

The full syntax is: `-encrypt encryptionType`.

The types of encryption are:

RC4	RSA RC4 stream cipher (fast).
RC2_ECB	RSA RC2 block cipher (Electronic Codebook).
RC2_CBC	RSA RC2 block cipher (Cipher Block Chaining).

These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.

Note: For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the

Parameter	Description
	keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.

-rp	Overrides the working folder or working directory for the StarTeam view's root folder.
	
	While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways.
	
	While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project.
	
	The UNIX shell interprets a backslash (\) as an escape character when it precedes certain characters, such as quotation marks. As a result, an error occurs in the following example:
	`stcmd ci -p "xxx" -rp "C:\" "*"`
	
	which is interpreted as:
	`stcmd ci -p "xxx" -rp "C:" *"`
	
	To avoid a situation like this, escape the final character in "`C:\`" as follows:
	`stcmd ci -p "xxx" -rp "C:\\" "*"`
	
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in "`C:\orion\`":
	`stcmd ci -p "xxx" -rp "C:\orion" "*"`
	
	The full syntax is: `-rp "folderName"` .
	
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-d	A user specified Description. However, we continue to support `-r` as an alternate to `-d` for the description, but strictly for backward compatibility
-dr	Specifies a derived view. All views, except blank views are derived. See also `-ba`, `-bn`, and `-ro`. When used without `-ba`, `-bn`, or `-ro`, a read/write reference view is created. The configuration of a read/write reference view is the same configuration as its parent view. Therefore, using `-dr` without `-ba`, `-bn`, or `-ro`, but with `-cfgl`, `-fgp`, or `-cfgd` results in an error message. When this option is not used, a blank view is created. For blank views, the value of the view property named **Set Items Shared Into View To Branch On Change** is initially cleared. Indicates the type of view to create. • Use `-dr` to create a read/write reference view. • Use `-dr -ro` to create a read-only reference view. • Use `-dr -ba` to create a branching view in which the behavior of existing items is set to branch on change. • Use `-dr -bn` to create a branching view in which the behavior of existing items is not set to branch on change. • If you do not use `-dr`, a blank view is created.
-ro	When used with `-dr`, specifies a read-only reference view.

Parameter	Description
-ba	When used with `-dr`, specifies a branching view in which the behavior of existing items is set to branch on change. The value of the view property **Set Items Shared Into View To Branch On Change** is initially set. This option must be used with `-dr`.
-bn	When used with `-dr`, specifies a branching view in which the behavior of existing items is not set to branch on change. The value of the view property **Set Items Shared Into View To Branch On Change** is initially cleared. This option must be used with `-dr`.
-cst	Configures the view as of the current time on the StarTeam Server. This option must be used with one of the following combinations: `-dr -ro` , `-dr -ba`, or `-dr -bn` . `-cst`, `-cfgd`, `-cfpg`, and `-cfgl` are mutually exclusive.
-cfgl	Configures the view using the specified label. Without `-cfgl`, `-cfgp`, or `-cfgd`, the view's current configuration is used.
-cfgd	Configures the view as of the specified date/time. Examples include: "12/29/13 10:52 AM" "December 29, 2013 10:52:00 AM PST" "Monday, December 29, 2013 10:52:00 AM PST"
-cfgp	Configures the view using the specified promotion state.
-pattern	Qualifies the datetime. It can be specified wherever a date-time is specified, such as `-cfgd`, `-vd`, etc. The pattern must match any valid pattern supported by the java JDK in `java.text.SimpleDateFormat.applyLocalizedPattern(String)`. The pattern may be localized. For every command that takes a `-pattern` parameter, a `-locale` parameter is optionally available. This is the "two character country code".

Examples

The following example uses `add-view` to create a branching view named Maintenance 5.1 on the computer named `Orion`. (Orion is running an instance of the StarTeam Server with a server configuration that uses port 1024.)

This command creates the view as a child of the existing `StarDraw` view and uses the `StarDraw` folder as its root folder. The new view is based on the label used for the last build of the 5.1 product before it shipped (Build 403). It has a working folder that is different from the parent's working folder. All existing items in the view will have their behavior set to branch on change.

Use `-p` with `add-view` or the stateful `set` command to set the context of the project/view/parent folder.

```
stcmd add-view -rp "C:\StarDraw\Maintenance 5.1" -d "Maintenance view for 5.1
product release" -dr -ba -cfgl "Build 403"
```

The following example uses `add-view` to create a read/write reference view named `Rooted At Source Code` on the computer named `Orion`. This command creates the view as a child of the existing `StarDraw` view and uses the `SourceCode` folder as its root folder. It has the same working folder as its parent. Because a read/write reference view must have the same configuration as its parent, none of the `-cfgl`, `-cfgp`, and `-cfgd` options can be used.

```
stcmd add-view -cmp -encrypt "RC4" -name "Rooted At SourceCode" -d "StarDraw
main view but with SourceCode folder as the root of the hierarchy" -dr
```

Apply Labels: apply-label

Use `apply-label` to label specified file revisions with view or revision labels. The labels must already exist in StarTeam. You can create the labels in StarTeam with the `label` command.

Syntax

The syntax for this command is:

```
stcmd{Ex} apply-label [-p "projectSpecifier" [-epwdfile "filePath"]
[-cmp] [-csf] [-encrypt encryptionType] ] [-is]
[-rp "folderPath" | -fp "folderPath"] [-ifp "file path"|-filter "fileStatus"]
[-vl "labelName" | -vd "asOfDate"]
[-pattern "date-pattern"]  | -vn revisionNumber | -vp promotionStateName]
-lbl "labelName" [-q|-pf "filterName"] [-u] [-l]
[-ofp "resultsOutputpath"][-folder folderHierarchyPath [-scope FO|FC|FTC]] [-
iip] [files...]
```

Note: `-ifp`, `-folder`, and `files` are mutually exclusive.

Parameter	Description
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.

Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.
- A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For

Parameter	Description

example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.

- A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view's root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile

The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .

The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.

Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.

In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.

The following is the syntax of the commands that can be used to store an encrypted password.

Use the following syntax to be prompted for the password that will be encrypted and stored in a file.

```
stcmd store-password -epwdfile "filePath"
```

Use the following syntax to include the encrypted password in the command as clear text.

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password "password"
```

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile` "`filePath`" as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

Parameter	Description
	This is an optional parameter. If not specified, then the platform default is not to compress.
-csf	When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder. The default is that StarTeam folders are not differentiated based on the case of letters in their names. With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.
-encrypt	Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks. This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server. The full syntax is: `-encrypt encryptionType`. The types of encryption are: **RC4** RSA RC4 stream cipher (fast). **RC2_ECB** RSA RC2 block cipher (Electronic Codebook). **RC2_CBC** RSA RC2 block cipher (Cipher Block Chaining). These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it. **Note:** For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.
-is	Applies the command to all child folders. Without this option, the command applies only to the specified folder. When this option is used with add-folder, you can add an entire branch of folders to the StarTeam folder hierarchy. When used with `add` or `ci`, the command recursively visits all modified files in all sub-folders and checks them in.
-iip	Ignores invalid path. If this parameter is specified, and if the folder path in the command is specified and invalid, then the command will exit successfully without throwing an exception.
-rp	Overrides the working folder or working directory for the StarTeam view's root folder. While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example,

Parameter	Description

	UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways.
	While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project.
	The UNIX shell interprets a backslash (\) as an escape character when it precedes certain characters, such as quotation marks. As a result, an error occurs in the following example:
	`stcmd ci -p "xxx" -rp "C:\" "*"`
	which is interpreted as:
	`stcmd ci -p "xxx" -rp "C:" *"`
	To avoid a situation like this, escape the final character in "`C:\`" as follows:
	`stcmd ci -p "xxx" -rp "C:\\" "*"`
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in "`C:\orion\`":
	`stcmd ci -p "xxx" -rp "C:\orion" "*"`
	The full syntax is: `-rp "folderName"` .
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-fp	Overrides the specified StarTeam folder's working folder or working directory. This is equivalent to setting an alternate working path for the folder.
	While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways.
	While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project.
	A backslash (\) is interpreted as an escape character when it precedes quotation marks. As a result, an error occurs in the following example:
	`stcmd ci -p "xxx" -fp "C:\" "*"`
	which is interpreted as:
	`stcmd ci -p "xxx" -fp "C:" *"`
	To avoid a situation like this, escape the final character in "C:\" as follows:
	`stcmd ci -p "xxx" -fp "C:\\" "*"`
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in `C:\orion\`:
	`stcmd ci -p "xxx" -fp "C:\orion" "*"`
	The full syntax is: `-rp "folderName"`.
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-ifp	Specifies a fully qualified path to a file which contains a list of item IDs. The items associated with item IDs are associated to the label. If `-ifp` is specified, `-filter "fileStatus"` cannot be specified.

Parameter	Description
-filter	Specifies a string of one or more characters, each of which represents a file status. Never include spaces or other white space in this string. Only files that currently have the specified status(es) will be actioned. Does not apply to files that are `Not In View`. • `C = Current` • `M = Modified` • `O = Out of date` • `N = Not In View` • `I = Missing` • `G = Merge` • `U = Unknown` For example, using `CM` applies a command only to files with a status of `Current` or `Modified`. `-filter` takes precedence over `-f NCI`. If you use `G`, `O`, or `U`, you must also specify `-I` or `-o`. Otherwise the `G`, `O`, or `U` is ignored. `-filter` also takes precedence over `-f NCO`. If you use `G`, `M`, or `U`, you must also specify `-o` to force the checkout operation. Otherwise, the `G`, `M`, or `U` is ignored.
-vl	Specifies a label (created using `stcmd label`) to be applied to the checked-in files. The label is enclosed in double quotation marks. This option can appear in the command more than once. The label can be either a view or revision label, but it must already exist in the application.
-vd	Specifies the as-of date/time used to identify the revisions to be checked out. The last revision before the specified date/time is the one checked out for each file. See the date/time examples for `-cfgd`.
-pattern	Qualifies the datetime. It can be specified wherever a date-time is specified, such as `-cfgd`, `-vd`, etc. The pattern must match any valid pattern supported by the java JDK in `java.text.SimpleDateFormat.applyLocalizedPattern(String)`. The pattern may be localized. For every command that takes a `-pattern` parameter, a `-locale` parameter is optionally available. This is the "two character country code".
-vn	Specifies the revision number
-vp	Specifies the promotion state.
-lbl	Specifies the label name on which to perform the action. This option can be used more than once. The application action is for all of the labels on the specified file or revisions.
-q	Enables quiet mode. The `-q` option is retained for backward compatibility with the old command line. If `-q` is specified, then `-pf` cannot be specified. The command will return no results.
-pf	Specifies the filter name whose associated filter properties produce the columns in the output matrix. Each command returns a result matrix. `-pf` determines the matrix columns. See `-ofp` for more information. If not specified, the primary descriptor property of the Type is returned as the command output. `-pf` does not apply to the select query command.
-l | -u | -nel	Locks the item(s). `-l` is exclusive lock, `-u` is unlocked, and `-nel` is non exclusive lock. These items are mutually exclusive and an optional parameter.
-ofp	Provides a file name with a fully qualified path into which to write the command output. By default, a "|" character separates each column in the output. A new line separates each row. The first row is the command name. The second row has the property names. All

Parameter	Description
	subsequent rows contain the data. If the file already exists, the output is appended to the end of the file. It is possible to override the "|" character separator by specifying `separator = fieldSeparator` as a parameter to the connect command. For example, `separator = ;;` specifies two adjacent semicolons (`;`) as the column separator.
- folder	Specifies the folder/path on which to apply the label. This path must match a StarTeam starting folder from the root folder down to the node folder and begin and end with a `/` or `\`. If a folder path is specified, then the files argument is ignored.
-scope	This argument must use one of the following: • `FO` - folder only. • `FC` - folder and contents. • `FTC` - folder tree and contents. If `-scope` is not specified, the default behavior for `-folder` is `FTC`.
files...	Specifies the files to be used in the command by name or by file name-pattern specification, such as `"*.c"`. All options are interpreted using the semantic conventions of UNIX instead of Windows because UNIX conventions are more specific. This means that `"*"`, rather than `"*.*"` means “all files.” The pattern `"*.*"` means “all files with file name extensions.” For example, `“star*.*”` finds `starteam.doc` and `starteam.cpp`, but not `starteam`. To find all of these, you could use `"star*"`. Without this option, the default is `"*"`. When used, this option must always be the last option. Any options after it are ignored. If you use `*`, rather than `"*"` to indicate all files, a UNIX shell expands it into a series of items and passes this series as a group of options to the `stcmd` command. This can cause problems, for example, when you are checking out missing files, so it is best to use `"*"` to avoid unwanted complications. If you use a set of file patterns, each pattern should be enclosed in its own set of quotation marks. For example, you can use `"*.bat" "*.c"`, but you cannot use `"*.bat *.c"`. **Note:** Always enclose this option in quotation marks. Failure to do so can result in unpredictable consequences on all supported platforms. Several special characters can be used in the file specification: ***** Matches any string including the empty string. For example, `*` matches any file name, with or without an extension. "xyz*" will match "xyz" and "xyz.cpp" and "xyzutyfj". **?** Matches any single character. For example, `"a?c"` will match `"abc"` but NOT `"ac"`. **[...]** Matches any one of the characters enclosed by the left and right brackets. **-** A pair of characters separated by a hyphen (`-`) specifies a range of characters to be matched. If the first character following the right bracket (`[`) is an exclamation point (`!`) or a caret (`^`), the rest of the characters are not matched. Any character not enclosed in the brackets is matched. For example, `"x[a-d]y"` matches `"xby"` but not `"xey"`. `"x[!a-d]y"` matches `"xey"` but not `"xby"`.

Parameter	Description
	A hyphen (-) or right bracket ( ]) may be matched by including it as the first or last character in the bracketed set. To use an asterisk (*), question mark (?), or left bracket ([ ) in a pattern, you must precede it with the escape character (which is the backslash (\).

Examples

The following example uses `apply-label` to apply the label `Beta` to the specified folder `Edge`:

```
apply-label –p “user:password@host:port/project/view” –lbl “Beta” –folder “/
Stargate/dev/src/Edge/’ –scope FO
```

The following example uses `apply-label` to apply the label `Beta` to files in `User Manual`, a child of the root folder `StarDraw` (in the `StarDraw` view of the `StarDraw` project). StarTeam applies the label to the revisions of those files that were current at noon on `July 7, 2013`.

Use `-p` with `apply-label` or the stateful `set` command to set the context of the project/view/parent folder.

```
stcmd apply-label -rp "1024/StarDraw/StarDraw/User Manual" -vd "07/07/13
12:00 PM" -lbl "Beta" -u -l "*"
```

Add User: add-user

Use the `add-user` command to add a user to the StarTeam Server. Only users with administrative privileges can run this command.

Syntax

The syntax for this command is:

```
stcmd{Ex} add-user –s “username:password@host:port” –logonname “logon name”
–password “password” [-name “user full name” ] [-group “group name”]
[-phone “phone number”] [-email “email address”] [-address “postal address”]
[-voicemail “voice mail number”] [-pager “pager number”] [-fax “fax number”][-
distinguishedname “distinguished name”]
[-directoryservicevalidation] [-fixedlicense]
```

`logonname` and `password` are required. All other parameters are optional.

Parameter	Description
-s	Identifies the StarTeam Server. The full syntax is: `-s "userName:password@host:portNumber"` For example: `-s "JMarsh:password@orion:49201"` If the user name is omitted, the current user name is used. The user name in the example is “`JMarsh`”. If the password is omitted, the user is prompted to enter the password. The password in the example is “`password`”. If the host name is omitted, the default is `localhost`. The host name in the example is “`orion`”. The port number is required. The default port number, `49201`, is used in the example.
-logon name	Name used for logging in to the Server.
-password	User's password.

Parameter	Description
-name	The user's full name. If you don't specify `-name`, then logon name is used as the name
-group	Name of the group the user is added to. If you don't specify `-group`, then the user is added to the `All Users` group. Otherwise, You must specify a valid group name with `-group`.
-phone	Phone number of the user.
-email	User's email.
-address	Postal address of user.
-voicemail	User's voicemail number.
-pager	User's page number.
-fax	User's fax number.
-distinguishedname	User's directory service distinguished name
-directoryservicevalidation	If specified, turns on directory service validation.
-fixedlicense	If specified, sets the user to inherit a fixed license.

Attachment: attach

Use the `attach` command to attach a file to a project.

The `attach` command returns the `ID` of the newly created or updated attachment.

Syntax

The syntax for this command is:

```
stcmd{Ex} attach –p “user:pwd@host:port/project/view”
[ -epwdile “pathToPasswordFile” ] –type “typeName” –id itemID
-afp attachmentFilePath  [ -an “attachmentName” ] [ -rmv ]
```

Parameter	Description
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.  **Note:** If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

Parameter	Description

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.
- A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.
- A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view’s root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile

The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .

The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.

Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.

In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.

The following is the syntax of the commands that can be used to store an encrypted password.

Use the following syntax to be prompted for the password that will be encrypted and stored in a file.

```
stcmd store-password -epwdfile "filePath"
```

Use the following syntax to include the encrypted password in the command as clear text.

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password "password"
```

Parameter	Description

	After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile` "`filePath`" as parameters. For example:
	`stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/ SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"`
	Important:
	If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:
	`-p user@hostname:port/projectName.viewName -epwdfile "pathToPasswordFile"`

-type	Specifies the type of item. The type is one of the stock type names, such as `changerequest`, `task`, `requirement`, `sprint`, `story`, `plan` or any custom type name that is applicable to the command.
-id	Can be either the unique item (view member) ID of the item. Find `View Member IDs` by looking in the property lists of the StarTeam Cross-Platform Client or can be queried using the `select` command, or the primary descriptor of the item, for example, `changerequest`, `task`, or `requirement` number.
-afp	Provides the fully qualified path to the attachment on disk.
-an	Describes an optional attachment name. If this is not specified, the file name is used as the attachment name. If an attachment with this name already exists, attached to this item, then the original attachment content is replaced.
-rmv	Is an optional parameter that specifies that the attachment with the given name (either from `-afp` or `-an`) should be removed.

Branch: branch

Use `branch` to change the behavior of any StarTeam artifact.

Syntax

The syntax for the command is:

```
branch -p "user:pwd@host:port/project/view" [
-epwdile "pathToPasswordFile" ] [ -csf ] -type "typeName" -id itemID [ -boc
true|false ] [ -vctip | -vclbl labelid |
-vcps promotionstateid | -vcdttm "datetimestring"
-pattern "datetimepattern" ]
```

Parameter	Description
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.

Parameter **Description**

Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.
- A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.
- A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view’s root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile

The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .

The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.

Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.

In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.

Parameter	Description

	The following is the syntax of the commands that can be used to store an encrypted password.
	Use the following syntax to be prompted for the password that will be encrypted and stored in a file.
	`stcmd store-password -epwdfile "filePath"`
	Use the following syntax to include the encrypted password in the command as clear text.
	Note: This action does not access the network with the clear value.
	`stcmd store-password -epwdfile "filePath" -password "password"`
	After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:
	`stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"`
	Important:
	If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:
	`-p user@hostname:port/projectName.viewName -epwdfile "pathToPasswordFile"`

-csf	When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder. The default is that StarTeam folders are not differentiated based on the case of letters in their names. With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.
-type	Specifies the type of item. The type is one of the stock type names, such as `changerequest`, `task`, `requirement`, `sprint`, `story`, `plan` or any custom type name that is applicable to the command.
-id	Specifies the unique item (view member) ID of the item. Look in the property lists of the CPC or query using the select command to find the View Member IDs. `-id` can also specify the primary descriptor of the item; e.g. file name, folder name, change request number.

The remaining parameters are optional. If not specified, they default to the view level behavior.

Parameter	Description
-boc	Set to `true` for branch on change ON, `false` for branch on change OFF.
-vctip	Specifies that the new share floats.
-vclabel	Specifies that the new share is pinned to the specified label. A label is identified by either the unique integer ID or the label name.

Parameter	Description
-vcps	Specifies that the new share is pinned to the specified promotion state. A promotion state is identified by either the unique integer promotion state ID or the promotion state name.
-vcdttm	Specifies that the new share is pinned to the specified point in time. An optional `-pattern` describes the syntax of the date time string, just like in other commands.

Check In Files: ci

Use `ci` to check files into a StarTeam repository (or vault) from a working folder using the command line.

You can simultaneously link the new file revisions to a process item. All the files successfully added using this command will be linked and pinned to the tip revision of the process item. Use the `-active` option to specify the currently active process item (previously set using a StarTeam client on your workstation).

If no item is active or you prefer to use another item, use the option that indicates the type of the process item (`-cr`, `-req`, or `-task`), followed by the complete path from the root folder of the StarTeam project view to the item, using the forward slash (`/`) as a delimiter between folder names. For out-of-view process items, specify the project name and view name in front of the complete folder path. Separate the view path with a colon (`:`). For example, `-cr MyProject/RootView:ChildView/SourceCode/37` specifies `change request 37` in the `SourceCode` folder of the `ChildView` view in the `MyProject` project. During execution, the process first assumes that the process item is in the current view, and it checks the current view to determine whether the full path corresponds to a folder path within that view. If the process item is not found in the current view, it is treated as an out-of-view process item, and the search for the process item begins from the project and view.

Use the `-mark` option to simultaneously mark the process item as fixed, finished, or complete, depending on its type.

Syntax

The syntax for this command is as follows:

```
stcmd{Ex} ci [[-p "projectSpecifier"] [-epwdfile "filePath"] [-cmp] [-csf]
[-encrypt encryptionType]][-rp "folderPath" | -fp "folderPath"]
[-filter "fileStatus"] [-l | -u | -nel] [-is] [-ro | -rw]] [-vl "labelName"]
[-nomove] [-f NCI] [-o] [-d|-r "comment" | -rf " fileName "][[ -active |
[-cr | -req | -task ] processItemPath] [-mark]] [-q|-pf "filterName"]
[-ofp "resultsOutputFilePath] [-cp "names"] -pi typename:"path" [files...]
```

Parameter	Description
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.

Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp`

Parameter	Description

`\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.
- A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.
- A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view’s root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile

The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .

The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.

Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.

In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.

The following is the syntax of the commands that can be used to store an encrypted password.

Use the following syntax to be prompted for the password that will be encrypted and stored in a file.

```
stcmd store-password -epwdfile "filePath"
```

Use the following syntax to include the encrypted password in the command as clear text.

Parameter	Description

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password "password"
```

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

This is an optional parameter. If not specified, then the platform default is not to compress.

-csf

When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder.

The default is that StarTeam folders are not differentiated based on the case of letters in their names.

With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.

-encrypt

Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks.

This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server.

The full syntax is: `-encrypt encryptionType`.

The types of encryption are:

RC4	RSA RC4 stream cipher (fast).
RC2_ECB	RSA RC2 block cipher (Electronic Codebook).
RC2_CBC	RSA RC2 block cipher (Cipher Block Chaining).

These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.

Parameter	Description
	Note: For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.

-rp	Overrides the working folder or working directory for the StarTeam view's root folder.
	
	While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways.
	
	While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project.
	
	The UNIX shell interprets a backslash (\) as an escape character when it precedes certain characters, such as quotation marks. As a result, an error occurs in the following example:
	`stcmd ci -p "xxx" -rp "C:\" "*"`
	
	which is interpreted as:
	`stcmd ci -p "xxx" -rp "C:" *"`
	
	To avoid a situation like this, escape the final character in "`C:\`" as follows:
	`stcmd ci -p "xxx" -rp "C:\\" "*"`
	
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in "`C:\orion \`":
	`stcmd ci -p "xxx" -rp "C:\orion" "*"`
	
	The full syntax is: `-rp "folderName"` .
	
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-fp	Overrides the specified StarTeam folder's working folder or working directory. This is equivalent to setting an alternate working path for the folder. While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways. While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project. A backslash (\) is interpreted as an escape character when it precedes quotation marks. As a result, an error occurs in the following example: `stcmd ci -p "xxx" -fp "C:\" "*"`

Parameter	Description

	which is interpreted as:
	`stcmd ci -p "xxx" -fp "C:" *"`
	To avoid a situation like this, escape the final character in "C:\" as follows:
	`stcmd ci -p "xxx" -fp "C:\\" "*"`
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in `C:\orion\`:
	`stcmd ci -p "xxx" -fp "C:\orion" "*"`
	The full syntax is: `-rp "folderName"`.
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-filter	Specifies a string of one or more characters, each of which represents a file status. Never include spaces or other white space in this string. Only files that currently have the specified status(es) will be actioned. Does not apply to files that are `Not In View`. • `C = Current` • `M = Modified` • `O = Out of date` • `N = Not In View` • `I = Missing` • `G = Merge` • `U = Unknown` For example, using `CM` applies a command only to files with a status of `Current` or `Modified`. `-filter` takes precedence over `-f NCI`. If you use `G`, `O`, or `U`, you must also specify `-I` or `-o`. Otherwise the `G`, `O`, or `U` is ignored. `-filter` also takes precedence over `-f NCO`. If you use `G`, `M`, or `U`, you must also specify `-o` to force the checkout operation. Otherwise, the `G`, `M`, or `U` is ignored.
-l | -u | -nel	Locks the item(s). `-l` is exclusive lock, `-u` is unlocked, and `-nel` is non exclusive lock. These items are mutually exclusive and an optional parameter.
-is	Applies the command to all child folders. Without this option, the command applies only to the specified folder. When this option is used with add-folder, you can add an entire branch of folders to the StarTeam folder hierarchy. When used with `add` or `ci`, the command recursively visits all modified files in all sub-folders and checks them in.
-ro	Makes the working file read-only after this operation. Without this option, the file remains as it was prior to the operation. Usually, you use `-ro` to prevent yourself from editing a file that is not locked by you. `-ro` must be used with `-l` or`-u` or `-nel`. If you use `-ro`, you cannot use `-rw`.
-rw	Makes the working file read-write after this operation. Without this option, the file remains as it was prior to the operation. `-rw` must be used with `-l` or `-u` or `-nel`. If you use `-rw`, you cannot use `-ro`.
-vl	Specifies a label (created using `stcmd label`) to be applied to the checked-in files. The label is enclosed in double quotation marks. This option can appear in the command more than once. The label can be either a view or revision label, but it must already exist in the application.

Parameter	Description
-nomove	Do not move labels if already attached.
-f NCI	Specifies the check-in of any file whose status is Modified. *NCI* stands for “needs check-in.” No other types of files are selected for check-in. `-f NCI` is ignored if `-filter` is used.
-o	Forces a check-in/check-out depending on which command is used. `-o` is supported with `-filter` and `-f NCD`, but not with `-f NCO`.
-d | -r	Description or reason for check-in. If `-d` or `-r` is used, `-rf` cannot be used. The reason should be enclosed in double quotation marks and should not exceed 254 characters in length.
-rf	Specifies the file name that contains the check-in reason. This is useful if the same reason should be applied to all check-ins across multiple command line runs.
-active	The active process item.
-cr, -req, -task	Complete path from the project view's root folder to the change request, requirement, or task number to be used as a process item. Use the forward slash (`/`) as a delimiter between folder names. For out-of-view process items, specify the project name and view name in front of the complete folder path. For example: `-cr MyProject/RootView/RootFolder/SourceCode/37` This specifies change request `37` in the `SourceCode` folder (under the root folder) of the `ChildView` view in the `MyProject` project. **Note:** For in-view process items, as long as the change request, requirement, or task numbers are the unique primary descriptors of their types (true by default), it is sufficient simply to specify the number, with no path. The project and view names are assumed from `-p`. If a process item is specified, then the files being checked in are attached to the process item and follow the project process rules. `-cr`, `-req` or `-task` are mutually exclusive. If any one of them is specified, `-filter`/`-f` are ignored.
-mark	Indicates that, if all the files are successfully added, the process item’s status will be changed to fixed (for a change request), finished (for a task), or complete (for a requirement). The files are pinned to the revision with the new status. The item is not marked as fixed, finished, or complete unless all the files are successfully added.
-q	Enables quiet mode. The `-q` option is retained for backward compatibility with the old command line. If `-q` is specified, then `-pf` cannot be specified. The command will return no results.
-pf	Specifies the filter name whose associated filter properties produce the columns in the output matrix. Each command returns a result matrix. `-pf` determines the matrix columns. See `-ofp` for more information. If not specified, the primary descriptor property of the Type is returned as the command output. `-pf` does not apply to the select query command.
-ofp	Provides a file name with a fully qualified path into which to write the command output. By default, a "|" character separates each column in the output. A new line separates each row. The first row is the command name. The second row has the property names. All subsequent rows contain the data. If the file already exists, the output is appended to the end of the file.

Parameter	Description
	It is possible to override the "|" character separator by specifying `separator = fieldSeparator` as a parameter to the connect command. For example, `separator = ;;` specifies two adjacent semicolons (`;`) as the column separator.
-cp	Name of the code page used for localization and internationalization of the content, file and folder names, keyword expansion, etc. Supported code page names are US-ASCII (the default), UTF-8, UTF-16, windows-1252, ISO-8859-1, ISO-8859-9, ISO-8859-15, windows-31j, EUC-JP, Shift_JIS, ISO-2022-JP, x-mswin-936, GB18030, x-EUC-CN, GBK.
files...	Specifies the files to be used in the command by name or by file name-pattern specification, such as `"*.c"`. All options are interpreted using the semantic conventions of UNIX instead of Windows because UNIX conventions are more specific. This means that `"*"`, rather than `"*.*"` means “all files.” The pattern `"*.*"` means “all files with file name extensions.” For example, `“star*.*”` finds `starteam.doc` and `starteam.cpp`, but not `starteam`. To find all of these, you could use `"star*"`. Without this option, the default is `"*"`. When used, this option must always be the last option. Any options after it are ignored. If you use `*`, rather than `"*"` to indicate all files, a UNIX shell expands it into a series of items and passes this series as a group of options to the `stcmd` command. This can cause problems, for example, when you are checking out missing files, so it is best to use `"*"` to avoid unwanted complications. If you use a set of file patterns, each pattern should be enclosed in its own set of quotation marks. For example, you can use `"*.bat" "*.c"`, but you cannot use `"*.bat *.c"`. **Note:** Always enclose this option in quotation marks. Failure to do so can result in unpredictable consequences on all supported platforms. Several special characters can be used in the file specification: ***** Matches any string including the empty string. For example, `*` matches any file name, with or without an extension. "xyz*" will match "xyz" and "xyz.cpp" and "xyzutyfj". **?** Matches any single character. For example, `"a?c"` will match `"abc"` but NOT `"ac"`. **[...]** Matches any one of the characters enclosed by the left and right brackets. **-** A pair of characters separated by a hyphen (`-`) specifies a range of characters to be matched. If the first character following the right bracket (`[`) is an exclamation point (`!`) or a caret (`^`), the rest of the characters are not matched. Any character not enclosed in the brackets is matched. For example, `"x[a-d]y"` matches `"xby"` but not `"xey"`. `"x[!a-d]y"` matches `"xey"` but not `"xby"`. A hyphen (`-`) or right bracket (`]`) may be matched by including it as the first or last character in the bracketed set. To use an asterisk (*), question mark (?), or left bracket (`[`) in a pattern, you must precede it with the escape character (which is the backslash (\).
-pi	The type name is the internal name of a component that supports being a process item type. for example, change request, task, requirement, story, or custom component name.

Parameter	Description
	For a type to be a process item type, it must have a status property for check in. The syntax to use is `-pi typename:"path"`.

Example

The following example uses `ci` to check in `.bmp` files to `Online Help`, a child of the root folder `StarDraw` (in the `StarDraw` view of the `StarDraw` project). The command unlocks the files, makes the working copy read only, and gives the files a revision comment (usually a reason for checking in the files).

Use the `-p` with `ci` or the stateful `set` command to set the context of the project/view/parent folder.

```
stcmd ci -rp "1024/StarDraw/StarDraw/SourceCode/Online Help" -u -ro -r
"revised for beta" "*.bmp"
```

Check Out Files: co

Use `co` to check out files from a StarTeam repository (or vault) to your working folder using the command line. Unless you use `-o`, this command pauses at each file with a `Modified`, `Merge` or `Unknown` status to let you know that the file will not be checked out.

Note: The functionality of the bulk checkout utility (BCO) has been fully added to `co`. BCO is no longer distributed with this version of StarTeam.

Syntax

The syntax for this command is:

```
stcmd{Ex} co [-p ["projectSpecifier"] [-epwdfile |-epwdfile "filename"]
[-cmp] [-encrypt RC4, RC2_ECB, RC2_CBC, RC2_CFB] [-cacheAgentThreads number]
[–useMPXCacheAgent | -useCA host:port | autolocate]] ]
[-cfgl "label" | -cfgp "promotion state" | -cfgd "date"] [-is]
[-pattern "datepattern"] [-rp "directory" | -fp "directory"] [-frp]
[-filter "filter"] [-o] [-e] [-l | -u | -nel] [-break] [-ro | -rw]]
[-vl "name" [-attached] | -vd "date" | -vn number | -vp "name"] [ –chgpkgid
1234567 ]
[-cp "name"] [-exclude <pattern> | #<pattern file>] [-cwf] [-f NCO | NCD]
[-ts] [-eol [ on | off | cr | lf | crlf | platform]] [-fs] [-q | -vb | -pf
"filterName"]
[-ofp "resultsOutputFilePath] [-uv] [-iip] [files...]
```

Parameter	Description
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands. **Note:** If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the

Parameter	Description

	password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.
	
	The full syntax is:
	`stcmd -p "userName:password@hostName:endpoint/projectName/[viewName/][folderHierarchy/]"`
	
	For example:
	`stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/SourceCode/”`
	
	• If the user name is omitted, the current user name is used.
	• If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
	• If the host name is omitted, the default is localhost.
	• Entering an endpoint (port number) is required. The default is 1024.
	• The project name is always required.
	• A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.
	• A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view’s root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` . The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore. **Note:** When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter. In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`. The following is the syntax of the commands that can be used to store an encrypted password.

Parameter	Description

	Use the following syntax to be prompted for the password that will be encrypted and stored in a file.
	`stcmd store-password -epwdfile "filePath"`
	Use the following syntax to include the encrypted password in the command as clear text.
	Note: This action does not access the network with the clear value.
	`stcmd store-password -epwdfile "filePath" -password "password"`
	After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:
	`stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"`
	Important:
	If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:
	`-p user@hostname:port/projectName.viewName -epwdfile "pathToPasswordFile"`

-cmp	Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place. Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data. This is an optional parameter. If not specified, then the platform default is not to compress.
-encrypt	Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks. This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server. The full syntax is: `-encrypt encryptionType`. The types of encryption are: **RC4** RSA RC4 stream cipher (fast). **RC2_ECB** RSA RC2 block cipher (Electronic Codebook). **RC2_CBC** RSA RC2 block cipher (Cipher Block Chaining). These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it. **Note:** For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs.

Parameter	Description
	When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.
-pattern	Qualifies the datetime. It can be specified wherever a date-time is specified, such as `-cfgd`, `-vd`, etc. The pattern must match any valid pattern supported by the java JDK in `java.text.SimpleDateFormat.applyLocalizedPattern(String)`. The pattern may be localized. For every command that takes a `-pattern` parameter, a `-locale` parameter is optionally available. This is the "two character country code".
-cacheAgentThreads	The number of threads allocated to the checkout.
-useMPXCacheAgent	Ir true, and if a cache agent is available, then use it.
useCA	Provides an address. `host:port` specifies the actual known address of the cache agent. `useCA` either can be used in the context of a known cache agent address port or turns on `autolocate`.
autlocate	StarTeam automatically gets the cache agent.
-cfgl	Configures the view using the specified label. Without `-cfgl`, `-cfgp`, or `-cfgd`, the view's current configuration is used.
-cfgp	Configures the view using the specified promotion state.
-cfgd	Configures the view as of the specified date/time. Examples include: "12/29/13 10:52 AM" "December 29, 2013 10:52:00 AM PST" "Monday, December 29, 2013 10:52:00 AM PST"
-is	Applies the command to all child folders. Without this option, the command applies only to the specified folder. When this option is used with add-folder, you can add an entire branch of folders to the StarTeam folder hierarchy. When used with `add` or `ci`, the command recursively visits all modified files in all sub-folders and checks them in.
-rp	Overrides the working folder or working directory for the StarTeam view's root folder. While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways. While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project.

Parameter	Description

	The UNIX shell interprets a backslash (\) as an escape character when it precedes certain characters, such as quotation marks. As a result, an error occurs in the following example:
	`stcmd ci -p "xxx" -rp "C:\" "*"`
	which is interpreted as:
	`stcmd ci -p "xxx" -rp "C:" *"`
	To avoid a situation like this, escape the final character in "`C:\`" as follows:
	`stcmd ci -p "xxx" -rp "C:\\" "*"`
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in "`C:\orion\`":
	`stcmd ci -p "xxx" -rp "C:\orion" "*"`
	The full syntax is: `-rp "folderName"` .
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-fp	Overrides the specified StarTeam folder's working folder or working directory. This is equivalent to setting an alternate working path for the folder.
	While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways.
	While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project.
	A backslash (\) is interpreted as an escape character when it precedes quotation marks. As a result, an error occurs in the following example:
	`stcmd ci -p "xxx" -fp "C:\" "*"`
	which is interpreted as:
	`stcmd ci -p "xxx" -fp "C:" *"`
	To avoid a situation like this, escape the final character in "C:\" as follows:
	`stcmd ci -p "xxx" -fp "C:\\" "*"`
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in `C:\orion\`:
	`stcmd ci -p "xxx" -fp "C:\orion" "*"`
	The full syntax is: `-rp "folderName"`.
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-frp	Forces the specified relative path. When used, the `-frp` parameter ensures that the entire folder tree is successfully checked out relative to the root folder, either with the default or via the `-rp` root path override.

Parameter	Description
	For example, if the default root folder path is `c:\stardraw`, then the command `co -p "Administrator:Adiministrator@localhost:49201"/StarDraw -is -frp` checks out the entire folder tree recursively to `c:\stardraw`. If the path is overridden using the `-rp` command, such as `co -p "Administrator:Adiministrator@localhost:49201"/StarDraw -is -rp "c:\temp" -frp`, the entire stardraw folder tree gets checked out to `c:\temp`. `-frp` does nothing if the entire folder tree is already relative to the root folder. However, consider the StarDraw example. If the sub-folder `Source Code` default path is set to a mapped drive on a machine, for example, `e:\\StarDraw` or a UNC path (`\\MicroFocus Build Server\`) and you run the command line on a different machine from where the mapped drive or the UNC path is unreachable, the `co` command without `-frp` will throw an exception. With `-frp`, the command will succeed, the `Source Code` folder and all descendant sub-folders are created relative to its parent, and the files in the folder hierarchy are checked out.
-filter	Specifies a string of one or more characters, each of which represents a file status. Never include spaces or other white space in this string. Only files that currently have the specified status(es) will be actioned. Does not apply to files that are `Not In View`. • `C = Current` • `M = Modified` • `O = Out of date` • `N = Not In View` • `I = Missing` • `G = Merge` • `U = Unknown` For example, using `CM` applies a command only to files with a status of `Current` or `Modified`. `-filter` takes precedence over `-f NCI`. If you use `G`, `O`, or `U`, you must also specify `-I` or `-o`. Otherwise the `G`, `O`, or `U` is ignored. `-filter` also takes precedence over `-f NCO`. If you use `G`, `M`, or `U`, you must also specify `-o` to force the checkout operation. Otherwise, the `G`, `M`, or `U` is ignored.
-o	Forces a check-in/check-out depending on which command is used. `-o` is supported with `-filter` and `-f NCD`, but not with `-f NCO`.
-e	If specified, `-e` throws an exception if `-filter` includes `M`, `G`, or `U` and any of the identified file statuses match `Merge`, `Modified`, or `Unknown`. The thrown exception will prevent all other files from being checked out as well.
-l | -u | -nel	Locks the item(s). `-l` is exclusive lock, `-u` is unlocked, and `-nel` is non exclusive lock. These items are mutually exclusive and an optional parameter.
-break	Breaks the current lock by another user if you have the access rights to break locks.

Parameter	Description
-ro	Makes the working file read-only after this operation. Without this option, the file remains as it was prior to the operation. Usually, you use `-ro` to prevent yourself from editing a file that is not locked by you. `-ro` must be used with `-l` or`-u` or `-nel`. If you use `-ro`, you cannot use `-rw`.
-rw	Makes the working file read-write after this operation. Without this option, the file remains as it was prior to the operation. `-rw` must be used with `-l` or `-u` or `-nel`. If you use `-rw`, you cannot use `-ro`.
-vl	Specifies a label (created using `stcmd label`) to be applied to the checked-in files. The label is enclosed in double quotation marks. This option can appear in the command more than once. The label can be either a view or revision label, but it must already exist in the application.
-attached	May be specified, but only in conjunction with `-vl "label name".` When `-vl` is specified, and if `-attached` is also specified, then only those files which are attached to the label (identified by `-vl`) will be checked out, and specifically at the tip revision. If `-attached` is not specified, its default value is always false. `-attached` used in conjunction with `-vl "label name"` alters the fundamental behavior of the `co` command. If `-attached` is false (the default), the files to be checked out are identified by the file(s) pattern specified by the co command, `-vl` specifies the revisions of the subset of files attached to the label.
-vd	Specifies the as-of date/time used to identify the revisions to be checked out. The last revision before the specified date/time is the one checked out for each file. See the date/time examples for `-cfgd`.
-vn	Specifies the revision number
-vp	Specifies the promotion state.
-chgpkgid	If specified, the checkout is based on a committed change package, with the specified view member ID. All files attached to that change package are checked out at the revisions they were at when the change package was committed. A change package checkout is a targeted sub-set of files checked out from history, at a revision timestamp equal to the change package commit time.
-cp	Name of the code page used for localization and internationalization of the content, file and folder names, keyword expansion, etc. Supported code page names are US-ASCII (the default), UTF-8, UTF-16, windows-1252, ISO-8859-1, ISO-8859-9, ISO-8859-15, windows-31j, EUC-JP, Shift_JIS, ISO-2022-JP, x-mswin-936, GB18030, x-EUC-CN, GBK.
-exclude <pattern> | #<patternFile>	Exclude files or folders whose names match a pattern (or set of patterns). You can either specify the pattern inline `-exclude <pattern>` or you can specify a set of patterns in a file `-exclude#patternFile`. A pattern can be an exact file or folder name or it may contain wildcard characters (e.g., `'*.class'`). To specify a folder name, precede the pattern name with a forward-slash (e.g., `'/bin'`). A single pattern can be provided with `-exclude`. Alternatively, one or

Parameter	Description
	more patterns can be specified on separate lines of the given `<pattern file>` (prefixed with #). Pattern file names may be fully qualified with their path on the file system, e.g. `#"c:\temp\patternfile.txt"` or relative to the current folder e.g. `#"patternfile.txt"`. In all cases, the # symbol precedes the double quotes and pattern file names must be enclosed in double quotation marks `"…"`. In all cases, you **must** have a space between the # and the pattern file path. For example: `-exclude # "c:/temp/exlcudefile.txt"` If the pattern matches a folder path, then all files in that folder path will be excluded. Finally, a pattern may also be a fully or partially qualified path to a file in StarTeam without wildcards, e.g. `/StarDraw/External Resources/StarDraw.ico` If the pattern matches an exact file name, then all instance of that file name, no matter where they are in the folder tree, will be excluded. In this case, only the file that exactly matches the parent folder path will be excluded. **Note:** To exclude a folder tree containing files and sub-folders, append `/*` to the 'root' of the tree. For example, if your folder tree is `/StarDraw/External resource/Documentation`, and you want to exclude the tree rooted at '`External resource`', specify `/External resource/*` in the pattern file. **Note:** When using `stcmd.exe`, always encase the entire pattern `#patternFile` in double quotes. For example, `"#c:/folder/folder too/file"` When using `stcmdEx.jar`, encase just the `patternFile` in double quotes. For example, `#"c:/folder/folder too/file"` This syntactic difference arises from the way special characters (such as #) are handled in C/C++ as opposed to Java, in the context of the Windows Operating System.
-cwf	Create working folders from StarTeam folders, even if they do not contain any files. **Note:** `-cwf` will only create the working folder for the specified folder. Use `-is` with `-cwf` to create working folders for all child folders.
-f NCO	Specifies the check-out of any file whose status is `Missing` or `Out of Date`. NCO stands for “needs check-out.” No other files are selected for check-out. `-f NCO` is ignored if `-filter` is used. **Note:** `-f NCO` and `-f NCD` are mutually exclusive.
-f NCD	Specifies the check-out of any file whose status is `Missing` or `Out of Date` and the deletion of all not-in-view files in the workspace. NCD stands for "needs check-out and delete". `-f NCD` is ignored if `-filter` is used.

Parameter	Description
	Note: `-f NCD` and `-f NCO` are mutually exclusive.
-ts	Sets each working file's time stamp to the check-out time. Without this option, the file is given the same time stamp as the checked-in revision of the file.
-fts	Sets each working folder time stamp to the check-out time. Without this option, the folder is given the same time stamp as the checked-in revision of the folder.
-iip	Ignores invalid path. If this parameter is specified, and if the folder path in the command is specified and invalid, then the command will exit successfully without throwing an exception.
-eol	Automatically convert end-of-line markers. Use `[cr|lf|crlf|off|platform]`. When on, text files are transferred from the StarTeam Server's repository to the workstation's working folder with the end-of-line convention for the platform executing the command as determined by the Java VM. When off, the default, no end-of-line conversion is performed. Using off is the same as not using `-eol` at all. For Microsoft Windows clients, the end-of-line marker is a carriage return/line feed (`crlf`) combination. For UNIX platforms, it is a line feed (`lf`). For MAC systems, a carriage return (`cr`). You would set this option to on or `lf`, for example, when you compare a file from the repository and a working file on a UNIX system (if the repository stores text files as `crlf`). **Note:** If a file has a fixed `EOL` value set in StarTeam, then `cr`, `lf` and `crlf` are all ignored, and the file is always checked out using the set (fixed) `eol` value. To override this behavior, pick `-eol platform`. **Note:** When `-eol platform` is selected, then all files are checked out using the platform specific `eol`, whether or not they are marked fixed for a different platform.
-fs	Prevents file statuses from being remembered after the check-out occurs. Subsequent status values for these files will be incorrect and indeterminate. Use this option where a file's status is irrelevant. For example, if you routinely delete the working folders before checking out files for a build, there are no files and their statuses do not matter. Additionally, with per folder status repository in use on the local machine, if files are checked out to empty working folders, empty `.sbas` folders will not be left on disk. Be aware that the file statuses may never be known, even if you use the `update-status` command later. You can do a force check out without the `-fs` option to obtain current files with correct statuses.
-q	Enables quiet mode. The `-q` option is retained for backward compatibility with the old command line. If `-q` is specified, then `-pf` cannot be specified. The command will return no results.
-vb	Verbose mode kept for backward compatibility with bulk check out. If `-vb` is specified, the list of all files considered for checkout is returned with the per file number of bytes checked out, the time taken for the checkout, and whether a specified cache agent could be leveraged to check the content out. In addition, if

Parameter	Description
	the file was not current prior to the checkout, the file status on disk is also recorded.
-pf	Specifies the filter name whose associated filter properties produce the columns in the output matrix. Each command returns a result matrix. `-pf` determines the matrix columns. See `-ofp` for more information. If not specified, the primary descriptor property of the Type is returned as the command output. `-pf` does not apply to the select query command.
-ofp	Provides a file name with a fully qualified path into which to write the command output. By default, a "`|`" character separates each column in the output. A new line separates each row. The first row is the command name. The second row has the property names. All subsequent rows contain the data. If the file already exists, the output is appended to the end of the file. It is possible to override the "`|`" character separator by specifying `separator = fieldSeparator` as a parameter to the connect command. For example, `separator = ;;` specifies two adjacent semicolons (`;`) as the column separator.
-uv	If this parameter is specified, then only user-visible folders on the local client machine (set via the StarTeam Cross-Platform Client) will be checked out.
files...	Specifies the files to be used in the command by name or by file name-pattern specification, such as "`*.c`". All options are interpreted using the semantic conventions of UNIX instead of Windows because UNIX conventions are more specific. This means that "`*`", rather than "`*.*`" means “all files.” The pattern "`*.*`" means “all files with file name extensions.” For example, “`star*.*`” finds `starteam.doc` and `starteam.cpp`, but not `starteam`. To find all of these, you could use "`star*`". Without this option, the default is "`*`". When used, this option must always be the last option. Any options after it are ignored. If you use `*`, rather than "`*`" to indicate all files, a UNIX shell expands it into a series of items and passes this series as a group of options to the `stcmd` command. This can cause problems, for example, when you are checking out missing files, so it is best to use "`*`" to avoid unwanted complications. If you use a set of file patterns, each pattern should be enclosed in its own set of quotation marks. For example, you can use "`*.bat" "*.c`", but you cannot use "`*.bat *.c`".  **Note:** Always enclose this option in quotation marks. Failure to do so can result in unpredictable consequences on all supported platforms. Several special characters can be used in the file specification: ***** Matches any string including the empty string. For example, `*` matches any file name, with or without an extension. "xyz*" will match "xyz" and "xyz.cpp" and "xyzutyfj". **?** Matches any single character. For example, "`a?c`" will match "`abc`" but NOT "`ac`". **[...]** Matches any one of the characters enclosed by the left and right brackets.

Parameter	Description
	- A pair of characters separated by a hyphen (-) specifies a range of characters to be matched. If the first character following the right bracket ([ ) is an exclamation point (!) or a caret (^), the rest of the characters are not matched. Any character not enclosed in the brackets is matched. For example, "x[a-d]y" matches "xby" but not "xey". "x[!a-d]y" matches "xey" but not "xby". A hyphen (-) or right bracket ( ]) may be matched by including it as the first or last character in the bracketed set. To use an asterisk (*), question mark (?), or left bracket ([ ) in a pattern, you must precede it with the escape character (which is the backslash (\).

Examples

The following example uses co to lock and check out .doc files from User Manual, a child of the root folder StarDraw (in the StarDraw view of the StarDraw project):

```
stcmd co -p "JMarsh:password@Orion:1024/StarDraw/StarDraw/User Manual" -l
"*.doc"
```

The next example uses co to merge a readme file:

```
stcmd co -p "NTesla:@10.50.5.179:49201/WebDev/WebDev" -encrypt RC4 -fp  "/
export/home0/johnson/working" -merge "README"
```

Either use the -p with co (as above) or the stateful connect and set commands (below) to set the context of the project/view/parent folder:

```
stmcd connect JMarsh:password@Orion:1024
stcmd set project = StarDraw view = StarDraw folderHierarchy = " StarDraw/
User Manual"
stcmd co -l "*.doc"
stcmd disconnect
```

stcmd supports both stateless (using -p) and stateful (connect…, set…, {commands}, disconnect) models.

The stateless approach causes each command to connect, set the project, execute the command, and then disconnect.

The stateful approach requires the script author to manage the connect, set and disconnect. However, this has the advantage of supporting multiple commands for execution within the context of a given connect, set, and disconnect session.

The next example uses stcmd co to checkout all files, recurse through the entire folder tree (-is) and return (for the set of checked out files) the set of all property values described by the property filter -pf:

```
stcmd co -p " JMarsh:password@Orion:1024 /StarDraw" -is -pf \"<All Files By
Status>\"
```

This example checks out files from a historical point in time, rolled back to a view configuration based on the label label abc:

```
stcmd co -p " JMarsh:password@Orion:1024 /StarDraw" -cfgl "label abc" -is -pf
\"<All Files By Status>\"
```

Compare File Revisions: diff

Use `diff` to display differences between two revisions of a file. The command can be applied to more than one file. If you do not specify any revisions using `-vn`, `-vd`, `-vl`, or `-vp`, the working copy of each specified file is compared to the tip revision in the repository (or vault) for this file. If you specify a single revision, the working copy of each specified file is compared to that revision. If you specify two revisions, those two revisions of each specified file are compared.

When comparing text files, the differences can be displayed. When comparing binary files, output results indicate whether the revisions of the file are the same or different.

Syntax

The syntax for this command is:

```
stcmd{Ex} diff [ -p "projectSpecifier" [-epwdfile "filePath"] [-cmp] [-csf]
[-encrypt encryptionType] ][-cfgl "labelName"| -cfgp "stateName" | -cfgd
"asOfDate"]
[-is] [-rp "folderPath" | -fp "folderPath"] [-filter "fileStatus"]
[-eol [on | off | cr| lf| crlf]] [-w | -Bpvcs | -b] [-I] [-m "maskSet"]
[-t number] [-c number] [-n] [-nd] [-vl "labelName" | -vd "asOfDate" |
-vn revisionNumber] [-pattern "date-pattern"] | -vp
promotionStateName[files...]
```

Parameter	Description
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.

Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.

Parameter	Description
	• The project name is always required. • A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date. • A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view's root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .
	
	The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.
	
	Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.
	
	In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.
	
	The following is the syntax of the commands that can be used to store an encrypted password.
	
	Use the following syntax to be prompted for the password that will be encrypted and stored in a file.
	`stcmd store-password -epwdfile "filePath"`
	
	Use the following syntax to include the encrypted password in the command as clear text.
	
	Note: This action does not access the network with the clear value.
	
	`stcmd store-password -epwdfile "filePath" -password "password"`
	
	After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:
	`stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"`
	
	Important:
	
	If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:
	`-p user@hostname:port/projectName.viewName -epwdfile "pathToPasswordFile"`

Parameter	Description
-cmp	Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place. Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data. This is an optional parameter. If not specified, then the platform default is not to compress.
-csf	When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder. The default is that StarTeam folders are not differentiated based on the case of letters in their names. With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.
-encrypt	Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks. This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server. The full syntax is: `-encrypt encryptionType`. The types of encryption are: **RC4** RSA RC4 stream cipher (fast). **RC2_ECB** RSA RC2 block cipher (Electronic Codebook). **RC2_CBC** RSA RC2 block cipher (Cipher Block Chaining). These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it. **Note:** For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.
-cfgl	Configures the view using the specified label. Without `-cfgl`, `-cfgp`, or `-cfgd`, the view's current configuration is used.
-cfgp	Configures the view using the specified promotion state.
-cfgd	Configures the view as of the specified date/time. Examples include: "12/29/13 10:52 AM"

Parameter	Description
	"December 29, 2013 10:52:00 AM PST"
	"Monday, December 29, 2013 10:52:00 AM PST"
-is	Applies the command to all child folders. Without this option, the command applies only to the specified folder. When this option is used with add-folder, you can add an entire branch of folders to the StarTeam folder hierarchy. When used with `add` or `ci`, the command recursively visits all modified files in all sub-folders and checks them in.

-rp	Overrides the working folder or working directory for the StarTeam view's root folder.
	While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways.
	While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project.
	The UNIX shell interprets a backslash (\) as an escape character when it precedes certain characters, such as quotation marks. As a result, an error occurs in the following example:
	`stcmd ci -p "xxx" -rp "C:\" "*"`
	which is interpreted as:
	`stcmd ci -p "xxx" -rp "C:" *"`
	To avoid a situation like this, escape the final character in "`C:\`" as follows:
	`stcmd ci -p "xxx" -rp "C:\\" "*"`
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in "`C:\orion\`":
	`stcmd ci -p "xxx" -rp "C:\orion" "*"`
	The full syntax is: `-rp "folderName"` .
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-fp	Overrides the specified StarTeam folder's working folder or working directory. This is equivalent to setting an alternate working path for the folder.
	While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways.
	While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project.
	A backslash (\) is interpreted as an escape character when it precedes quotation marks. As a result, an error occurs in the following example:
	`stcmd ci -p "xxx" -fp "C:\" "*"`
	which is interpreted as:
	`stcmd ci -p "xxx" -fp "C:" *"`

Parameter	Description

	To avoid a situation like this, escape the final character in "C:\" as follows:
	`stcmd ci -p "xxx" -fp "C:\\" "*"`
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in `C:\orion\`:
	`stcmd ci -p "xxx" -fp "C:\orion" "*"`
	The full syntax is: `-rp "folderName"`.
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-filter	Specifies a string of one or more characters, each of which represents a file status. Never include spaces or other white space in this string. Only files that currently have the specified status(es) will be actioned. Does not apply to files that are `Not In View`. • `C = Current` • `M = Modified` • `O = Out of date` • `N = Not In View` • `I = Missing` • `G = Merge` • `U = Unknown` For example, using `CM` applies a command only to files with a status of `Current` or `Modified`. `-filter` takes precedence over `-f NCI`. If you use `G`, `O`, or `U`, you must also specify `-I` or `-o`. Otherwise the `G`, `O`, or `U` is ignored. `-filter` also takes precedence over `-f NCO`. If you use `G`, `M`, or `U`, you must also specify `-o` to force the checkout operation. Otherwise, the `G`, `M`, or `U` is ignored.
-eol	Automatically convert end-of-line markers. Use `[cr|lf|crlf|off|platform]`. When on, text files are transferred from the StarTeam Server's repository to the workstation's working folder with the end-of-line convention for the platform executing the command as determined by the Java VM. When off, the default, no end-of-line conversion is performed. Using off is the same as not using `-eol` at all. For Microsoft Windows clients, the end-of-line marker is a carriage return/line feed (`crlf`) combination. For UNIX platforms, it is a line feed (`lf`). For MAC systems, a carriage return (`cr`). You would set this option to on or `lf`, for example, when you compare a file from the repository and a working file on a UNIX system (if the repository stores text files as `crlf`). **Note:** If a file has a fixed `EOL` value set in StarTeam, then `cr`, `lf` and `crlf` are all ignored, and the file is always checked out using the set (fixed) `eol` value. To override this behavior, pick `-eol platform`. **Note:** When `-eol platform` is selected, then all files are checked out using the platform specific `eol`, whether or not they are marked fixed for a different platform.

-w	Ignores all whitespace (tabs and spaces) when comparing two lines in text files. For example, the following lines would be equivalent:
	`"a = ( b + 2);"`
	`"a=(b+2);"`

Parameter	Description
	The `-w`, `-Bpvcs`, and `-b` options are mutually exclusive.

-Bpvcs	When comparing two lines of text files, ignores leading and trailing whitespace. For example, the following lines are equivalent because there is only one space between "hi" and "mom":
	`" hi mom "`
	`" hi mom"`
	but the next line is not equivalent:
	`"hi mom"`

-b	When comparing two lines of text files, ignores trailing whitespace and treats all other strings of whitespace as equal in length. For example, the following lines are equivalent:
	`" hi mom "`
	`" hi mom"`

-I	Ignores the case of letters when comparing two text files. For example, "A" is equivalent to "a".

-m	When comparing two text files, ignores the characters in certain columns as specified by one or more masks. Each mask has the following syntax:
	`"columnNumber-columnNumber[(numeric)]"`
	For example, "1-6" ignores the characters in the first six columns of each line, and "1-6 (numeric)" ignores the first six columns of each line if the character in column 1 is a digit in both files.
	You can use a series of masks, but they must be separated by commas. The syntax is:
	`"mask[,mask]..."`

-t	Specifies the number of spaces to use for each tab stop when displaying the file differences for text files. The default is four. Use `-t 0` to suppress tab conversion.
-c	Specifies the number of unchanged lines to display before and after a difference is found in text files. Without this option, all lines of the files are displayed. For example, `-c 2` places two unchanged lines before and after each line or set of lines that has changed.
-n	Suppresses the display of line numbers in the two text files.
-nd	Suppresses the display of differences in two text files. Comparisons of binary files do not display differences.
-vl	Specifies a label (created using `stcmd label`) to be applied to the checked-in files. The label is enclosed in double quotation marks. This option can appear in the command more than once. The label can be either a view or revision label, but it must already exist in the application.
-vd	Specifies the as-of date/time used to identify the revisions to be checked out. The last revision before the specified date/time is the one checked out for each file. See the date/time examples for `-cfgd`.
-vn	Specifies the revision number
-pattern	Qualifies the datetime. It can be specified wherever a date-time is specified, such as `-cfgd`, `-vd`, etc. The pattern must match any valid pattern supported by the java JDK in `java.text.SimpleDateFormat.applyLocalizedPattern(String)`. The pattern may be localized.

Parameter	Description
	For every command that takes a `-pattern` parameter, a `-locale` parameter is optionally available. This is the "two character country code".
-vp	Specifies the promotion state.

Example

The following example uses `diff` to compare the `Beta1` and `Beta2` revisions of each of the `.cpp` files in the folder `SourceCode`, a child of the root folder `StarDraw` (in the `StarDraw` view of the `StarDraw` project). It ignores all white space.

Use the `-p` with `diff` or the stateful `set` command to set the context of the project/view/parent folder.

```
stcmd diff "SourceCode" -w -vl "Beta1" -vl "Beta2" "*.cpp"
```

Note that `diff` compares versions of files at differing specified revisions of the view, which allows you flexibility in determining how to specify the revisions of interest. A few examples are listed below.

Compare `.cpp` files in the view between labels `Beta1` and `Beta2`:

```
stcmd diff "SourceCode" -w -vl "Beta1" -vl "Beta2" "*.cpp"
```

Compare `.java` files in the view between dates `March 01 1997` to `Jan 01 2013`:

```
stcmd diff "SourceCode" -w -pattern MM/DD/yyyy -vd 03/01/1997 -vd 01/01/2013
"*.java"
```

Compare `.java` files in the view between date `March 01 1997` and the label `Beta2`:

```
stcmd diff "SourceCode" -w -pattern MM/DD/yyyy -vd 03/01/1997 -vl "Beta2"
"*.java"
```

Compare `.cs` files in the view between date `December 31 2013` and the tip:

```
stcmd diff "SourceCode" -w -pattern MM/DD/yyyy -vd 12/31/2013 "*.cs"
```

Also, the best use of `-vd` is in conjunction with the `-pattern` option. The `-pattern` specifications is part of the java `SimpleDateFormat` and permits the engine to precisely determine what one has in mind when specifying a date without attempting to guess at intent.

Connect: connect

Use `connect` to connect to the StarTeam Server. The connection persists until you use the `disconnect` command. Because the connection persists, you can specify all subsequent commands without using the `-p` option to connect with each command. However, to switch between projects, views, and working folders, use `set project...[viewHierarchy ] [folderHierarchy ]` while in the same session.

Note: This command is part of the stateful model and only works with `stcmd`, not `stcmdEx`.

Before attempting to connect to the same session, you must disconnect. Use the `disconnect` command to disconnect from the session.

Syntax

The syntax for this command is as follows:

```
stcmd connect [username[:password]@]address:port
[passwordFromFile passwordFile] [storePasswordToFile passwordFile]
[caseSensitiveFolders | -csf] [encryption = RC4|RC2_ECB|RC2_CBC_RC2_FCB]
[compression | -cmp] [mpx=on|off (default ON)
[profile=eventHandlerProfilename]
[cacheAgent@address:port (default autoLocate] | =off]
```

```
[[cacheAgentThreads=noOfThreads][-mode [lock | exlock | unlock]]
[separator=fieldSeparator] [headers = on|off]
```

Parameter	Description
passwordFromFile	Location of file containing password.
storePasswordToFile	Location to store the password file.
-csf	When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder. The default is that StarTeam folders are not differentiated based on the case of letters in their names. With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.
-encrypt	Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks. This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server. The full syntax is: `-encrypt encryptionType`. The types of encryption are: **RC4** RSA RC4 stream cipher (fast). **RC2_ECB** RSA RC2 block cipher (Electronic Codebook). **RC2_CBC** RSA RC2 block cipher (Cipher Block Chaining). These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.  **Note:** For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.
-cmp	Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Parameter	Description
	Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data. This is an optional parameter. If not specified, then the platform default is not to compress.
mpx	Turn MPX on or off. Default value is `on`.
profile	You can specify an MPX profile by `eventhandler` name, if your server offers you the ability to connect to more than one MPX server that allows you to choose between event handlers.
cacheAgent@address:port	Provides an address. `host:port` specifies the actual known address of the cache agent.
autoLocate	StarTeam automatically gets the cache agent.
cacheAgentThreads	The number of threads allocated to the cacheAgent.
-mode	Indicates whether the server is to be locked, exclusively locked, or unlocked. **-mode lock** Only server administration commands are accepted until the server is unlocked. For example, you might use this command while running a backup program. **-mode exlock** Only you can access the server until it is unlocked. For example, you might do this when creating a custom field. **-mode unlock** Use to make the server available to users again.
lock	Non-exclusively locks the StarTeam Server. Only administrative commands can be performed.
unlock	Unlocks the StarTeam Server so that anyone with the appropriate access rights can access it.
exlock	Exclusively locks the server so that no one else can access it.
headers	By default, `headers = on`. If `headers = off`, output files are written without headers.

Example

The following example uses `connect` to connect to the server using port 1024 on Orion and non-exclusively locks the server.

```
connect "JMarsh:password@Orion:1024" -mode lock
```

Create Labels: label

Use `label` to create or update a view or revision label. A view label can be designated as a build label. By default, view labels are automatically applied to every folder, file, change request, requirement, topic, and task in the view. By default, revision labels are not applied to any items.

You can use `apply-label` to apply labels created with `label` to specified files. You can also use the label option (`-vl`) in `ci` to attach your new label to files as you check them in.

Syntax

The syntax for this command is:

```
stcmd{Ex} label -p "projectSpecifier" -nl "labelName" [-vl "labelName" |
-vd "asOfDate" | -vp stateName] [-d "description"] [-b | -r] [-f] [-u]
[–pattern “date-pattern”] [-ps promotionStateName ]
```

Parameter	Description

-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.
	  **Note:** If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.
	The full syntax is:
	`stcmd -p "userName:password@hostName:endpoint/projectName/ [viewName/][folderHierarchy/]"`
	For example:
	`stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/ SourceCode/”`
	• If the user name is omitted, the current user name is used.
	• If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
	• If the host name is omitted, the default is localhost.
	• Entering an endpoint (port number) is required. The default is 1024.
	• The project name is always required.
	• A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.
	• A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view’s root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-nl	Specifies the new label’s name. If not found, a new label will be created. If found, the existing label description will be updated and the label will be marked frozen or unlocked.

Parameter	Description
-vl	Specifies a label (created using `stcmd label`) to be applied to the checked-in files. The label is enclosed in double quotation marks. This option can appear in the command more than once. The label can be either a view or revision label, but it must already exist in the application.
-vd	Specifies the as-of date/time used to identify the revisions to be checked out. The last revision before the specified date/time is the one checked out for each file. See the date/time examples for `-cfgd`.
-vp	Specifies the promotion state.
-d	A user specified Description. However, we continue to support `-r` as an alternate to `-d` for the description, but strictly for backward compatibility
-b	Specifies that the new label is a build label. Without either`-b` or `-r`, the label is a view label. View labels (and a build label is a special type of view label) are immediately and automatically applied to every folder, file, change request, task, and topic in the view.
-r	Specifies that the new label is a revision label. You can use the new label to label files that you check in. This command does not attach the new label to any items unless you create the label by copying an existing revision label that is attached to one or more items. See the `-vl` option.
-f	Creates the new label as a frozen label or updates an existing label and marks it as a frozen label.
-u	Creates the new label as an unlocked label or updates an existing frozen label and marks it unlocked.
-pattern	Qualifies the datetime. It can be specified wherever a date-time is specified, such as `-cfgd`, `-vd`, etc. The pattern must match any valid pattern supported by the java JDK in `java.text.SimpleDateFormat.applyLocalizedPattern(String)`. The pattern may be localized. For every command that takes a `-pattern` parameter, a `-locale` parameter is optionally available. This is the "two character country code".
-ps	When `-ps` is specified, the label is assigned to the promotion state specified by -ps. If the label name does not exist, then a new label is created following existing rules. If the label name already exists, then it is simply assigned to the promotion state. **Note:** If none of `-vd`, `-vl` or `-vp` are specified, then the created label is based on the current server time.

Example

The following example uses label to create a new build label named `Beta` for the `StarDraw` view of the `StarDraw` project.

Use the `-p` with `label` or the stateful `set` command to set the context of the project/view/parent folder.

```
stcmd label -nl "Beta" -b
```

Delete: delete

Use `delete` to delete all objects that satisfy the `WHERE` clause. The syntax of the where clause is identical to `select`, `update` or `delete`, and is fully described in the `select` command section. A check-in change package is created that records the set of items that were deleted. The items are deleted in a server transaction. As a result, either all items are deleted (and the transaction succeeds) or not a single item is

deleted and the transaction fails. If the transaction fails, it is rolled back. Values that contain spaces should be enclosed in double quotes. This command has been modeled on the standard SQL `DELETE` syntax.

Syntax

The syntax for this command is:

```
stcmd{Ex} delete  type {local} {output* | {propertyName,...} |
filter='myFilter' into "outputFilePath" {separator 'fieldSeparator'}}
where {{ attached-label = 'labelName' } | { query = 'myquery' } |
propertyName relation value and/or propertyName relation value and/or...}
{for} {folder = 'myfolder' {recurse} or folder = 'myfolderhierarchy'
{recurse}
or folder = . {recurse}} or ...}
type { File | Folder | ChangeRequest | Requirement | Task | Topic |
CustomComponentTypeName | Trace | ChangePackage }
[–epwdfile “passwordfilepat”]
[-p "userName:password@hostName:endpoint/projectName/[viewName/]
[folderHierarchy/]"]
```

Parameter	Description
type	Specifies the StarTeam item type by name. Types are mutually exclusive.
myFilter	Specifies a filter by name, whose properties are written to the output file.
myFolder	Specifies the StarTeam folder name in the current view. If there are multiple folders with the same name, the command performs the action on all folders with that name.
myFolderHierarchy	Specifies the folder hierarchy in the "`/`" format. Start from the root folder and end in a branch folder. For example: `/StarDraw/SourceCode/On-line Help/`.
output	Turns on logging of the command to a log file specified by `INTO`. The `INSERT`, `DELETE`, and `UPDATE` commands log the selected properties of the inserted items to a log file. The property values are separated by the specified `fieldSeparator`, or "`|`" if a separator is not specified.
recurse	Designates all descendants from the folder specified.
.	Implies the current working folder, requiring the tool to find StarTeam folders with paths mapping to the current working folder. The Command processor must be running inside the StarTeam folder hierarchy.
query='myQuery'	Specifies the saved StarTeam query name for the type. It acts as the equivalent of a compound where clause of a SQL statement, such as combinations of relations and operators. If no query name is specified, the command performs the action on all objects of the type.
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new

Parameter	Description

command line syntax.) Stateless commands cause more client server traffic than stateful commands.

Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.
- A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.
- A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view’s root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile

The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .

The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.

Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.

Parameter	Description
	In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.
	The following is the syntax of the commands that can be used to store an encrypted password.
	Use the following syntax to be prompted for the password that will be encrypted and stored in a file.

```
stcmd store-password -epwdfile "filePath"
```

Use the following syntax to include the encrypted password in the command as clear text.

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password
"password"
```

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

Example

The following example deletes all change requests with an open status from the `StarDraw` project `StarDraw` view.

```
stcmd connect Administrator:Administrator@localhost:49201
stcmd set project = ‘StarDraw’ view = ‘StarDraw’
stcmd delete changerequest where query = "Status = Open" disconnect
```

Delete Local Files: delete-local

Use `delete-local` to delete files from a working folder and the working folder itself, if empty of files, resulting from executing this command . You can delete files that are under version control, as well as files that are not in StarTeam. This action does not remove any files from version control. It merely reduces the amount of data stored on your workstation in a working folder. If you are deleting files based on their StarTeam status, it is a good idea to use `update-status` first.

Syntax

The syntax for this command is:

```
stcmd{Ex} delete-local [-p "projectSpecifier" [-epwdfile "filePath"]
[-cmp] [-csf] [-encrypt encryptionType] ] [-is] [-nivf] [-rp "folderPath" |
-fp "folderPath"] [-filter "fileStatus"] [-cfgl "labelName" |
-cfgp "stateName" | -cfgd "asOfDate"] [-q|-pf “filterName”]
[-ofp “resultsOutputFilePath”]  [files...]
```

Parameter	Description

-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.
	
	Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.
	
	The full syntax is:
	`stcmd -p "userName:password@hostName:endpoint/projectName/ [viewName/][folderHierarchy/]"`
	
	For example:
	`stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/ SourceCode/”`
	
	• If the user name is omitted, the current user name is used.
	• If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
	• If the host name is omitted, the default is localhost.
	• Entering an endpoint (port number) is required. The default is 1024.
	• The project name is always required.
	• A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.
	• A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view’s root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` . The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.

Parameter	Description

Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.

In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.

The following is the syntax of the commands that can be used to store an encrypted password.

Use the following syntax to be prompted for the password that will be encrypted and stored in a file.

```
stcmd store-password -epwdfile "filePath"
```

Use the following syntax to include the encrypted password in the command as clear text.

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password "password"
```

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile` "`filePath`" as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

This is an optional parameter. If not specified, then the platform default is not to compress.

-csf

When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder.

The default is that StarTeam folders are not differentiated based on the case of letters in their names.

With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.

-encrypt

Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks.

This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server.

Parameter	Description
	The full syntax is: `-encrypt encryptionType`. The types of encryption are: **RC4** RSA RC4 stream cipher (fast). **RC2_ECB** RSA RC2 block cipher (Electronic Codebook). **RC2_CBC** RSA RC2 block cipher (Cipher Block Chaining). These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.  **Note:** For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.
-is	Applies the command to all child folders. Without this option, the command applies only to the specified folder. When this option is used with add-folder, you can add an entire branch of folders to the StarTeam folder hierarchy. When used with `add` or `ci`, the command recursively visits all modified files in all sub-folders and checks them in.
-nivf	If `-nivf` is included, then files in `Not in View` folders are also included in the action.

-rp	Overrides the working folder or working directory for the StarTeam view's root folder.
	
	While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways.
	
	While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project.
	
	The UNIX shell interprets a backslash (`\`) as an escape character when it precedes certain characters, such as quotation marks. As a result, an error occurs in the following example:
	`stcmd ci -p "xxx" -rp "C:\" "*"`
	
	which is interpreted as:
	`stcmd ci -p "xxx" -rp "C:" *"`
	
	To avoid a situation like this, escape the final character in "`C:\`" as follows:
	`stcmd ci -p "xxx" -rp "C:\\" "*"`
	
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in "`C:\orion\`":
	`stcmd ci -p "xxx" -rp "C:\orion" "*"`
	
	The full syntax is: `-rp "folderName"` .
	
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

Parameter	Description

-fp	Overrides the specified StarTeam folder's working folder or working directory. This is equivalent to setting an alternate working path for the folder.
	
	While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways.
	
	While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project.
	
	A backslash (\) is interpreted as an escape character when it precedes quotation marks. As a result, an error occurs in the following example:
	`stcmd ci -p "xxx" -fp "C:\" "*"`
	
	which is interpreted as:
	`stcmd ci -p "xxx" -fp "C:" *"`
	
	To avoid a situation like this, escape the final character in "C:\" as follows:
	`stcmd ci -p "xxx" -fp "C:\\" "*"`
	
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in `C:\orion\`:
	`stcmd ci -p "xxx" -fp "C:\orion" "*"`
	
	The full syntax is: `-rp "folderName"`.
	
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-filter	Specifies a string of one or more characters, each of which represents a file status. Never include spaces or other white space in this string. Only files that currently have the specified status(es) will be actioned. Does not apply to files that are `Not In View`. • `C = Current` • `M = Modified` • `O = Out of date` • `N = Not In View` • `I = Missing` • `G = Merge` • `U = Unknown` For example, using `CM` applies a command only to files with a status of `Current` or `Modified`. `-filter` takes precedence over `-f NCI`. If you use `G`, `O`, or `U`, you must also specify `-I` or `-o`. Otherwise the `G`, `O`, or `U` is ignored. `-filter` also takes precedence over `-f NCO`. If you use `G`, `M`, or `U`, you must also specify `-o` to force the checkout operation. Otherwise, the `G`, `M`, or `U` is ignored.
-cfgl	Configures the view using the specified label. Without `-cfgl`, `-cfgp`, or `-cfgd`, the view's current configuration is used.
-cfgp	Configures the view using the specified promotion state.
-cfgd	Configures the view as of the specified date/time. Examples include: "12/29/13 10:52 AM"

Parameter	Description
	"December 29, 2013 10:52:00 AM PST"
	"Monday, December 29, 2013 10:52:00 AM PST"
-q	Enables quiet mode. The `-q` option is retained for backward compatibility with the old command line. If `-q` is specified, then `-pf` cannot be specified. The command will return no results.
-pf	Specifies the filter name whose associated filter properties produce the columns in the output matrix. Each command returns a result matrix. `-pf` determines the matrix columns. See `-ofp` for more information. If not specified, the primary descriptor property of the Type is returned as the command output. `-pf` does not apply to the select query command.
-ofp	Provides a file name with a fully qualified path into which to write the command output. By default, a "|" character separates each column in the output. A new line separates each row. The first row is the command name. The second row has the property names. All subsequent rows contain the data. If the file already exists, the output is appended to the end of the file. It is possible to override the "|" character separator by specifying `separator = fieldSeparator` as a parameter to the connect command. For example, `separator = ;;` specifies two adjacent semicolons (`;`) as the column separator.
files...	Specifies the files to be used in the command by name or by file name-pattern specification, such as `"*.c"`. All options are interpreted using the semantic conventions of UNIX instead of Windows because UNIX conventions are more specific. This means that `"*"`, rather than `"*.*"` means “all files.” The pattern `"*.*"` means “all files with file name extensions.” For example, `“star*.*”` finds `starteam.doc` and `starteam.cpp`, but not `starteam`. To find all of these, you could use `"star*"`. Without this option, the default is `"*"`. When used, this option must always be the last option. Any options after it are ignored. If you use `*`, rather than `"*"` to indicate all files, a UNIX shell expands it into a series of items and passes this series as a group of options to the `stcmd` command. This can cause problems, for example, when you are checking out missing files, so it is best to use `"*"` to avoid unwanted complications. If you use a set of file patterns, each pattern should be enclosed in its own set of quotation marks. For example, you can use `"*.bat" "*.c"`, but you cannot use `"*.bat *.c"`.  **Note:** Always enclose this option in quotation marks. Failure to do so can result in unpredictable consequences on all supported platforms. Several special characters can be used in the file specification: ***** Matches any string including the empty string. For example, * matches any file name, with or without an extension. "xyz*" will match "xyz" and "xyz.cpp" and "xyzutyfj". **?** Matches any single character. For example, `"a?c"` will match `"abc"` but NOT `"ac"`. **[...]** Matches any one of the characters enclosed by the left and right brackets. **-** A pair of characters separated by a hyphen (`-`) specifies a range of characters to be matched. If the first character following the right bracket (`[`) is an exclamation point (`!`) or a caret (`^`), the rest of the characters are not matched. Any character not enclosed in the brackets is

Parameter	Description
	matched. For example, "x[a-d]y" matches "xby" but not "xey". "x[!a-d]y" matches "xey" but not "xby". A hyphen (-) or right bracket ( ]) may be matched by including it as the first or last character in the bracketed set. To use an asterisk (*), question mark (?), or left bracket ([ ) in a pattern, you must precede it with the escape character (which is the backslash (\).

Example

The following example uses `delete-local` to delete some files from the working folder for the StarTeam folder named `SourceCode`. `SourceCode` is a child of the root folder `StarDraw` (in the `StarDraw` view of the `StarDraw` project). This example deletes all files that are not under version control. Those files have the file status `Not In View`.

Use the `-p` with `delete-local` or the stateful `set` command to set the context of the project/view/parent folder.

```
stcmd delete-local -filter "N" "*"
```

Describe Schema: describe

Use the `describe` command to provide a description of the schema of the specified type.

Syntax

The syntax for this command is:

```
stcmd{Ex} describe {type}{.property} {-s "server specifier"}
```

Parameter	Description
type	Specifies the StarTeam item type by name. Types are mutually exclusive.
type.property	The name of an enumerated property on the specified type. For example: `Story.Tag`. If type is specified without `{.property}` then all properties of the type are listed. When `type.property` is specified, and property is an enumerated property, then all the enumerated values of that enumerated property are listed.
-s	Identifies the StarTeam Server. The full syntax is: `-s "userName:password@host:portNumber"` For example: `-s "JMarsh:password@orion:49201"` If the user name is omitted, the current user name is used. The user name in the example is “`JMarsh`”. If the password is omitted, the user is prompted to enter the password. The password in the example is “`password`”. If the host name is omitted, the default is `localhost`. The host name in the example is “`orion`”. The port number is required. The default port number, `49201`, is used in the example.

Example

The following example obtains a description of the Change Request schema for the specified address.

`describe` may be used in the context of a `connect`/`disconnect`, or the connection details may be specified in-line using `-s`.

```
stcmd connect user:password@host:port
stcmd describe ChangeRequest
stcmd disconnect
```

or

```
stcmd{Ex} describe ChangeRequest –s “user:password@host:port”
```

Detach Label: detach-label

Use the `detach-label` command to remove labels from the specified IDs.

Syntax

The syntax for this command is:

```
stcmd{Ex} detach-label [-p "projectSpecifier" [-pattern "pattern"][-epwdfile
"filePath"] [-cmp]
[-csf] [-encrypt encryptionType]] -lbl "labelName" [-all | -type typeName |
–ifp “inputFilePath” ] [–q | -pf “filter name”] [ -ofp “output file path”]
```

Parameter	Description
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.

Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.

Parameter	Description
	• A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date. • A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view's root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".
-pattern	Qualifies the datetime. It can be specified wherever a date-time is specified, such as `-cfgd`, `-vd`, etc. The pattern must match any valid pattern supported by the java JDK in `java.text.SimpleDateFormat.applyLocalizedPattern(String)`. The pattern may be localized. For every command that takes a `-pattern` parameter, a `-locale` parameter is optionally available. This is the "two character country code".

-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .
	The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.
	Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.
	In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.
	The following is the syntax of the commands that can be used to store an encrypted password.
	Use the following syntax to be prompted for the password that will be encrypted and stored in a file.
	`stcmd store-password -epwdfile "filePath"`
	Use the following syntax to include the encrypted password in the command as clear text.
	Note: This action does not access the network with the clear value.
	`stcmd store-password -epwdfile "filePath" -password "password"`
	After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:
	`stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"`
	Important:

Parameter	Description

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

This is an optional parameter. If not specified, then the platform default is not to compress.

-csf

When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder.

The default is that StarTeam folders are not differentiated based on the case of letters in their names.

With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.

-encrypt

Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks.

This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server.

The full syntax is: `-encrypt encryptionType`.

The types of encryption are:

RC4	RSA RC4 stream cipher (fast).
RC2_ECB	RSA RC2 block cipher (Electronic Codebook).
RC2_CBC	RSA RC2 block cipher (Cipher Block Chaining).

These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.

Note: For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.

-lbl

Specifies the label name on which to perform the action. This option can be used more than once. The application action is for all of the labels on the specified file or revisions.

Parameter	Description
-all	Specifies that the label will be detached from all items it is attached to.
-type	Specifies the type of item. The type is one of the stock type names, such as `changerequest`, `task`, `requirement`, `sprint`, `story`, `plan` or any custom type name that is applicable to the command.
-ifp	Specifies a fully qualified path to a file which contains a list of item IDs. The items associated with item IDs are associated to the label. If `-ifp` is specified, `-filter "fileStatus"` cannot be specified.
-q	Enables quiet mode. The `-q` option is retained for backward compatibility with the old command line. If `-q` is specified, then `-pf` cannot be specified. The command will return no results.
-pf	Specifies the filter name whose associated filter properties produce the columns in the output matrix. Each command returns a result matrix. `-pf` determines the matrix columns. See `-ofp` for more information. If not specified, the primary descriptor property of the Type is returned as the command output. `-pf` does not apply to the select query command.
-ofp	Provides a file name with a fully qualified path into which to write the command output. By default, a `"|"` character separates each column in the output. A new line separates each row. The first row is the command name. The second row has the property names. All subsequent rows contain the data. If the file already exists, the output is appended to the end of the file. It is possible to override the `"|"` character separator by specifying `separator = fieldSeparator` as a parameter to the connect command. For example, `separator = ;;` specifies two adjacent semicolons (`;`) as the column separator.

Important: `-all`, `-type`, `-ifp` are mutually exclusive. You must specify just one of them. An exception will be thrown if there are none or more than one specified.

Note:

If a folder path is specified in `-p`, in conjunction with `-all` or `-type`, then only the items of the appropriate types in that folder path will be detached. Items not in that folder path will be retained.

Example

The item IDs are expected to be attached to the label. The `detach-labels` command removes them from the label.

For example, the file could be the output of a select command, such as:

```
stcmd select viewmemberid from file 'c:/temp/fileids.txt' where attached-
label = 'x'
stcmd detach-label -p… -ifp “c:/temp/fields.txt” lbl ‘x’
```

Using the `-pattern` parameter:

```
 co -p "Administrator:Administrator@localhost:49201/project/view" -pattern
"d/M/yy h:m:s" -is -o
```

Produces:

```
 $Date: 28/8/15 8:34:38$
```

Disconnect: disconnect

Use `disconnect` to disconnect from the StarTeam Server. If you have previously connected to the server, the connection persists until you use the `disconnect` command.

Note: This command is part of the stateful model and only works with `stcmd`, not `stcmdEx`.

Syntax

The syntax for this command is as follows:

```
disconnect
```

Example

The following example uses `disconnect` to disconnect from the server.

```
stcmd disconnect
```

Insert: insert

Use the `insert` command to execute a single item insert of the specified values.

The insert statement executes a single item insert if the values are specified in-line or a transacted set of inserts if the values are specified though an input file.

The value clause specification should match that of the property list specification, whether in line or provided through the input file. The types and the number of values should match their corresponding property specifications.

The items are created in the folder described in the folder hierarchy argument of the set statement. If no folder hierarchy is provided, the items are created in the root folder of the selected view. The items are created and saved to the StarTeam Server in a server transaction. All the items are successfully created or none are created. If the insert succeeds, a check-in change package is created, which records the newly created items and their property values. Values that contain spaces should be enclosed in double quotes. This command has been modeled on the standard SQL `Insert` syntax.

Note: Only user modifiable properties can be specified for a value update. Run the describe type command to identify the set of user modifiable properties.

Syntax

The syntax for this command is:

```
stcmd{Ex} insert into type ( propertyName, propertyName,... )
revisions | values [ ( value, value,... ) |
from ‘filePath’ { separator ‘fieldSeparator’} {-pattern "pattern"} ]
{output* | {propertyName,...} | filter='myFilter' into "outputFilePath"
[-p "userName:password@hostName:endpoint/projectName/[viewName/]
[folderHierarchy/]"]
```

Parameter	Description
propertyName	Specifies the subset of properties for the type.
revisions	Treats the contents in the file as a set of revisions of the same item. Accordingly, the first (oldest) revision is added to and all subsequent revisions are updated in sequence.

Parameter	Description
	In this case, the content is expected to be presented so that the oldest revision is the first row of the file and the newest revision is the final row of the file. The revisions syntax provides customers a mechanism to make one-off copies of items (with history) from non-StarTeam to StarTeam repositories or to make copies of StarTeam assets across projects/repositories.
filePath	The path to a file containing multiple items whose values will be inserted. Each row is separated by a new line. Each column is separated by the specified `fieldSeparator` or "|" if a separator is not specified.
-pattern	Qualifies the datetime. It can be specified wherever a date-time is specified, such as `-cfgd`, `-vd`, etc. The pattern must match any valid pattern supported by the java JDK in `java.text.SimpleDateFormat.applyLocalizedPattern(String)`. The pattern may be localized. For every command that takes a `-pattern` parameter, a `-locale` parameter is optionally available. This is the "two character country code".
output	Turns on logging of the command to a log file specified by `INTO`. The `INSERT`, `DELETE`, and `UPDATE` commands log the selected properties of the inserted items to a log file. The property values are separated by the specified `fieldSeparator`, or "|" if a separator is not specified.
myFilter	Specifies a filter by name, whose properties are written to the output file.
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.

Note: If the clear text password contains the @ or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The @ or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.

Parameter	Description
	• The project name is always required. • A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date. • A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view's root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

Property Values

The following are the property values for the command:

Property Type	Value
Text	Literal string.
Integer	A string in the form of an integer like "`1234`".
Double	A string in the form of a double like "`1234.5678`".
Long	A string in the form of a long like "`1234567890`".
Boolean	The string "`true`" or "`false`" - case insensitive.
Date	String format `yyyy-mm-dd`, 4 digit year, `1 <= mm <= 12`, `1 <= dd <= 31`.
DateTime	If `-pattern "pattern"` is specified, then it is parsed using `java.text.SimpleDateFormat`, localized pattern set to "`pattern`". See *http://docs.oracle.com/javase/7/docs/api/java/text/SimpleDateFormat.html*. If -pattern is not specified, attempt to match patterns using `java.text.DateFormat {SHORT, MEDIUM, LONG, FULL}` in that order. See *http://docs.oracle.com/javase/7/docs/api/java/text/DateFormat.html*. If all else fails, try ISO8601 parsing e.g.: `yyyy-mm-ddThh:mm:ssZ` (ignore fractional content after seconds).
TimeSpan	String format `[ws][-][d. |d]hh:mm:ss[.ff][ws]`, items in brackets optional. See `com.starteam.util.TimeSpan`. `ws` whitespace, `d` days, `ff` fractional second, `hh` hours, `mm` minutes `0 <= mm <= 59`, `ss` seconds `0 <= ss <= 59`.

Enumerated	String. Enumerated value specified may be internal name, display name, or string representation of integer enumeration code. If the `Enumerated` property is multi-selectable, two or more enums may be specified as values. In this case, they must be separated by a period. For example: `101.102.103`. Here are some examples:
	`stcmd insert into story (name, tag) values ("This is a story name", 101.103)`
	`stcmd update story set tag = 102.103 where viewmemberid = 1234`

Property Type	Value	
Object	**User string**	Value specified may be user name or string representation of integer user id.
	Group string	Value specified may be group name or string representation of integer group id.
	Label string	Value specified may be label name or string representation of integer label id.
	LinkValue	A string in the form of an integer like "1234" which represents the viewmemberid of the source or target link of a trace.

Example

The following is an example of insert from file:

```
stcmdEx insert into story (name, tag) from c:\temp\story.csv separator “|”
```

The contents of `story.csv` would look like this

```
"This is a story name"|101.103
 "This is a second story"|105
```

Whatever the user modifiable property names you specify in the command must match up in order with the content of the CSV file. Note that the separator in the file must match what you tell the command as the separator.

List-Groups: list-groups

Use the `list-groups` command to list all of the groups in the server.

Important: Requires administrative permissions.

Syntax

```
stcmd{Ex} list-groups [-{e}pwdfile "filePath"] [-cmp] [-encrypt
encryptionType] –s username[:password]@host:port
```

Parameter	Description
-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` . The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore. **Note:** When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter. In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`. The following is the syntax of the commands that can be used to store an encrypted password.

Parameter	Description

Use the following syntax to be prompted for the password that will be encrypted and stored in a file.

```
stcmd store-password -epwdfile "filePath"
```

Use the following syntax to include the encrypted password in the command as clear text.

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password "password"
```

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

This is an optional parameter. If not specified, then the platform default is not to compress.

-encrypt

Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks.

This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server.

The full syntax is: `-encrypt encryptionType`.

The types of encryption are:

RC4	RSA RC4 stream cipher (fast).
RC2_ECB	RSA RC2 block cipher (Electronic Codebook).
RC2_CBC	RSA RC2 block cipher (Cipher Block Chaining).

These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.

Note: For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.

Parameter	Description
-s	Identifies the StarTeam Server. The full syntax is: `-s "userName:password@host:portNumber"` For example: `-s "JMarsh:password@orion:49201"` If the user name is omitted, the current user name is used. The user name in the example is "`JMarsh`". If the password is omitted, the user is prompted to enter the password. The password in the example is "`password`". If the host name is omitted, the default is `localhost`. The host name in the example is "`orion`". The port number is required. The default port number, `49201`, is used in the example.

List Labels: list-labels

Use the `list-labels` command to list the active labels for the selected project or view.

The command can be used in stateful (`connect ...`, `set ...`, `list-labels`, `disconnect`) or stateless (`list-labels -p ...`) modes.

Syntax

The syntax for this command is:

```
stcmd{Ex} list-labels [-p] [-d]
```

Parameter	Description

-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .
	
	The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.
	
	Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.
	
	In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.
	
	The following is the syntax of the commands that can be used to store an encrypted password.
	
	Use the following syntax to be prompted for the password that will be encrypted and stored in a file.
	`stcmd store-password -epwdfile "filePath"`
	
	Use the following syntax to include the encrypted password in the command as clear text.
	
	Note: This action does not access the network with the clear value.
	
	`stcmd store-password -epwdfile "filePath" -password "password"`

Parameter	Description

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-p

Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.

Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.
- A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.
- A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit

Parameter	Description
	the folder hierarchy if the file is in the view's root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".
-d	A user specified Description. However, we continue to support `-r` as an alternate to `-d` for the description, but strictly for backward compatibility
-cmp	Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place. Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data. This is an optional parameter. If not specified, then the platform default is not to compress.
-encrypt	Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks. This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server. The full syntax is: `-encrypt encryptionType`. The types of encryption are: **RC4** RSA RC4 stream cipher (fast). **RC2_ECB** RSA RC2 block cipher (Electronic Codebook). **RC2_CBC** RSA RC2 block cipher (Cipher Block Chaining). These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it. **Note:** For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.
-pattern	Qualifies the datetime. It can be specified wherever a date-time is specified, such as `-cfgd`, `-vd`, etc. The pattern must match any valid pattern supported by the java JDK in `java.text.SimpleDateFormat.applyLocalizedPattern(String)`. The pattern may be localized. For every command that takes a `-pattern` parameter, a `-locale` parameter is optionally available. This is the "two character country code".

Example

```
stcmd list-labels -p Administrator:Administrator@localhost:49201/StarDraw/
Release 1.0 Maintenance
```

List Projects: list-projects

Use the `list-projects` command to list all of the projects in the StarTeam Server.

Syntax

The syntax for this command is:

```
stcmd{Ex} list-projects -s ...
```

Parameter	Description

-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .
	
	The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.
	
	Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.
	
	In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.
	
	The following is the syntax of the commands that can be used to store an encrypted password.
	
	Use the following syntax to be prompted for the password that will be encrypted and stored in a file.
	`stcmd store-password -epwdfile "filePath"`
	
	Use the following syntax to include the encrypted password in the command as clear text.
	
	Note: This action does not access the network with the clear value.
	
	`stcmd store-password -epwdfile "filePath" -password "password"`
	
	After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:
	`stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"`
	
	Important:
	
	If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:
	`-p user@hostname:port/projectName.viewName -epwdfile "pathToPasswordFile"`

-cmp	Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place. Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data. This is an optional parameter. If not specified, then the platform default is not to compress.

Parameter	Description
-encrypt	Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks. This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server. The full syntax is: `-encrypt encryptionType`. The types of encryption are: **RC4** RSA RC4 stream cipher (fast). **RC2_ECB** RSA RC2 block cipher (Electronic Codebook). **RC2_CBC** RSA RC2 block cipher (Cipher Block Chaining). These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it. **Note:** For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.
-s	Identifies the StarTeam Server. The full syntax is: `-s "userName:password@host:portNumber"` For example: `-s "JMarsh:password@orion:49201"` If the user name is omitted, the current user name is used. The user name in the example is "`JMarsh`". If the password is omitted, the user is prompted to enter the password. The password in the example is "`password`". If the host name is omitted, the default is `localhost`. The host name in the example is "`orion`". The port number is required. The default port number, `49201`, is used in the example.

Example

```
stcmd list-projects -s "Administrator:Administrator@localhost:49201"
```

Note: The server credentials and authentication information must be enclosed in double quotes

List Users: list-users

Use the `list-users` command to list all of the users in the server.

Important: Requires administrative permissions.

Syntax

```
stcmd{Ex} list-users [-{e}pwdfile "filePath"] [-cmp] [-encrypt
encryptionType] -s username[:password]@host:port
```

Parameter	Description

-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .
	
	The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.
	
	Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.
	
	In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.
	
	The following is the syntax of the commands that can be used to store an encrypted password.
	
	Use the following syntax to be prompted for the password that will be encrypted and stored in a file.
	`stcmd store-password -epwdfile "filePath"`
	
	Use the following syntax to include the encrypted password in the command as clear text.
	
	Note: This action does not access the network with the clear value.
	
	`stcmd store-password -epwdfile "filePath" -password "password"`
	
	After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:
	`stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"`
	
	Important:
	
	If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:
	`-p user@hostname:port/projectName.viewName -epwdfile "pathToPasswordFile"`

-cmp	Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place. Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data. This is an optional parameter. If not specified, then the platform default is not to compress.
-encrypt	Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks. This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server.

Parameter	Description
	The full syntax is: `-encrypt encryptionType`. The types of encryption are: **RC4** RSA RC4 stream cipher (fast). **RC2_ECB** RSA RC2 block cipher (Electronic Codebook). **RC2_CBC** RSA RC2 block cipher (Cipher Block Chaining). These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.  **Note:** For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.
-s	Identifies the StarTeam Server. The full syntax is: `-s "userName:password@host:portNumber"` For example: `-s "JMarsh:password@orion:49201"` If the user name is omitted, the current user name is used. The user name in the example is "`JMarsh`". If the password is omitted, the user is prompted to enter the password. The password in the example is "`password`". If the host name is omitted, the default is `localhost`. The host name in the example is "`orion`". The port number is required. The default port number, `49201`, is used in the example.

List Views: list-views

Use the `list-views` command to list the set of all accessible views on a given project.

Syntax

The syntax for this command is:

```
stcmd{Ex} list-views –p "Project specifier"
```

Parameter	Description
-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` . The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.

Parameter	Description

Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.

In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.

The following is the syntax of the commands that can be used to store an encrypted password.

Use the following syntax to be prompted for the password that will be encrypted and stored in a file.

```
stcmd store-password -epwdfile "filePath"
```

Use the following syntax to include the encrypted password in the command as clear text.

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password "password"
```

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile` "`filePath`" as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

This is an optional parameter. If not specified, then the platform default is not to compress.

-encrypt

Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks.

This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server.

The full syntax is: `-encrypt encryptionType`.

The types of encryption are:

RC4	RSA RC4 stream cipher (fast).
RC2_ECB	RSA RC2 block cipher (Electronic Codebook).
RC2_CBC	RSA RC2 block cipher (Cipher Block Chaining).

These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.

Note: For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the

Parameter	Description
	user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.
-p	Describes the project in addition to server connectivity and authentication credentials.

Example

```
stcmd list-views -p "user:password@host:port/projectName"
```

This would return:

```
View,CreatedUserID,CreatedTime,StartDate,EndDate,Type,Folder
Path,Description,View Path
Main,James Wogulis,1999-02-12T00:24:04.000Z,1899-12-30T00:00:00.000Z,
1899-12-30T00:00:00.000Z,Unknown,c:\projects\Main,JavaWorld Main View,Main
StarTeam 4.2,Ron Sauers,2000-01-21T16:17:26.000Z,1899-12-30T00:00:00.000Z,
1899-12-30T00:00:00.000Z,Unknown,c:\whitestar\,,Main/StarTeam 4.2
StarGate 5.0,Ron Sauers,2000-05-19T00:36:36.000Z,1899-12-30T00:00:00.000Z,
1899-12-30T00:00:00.000Z,Unknown,c:\whitestar\,,Main/StarGate 5.0
StarGate 5.1,Ron Sauers,2001-03-19T17:56:56.000Z,1899-12-30T00:00:00.000Z,
1899-12-30T00:00:00.000Z,Unknown,l:\whitestar\,,Main/StarGate 5.1
```

The stateful equivalent would be:

```
stcmd connect Administrator:Administrator@localhost:49201
stcmd set project = StarDraw
stcmd list-views
…
stcmd disconnect
```

Lock Unlock Files: lck

Use `lck` to lock or unlock files from the command line.

Syntax

The syntax for this command is:

```
stcmd{Ex} lck [-is] [-rp "folderPath" | -fp "folderPath"]
[-filter "fileStatus"] [-break] [-l |-u | -nel] [-type -id] [-ro | -rw]]
[-pf “filterName”] [-ofp “resultsOutputFilePath”] [files...]
```

Parameter	Description
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.

Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@`

Parameter	Description
	or `:` symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.
- A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.
- A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view’s root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile

The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .

The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.

Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.

In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.

The following is the syntax of the commands that can be used to store an encrypted password.

Parameter	Description

Use the following syntax to be prompted for the password that will be encrypted and stored in a file.

```
stcmd store-password -epwdfile "filePath"
```

Use the following syntax to include the encrypted password in the command as clear text.

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password "password"
```

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

This is an optional parameter. If not specified, then the platform default is not to compress.

-encrypt

Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks.

This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server.

The full syntax is: `-encrypt encryptionType`.

The types of encryption are:

RC4	RSA RC4 stream cipher (fast).
RC2_ECB	RSA RC2 block cipher (Electronic Codebook).
RC2_CBC	RSA RC2 block cipher (Cipher Block Chaining).

These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.

Note: For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.

Parameter	Description
-is	Applies the command to all child folders. Without this option, the command applies only to the specified folder. When this option is used with add-folder, you can add an entire branch of folders to the StarTeam folder hierarchy. When used with `add` or `ci`, the command recursively visits all modified files in all sub-folders and checks them in.

-rp	Overrides the working folder or working directory for the StarTeam view's root folder.
	
	While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways.
	
	While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project.
	
	The UNIX shell interprets a backslash (\) as an escape character when it precedes certain characters, such as quotation marks. As a result, an error occurs in the following example:
	`stcmd ci -p "xxx" -rp "C:\" "*"`
	
	which is interpreted as:
	`stcmd ci -p "xxx" -rp "C:" *"`
	
	To avoid a situation like this, escape the final character in "`C:\`" as follows:
	`stcmd ci -p "xxx" -rp "C:\\" "*"`
	
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in "`C:\orion\`":
	`stcmd ci -p "xxx" -rp "C:\orion" "*"`
	
	The full syntax is: `-rp "folderName"` .
	
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-fp	Overrides the specified StarTeam folder's working folder or working directory. This is equivalent to setting an alternate working path for the folder.
	
	While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways.
	
	While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project.
	
	A backslash (\) is interpreted as an escape character when it precedes quotation marks. As a result, an error occurs in the following example:
	`stcmd ci -p "xxx" -fp "C:\" "*"`
	
	which is interpreted as:
	`stcmd ci -p "xxx" -fp "C:" *"`
	
	To avoid a situation like this, escape the final character in "C:\" as follows:
	`stcmd ci -p "xxx" -fp "C:\\" "*"`

Parameter	Description
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in `C:\orion\`: `stcmd ci -p "xxx" -fp "C:\orion" "*"` The full syntax is: `-rp "folderName"`. Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.
-filter	Specifies a string of one or more characters, each of which represents a file status. Never include spaces or other white space in this string. Only files that currently have the specified status(es) will be actioned. Does not apply to files that are `Not In View`. • `C = Current` • `M = Modified` • `O = Out of date` • `N = Not In View` • `I = Missing` • `G = Merge` • `U = Unknown` For example, using `CM` applies a command only to files with a status of `Current` or `Modified`. `-filter` takes precedence over `-f NCI`. If you use `G`, `O`, or `U`, you must also specify `-I` or `-o`. Otherwise the `G`, `O`, or `U` is ignored. `-filter` also takes precedence over `-f NCO`. If you use `G`, `M`, or `U`, you must also specify `-o` to force the checkout operation. Otherwise, the `G`, `M`, or `U` is ignored.
-break	Breaks the current lock by another user if you have the access rights to break locks.
-l | -u | -nel	Locks the item(s). `-l` is exclusive lock, `-u` is unlocked, and `-nel` is non exclusive lock. These items are mutually exclusive and an optional parameter.
-type	Specifies the type of item. The type is one of the stock type names, such as `changerequest`, `task`, `requirement`, `sprint`, `story`, `plan` or any custom type name that is applicable to the command.
-id	Specifies the unique item (view member) ID of the item. Look in the property lists of the CPC or query using the select command to find the View Member IDs. `-id` can also specify the primary descriptor of the item; e.g. file name, folder name, change request number.
-ro	Makes the working file read-only after this operation. Without this option, the file remains as it was prior to the operation. Usually, you use `-ro` to prevent yourself from editing a file that is not locked by you. `-ro` must be used with `-l` or`-u` or `-nel`. If you use `-ro`, you cannot use `-rw`.
-rw	Makes the working file read-write after this operation. Without this option, the file remains as it was prior to the operation. `-rw` must be used with `-l` or `-u` or `-nel`. If you use `-rw`, you cannot use `-ro`.
-pf	Specifies the filter name whose associated filter properties produce the columns in the output matrix. Each command returns a result matrix. `-pf` determines the matrix columns. See `-ofp` for more information. If not specified, the primary descriptor property of the Type is returned as the command output. `-pf` does not apply to the select query command.
-ofp	Provides a file name with a fully qualified path into which to write the command output. By default, a "|" character separates each column in the output. A new line separates each row.

Parameter	Description
	The first row is the command name. The second row has the property names. All subsequent rows contain the data. If the file already exists, the output is appended to the end of the file. It is possible to override the "|" character separator by specifying `separator = fieldSeparator` as a parameter to the connect command. For example, `separator = ;;` specifies two adjacent semicolons (`;`) as the column separator.
files...	Specifies the files to be used in the command by name or by file name-pattern specification, such as `"*.c"`. All options are interpreted using the semantic conventions of UNIX instead of Windows because UNIX conventions are more specific. This means that `"*"`, rather than `"*.*"` means “all files.” The pattern `"*.*"` means “all files with file name extensions.” For example, `“star*.*”` finds `starteam.doc` and `starteam.cpp`, but not `starteam`. To find all of these, you could use `"star*"`. Without this option, the default is `"*"`. When used, this option must always be the last option. Any options after it are ignored. If you use `*`, rather than `"*"` to indicate all files, a UNIX shell expands it into a series of items and passes this series as a group of options to the `stcmd` command. This can cause problems, for example, when you are checking out missing files, so it is best to use `"*"` to avoid unwanted complications. If you use a set of file patterns, each pattern should be enclosed in its own set of quotation marks. For example, you can use `"*.bat" "*.c"`, but you cannot use `"*.bat *.c"`. **Note:** Always enclose this option in quotation marks. Failure to do so can result in unpredictable consequences on all supported platforms. Several special characters can be used in the file specification: ***** Matches any string including the empty string. For example, * matches any file name, with or without an extension. "xyz*" will match "xyz" and "xyz.cpp" and "xyzutyfj". **?** Matches any single character. For example, `"a?c"` will match `"abc"` but NOT `"ac"`. **[...]** Matches any one of the characters enclosed by the left and right brackets. **-** A pair of characters separated by a hyphen (`-`) specifies a range of characters to be matched. If the first character following the right bracket (`[`) is an exclamation point (`!`) or a caret (`^`), the rest of the characters are not matched. Any character not enclosed in the brackets is matched. For example, `"x[a-d]y"` matches `"xby"` but not `"xey"`. `"x[!a-d]y"` matches `"xey"` but not `"xby"`. A hyphen (`-`) or right bracket (`]`) may be matched by including it as the first or last character in the bracketed set. To use an asterisk (*), question mark (?), or left bracket (`[`) in a pattern, you must precede it with the escape character (which is the backslash (\).

Example

The following example uses `stcmd lck` to unlock all files in `SourceCode`, a child of the root folder `StarDraw` (in the `StarDraw` view of the `StarDraw` project), as well as all files in child folders of `SourceCode`.

Use the `-p` with `lck` or the stateful `set` command to set the context of the project/view/parent folder.

```
stcmd lck -is -u "*"
```

Use `lck` with `-type` and `-id` to specify a lock on a non-file artifact. For example:

```
stcmd  lck –l –type task –id 1234567 –p “user:pwd@host:port/projectName/
viewName/”
```

The `-id` parameter is either the view member ID (preferred) of the artifact or the primary property value, e.g. task number, which is slower.

Make Public: make-public

Use `make-public` to convert private filters or queries into public filters or queries. This command is available on StarTeam Server 16.1 and later.

The type name is required.

Either one or both of `filterName` or `queryName` must be provided. If both are provided, then both the filter and query pair will be made public. If neither one is provided, the command fails with an exception. If only one of the two is provided, then the specified filter or query will be made public.

When working with private filter or query pairs, both filter and query names must be specified. Otherwise, the server will throw an exception.

Syntax

The syntax for this command is:

```
make-public –type typeName [ -filter filtername] [ -query queryName ] –s
“user:[password]@host:port”  [ -epwdfile “path to password file” ]
```

Parameter	Description
-type	Specifies the type of item. The type is one of the stock type names, such as `changerequest`, `task`, `requirement`, `sprint`, `story`, `plan` or any custom type name that is applicable to the command.
-filter	Specifies a string of one or more characters, each of which represents a file status. Never include spaces or other white space in this string. Only files that currently have the specified status(es) will be actioned. Does not apply to files that are `Not In View`. • `C = Current` • `M = Modified` • `O = Out of date` • `N = Not In View` • `I = Missing` • `G = Merge` • `U = Unknown` For example, using `CM` applies a command only to files with a status of `Current` or `Modified`. `-filter` takes precedence over `-f NCI`. If you use `G`, `O`, or `U`, you must also specify `-I` or `-o`. Otherwise the `G`, `O`, or `U` is ignored. `-filter` also takes precedence over `-f NCO`. If you use `G`, `M`, or `U`, you must also specify `-o` to force the checkout operation. Otherwise, the `G`, `M`, or `U` is ignored.
-query	Specifies the query name to make public.

Parameter	Description
-s	Identifies the StarTeam Server. The full syntax is: `-s "userName:password@host:portNumber"` For example: `-s "JMarsh:password@orion:49201"` If the user name is omitted, the current user name is used. The user name in the example is "`JMarsh`". If the password is omitted, the user is prompted to enter the password. The password in the example is "`password`". If the host name is omitted, the default is `localhost`. The host name in the example is "`orion`". The port number is required. The default port number, `49201`, is used in the example.

-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .
	
	The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.
	
	Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.
	
	In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.
	
	The following is the syntax of the commands that can be used to store an encrypted password.
	
	Use the following syntax to be prompted for the password that will be encrypted and stored in a file.
	`stcmd store-password -epwdfile "filePath"`
	
	Use the following syntax to include the encrypted password in the command as clear text.
	
	Note: This action does not access the network with the clear value.
	
	`stcmd store-password -epwdfile "filePath" -password "password"`
	
	After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:
	`stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"`
	
	Important:
	
	If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:
	`-p user@hostname:port/projectName.viewName -epwdfile "pathToPasswordFile"`

Example

The following example changes both a private filter and a private query to public.

```
make-public –type changerequest –filter “my private filter” –query “my private query” –s “Administrator:Administrator@localhost:49201”
```

Manage User: manage-user

To manage another user, the command must be run by an administrator, and must specify the logon name of the user being managed.

Note: It is not possible to manage your own account with this command unless you are changing a password with `-changePassword`.

Syntax

The syntax for this command is:

```
stcmd{ex} manage-user -s "userName:password@host:port" -logonName
userlogonName [-suspend] [ -activate ] [ -forcePasswordChange ] [ -
forceLogoff ] [ -changePassword newPassword ]
```

Parameter	Description
-s	Identifies the StarTeam Server. The full syntax is: `-s "userName:password@host:portNumber"` For example: `-s "JMarsh:password@orion:49201"` If the user name is omitted, the current user name is used. The user name in the example is "`JMarsh`". If the password is omitted, the user is prompted to enter the password. The password in the example is "`password`". If the host name is omitted, the default is `localhost`. The host name in the example is "`orion`". The port number is required. The default port number, `49201`, is used in the example.
-logonName	Designates the user you want to manage. `-logonName` is required for all uses of this command. Type the users logon name.
-suspend	Suspends the designated user account.
-activate	Activates the designated user account.
-forcePasswordChange	Forces a password change for the designated user account.
-forceLogoff	Forces the designated user to logoff.
-changePassword newPassword	Changes the designated users password to the specified new password. This is the only parameter where a user can change their own account.

Example

The following examples can be used by and administrator to manage another user account:

```
manage-user –s “Administrator:password@host:port”  -logonName “joeuser” –
suspend
```

```
manage-user –s “Administrator:password@host:port”  -logonName “joeuser” –
activate
```

```
manage-user –s “Administrator:password@host:port”  -logonName “joeuser” –
forceLogoff
```

```
manage-user –s “Administrator:password@host:port”  -logonName “joeuser” –
changePassword “joepassword”
```

A non-administrative user can also run this command, but only to change his/her own password as follows:

```
manage-user –s “joeuser:password@host:port”  -logonName joeuser –
changePassword “mynewpassword”
```

Note: In this case, `-logonName` specifies the same name as the login user name.

Merge Label: merge-label

The `merge-label` command creates a new label (if it does not already exist) in the target view, copying the properties of the source label from the source view.

Once the label has been created, the command finds the items in the source view attached to the source label identifies the set of items in the target view that need to be attached to the target label attaches them at the tip and then moves them back to the revision at which the items were attached in the source.

Syntax

The syntax for this command is:

```
merge-label -sourceview "[project name/]view name"
-lbl "label name" {-type typeName}
-p "user:password@host:port/project/view" -epwdfile
```

While this command supports both view and revision labels, it is more useful for view labels (which are not supported through the StarTeam Cross-Platform Client), unlike (cross-project) Revision Label copy, which is.

This command supports attaching labels both to items that have been shared across (in-project) views as well as to items that have been moved across projects.

Note: Only item types are supported. For example *File*, *ChangeRequest*, *Task*, etc.

Parameters

-sourceview Required parameter. It can describe a view within the project or a view from a different project. To specify cross-project views, the source view value can optionally take a project name followed by a `/` followed by the view name. Source view may also specify the target view name, for in view label merges.

-lbl Specifies the label name on which to perform the action. This option can be used more than once. The application action is for all of the labels on the specified file or revisions.

-type Specifies the type of item. The type is one of the stock type names, such as `changerequest`, `task`, `requirement`, `sprint`, `story`, `plan` or any custom type name that is applicable to the command.

-p Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.

Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the

password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp \pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.
- A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.
- A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view’s root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile

The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .

The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.

Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.

In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.

The following is the syntax of the commands that can be used to store an encrypted password.

Use the following syntax to be prompted for the password that will be encrypted and stored in a file.

```
stcmd store-password -epwdfile "filePath"
```

Use the following syntax to include the encrypted password in the command as clear text.

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password "password"
```

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile` "`filePath`" as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

This is an optional parameter. If not specified, then the platform default is not to compress.

-encrypt

Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks.

This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server.

The full syntax is: `-encrypt encryptionType`.

The types of encryption are:

RC4	RSA RC4 stream cipher (fast).
RC2_ECB	RSA RC2 block cipher (Electronic Codebook).
RC2_CBC	RSA RC2 block cipher (Cipher Block Chaining).

These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.

Note: For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.

Move: move

Use the `move` command to move StarTeam items. You can use this command to move all item types: Folder, File, Change request, Task, Topic, Requirement, Sprint, Story, Concept, WhiteBoard, and custom components.

Syntax

The syntax for this command is:

```
stcmd move –p “user:pwd@host:port/project/view” [ -epwdfile
“pathToPasswordFile” ] [ -csf ] –type “typeName” –id itemID
[ -tp targetProjectName ] [ –tv “targetViewName” ] [ -tfp “target folder
path” ]
```

Parameter	Description
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.

Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.
- A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is

Parameter	Description

not recommended, however, because another view with that name could be created at a later date.

- A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view's root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile

The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .

The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.

Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.

In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.

The following is the syntax of the commands that can be used to store an encrypted password.

Use the following syntax to be prompted for the password that will be encrypted and stored in a file.

```
stcmd store-password -epwdfile "filePath"
```

Use the following syntax to include the encrypted password in the command as clear text.

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password "password"
```

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"'` as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

This is an optional parameter. If not specified, then the platform default is not to compress.

Parameter	Description
-encrypt	Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks. This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server. The full syntax is: `-encrypt encryptionType`. The types of encryption are: **RC4** RSA RC4 stream cipher (fast). **RC2_ECB** RSA RC2 block cipher (Electronic Codebook). **RC2_CBC** RSA RC2 block cipher (Cipher Block Chaining). These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.  **Note:** For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.
-csf	When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder. The default is that StarTeam folders are not differentiated based on the case of letters in their names. With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.
-type	Specifies the type of item. The type is one of the stock type names, such as `changerequest`, `task`, `requirement`, `sprint`, `story`, `plan` or any custom type name that is applicable to the command.
-fp	Overrides the specified StarTeam folder's working folder or working directory. This is equivalent to setting an alternate working path for the folder. While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways. While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project.

Parameter	Description

	A backslash (\) is interpreted as an escape character when it precedes quotation marks. As a result, an error occurs in the following example:
	`stcmd ci -p "xxx" -fp "C:\" "*"`
	which is interpreted as:
	`stcmd ci -p "xxx" -fp "C:" *"`
	To avoid a situation like this, escape the final character in "C:\" as follows:
	`stcmd ci -p "xxx" -fp "C:\\" "*"`
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in `C:\orion\`:
	`stcmd ci -p "xxx" -fp "C:\orion" "*"`
	The full syntax is: `-rp "folderName"`.
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-id	Specifies the unique item (view member) ID of the item. Look in the property lists of the CPC or query using the select command to find the View Member IDs. `-id` can also specify the primary descriptor of the item; e.g. file name, folder name, change request number.
-tp	Describes an optional specific target project into which to move. If not specified, then the item will be moved into the source view described by `-tv`, assuming that is specified. If neither is specified, then the move will occur into the original view as specified by `-p`. If `-tp` is specified, and `-tv` is not, then the move will occur into the default (root) view of the project described by `-tp`.
-tv	Describes an optional specific target view into which to move. If not specified (and `-tp` is not specified), then the item will be moved into the source view described by `-p`.
-tfp	Describes the folder path to which to move. If a folder name is specified, then this name must be unique across the folder tree. Otherwise, a fully qualified folder path is required, starting from the root folder of the view down to the leaf folder into which the move must be created. If `-tfp` is not specified, then the move will be created in the root folder of the target view `-tv`.

Note also that if a share of the item already exists in the target folder, then the move will not occur. An error is returned.

Note that either one of `-tp`, `-tv` or `-tfp` must be specified.

The move command returns no results.

Examples

The following finds `FolderX` in project `AProject`, view `AProject`, under the folder hierarchy `AProject/ChildFolder` and moves it to project `StarDraw`, view `Release 1.0 Maintenance`, as a child folder to `Documents/Images`:

```
stcmd move –p “Administrator:Administrator@localhost:49201/StarDraw/Release
1.0 Maintenance/Documents/Images” –type Folder –fp “AProject/AProject/
ChildFolder/FolderX”
```

The following finds the first occurrence of `FolderX` in project `AProject`, view `AProject` and moves it to project `StarDraw`, view `Release 1.0 Maintenance`, as a child folder to `Documents/Images`:

```
stcmd move –p “Administrator:Administrator@localhost:49201/StarDraw/Release
1.0 Maintenance/Documents/Images” –type Folder –fp “AProject/AProject/FolderX”
```

Network Monitor: monitor

Use `monitor` to start the **Network Monitor** (Netmon). The monitor records server commands issued by the command processor to the StarTeam Server. You can specify monitoring the server, the cache agent, or both. This is useful to track access to a machine or StarTeam Server, as well as troubleshooting.

Note: This command is part of the stateful model and only works with `stcmd`, not `stcmdEx`.

Syntax

The syntax for this command is as follows:

```
stcmd monitor stop | start [server | ca | both] –ofp “full path to a writable output log file”
```

Parameter	Description
start	Turns on command logging. Specify which service to monitor: `server`, `ca` (cache agent), or `both`.
stop	Turns off command logging.
-ofp	Provides a file name with a fully qualified path into which to write the command output. By default, a "|" character separates each column in the output. A new line separates each row. The first row is the command name. The second row has the property names. All subsequent rows contain the data. If the file already exists, the output is appended to the end of the file. It is possible to override the "|" character separator by specifying `separator = fieldSeparator` as a parameter to the connect command. For example, `separator = ;;` specifies two adjacent semicolons (`;`) as the column separator.

Example

The following example uses `monitor` to monitor the server network connection and output the report to a Temp folder on the `C:` drive. If the file already exists, it is appended to the end of the report.

```
stcmd monitor start server -ofp "C:\Temp\netmon.txt"
```

The following example stops the Netmon:

```
stcmd monitor stop
```

Remove Label: remove-label

Use `remove-label` to delete a view or revision label.

Syntax

The syntax for this command is:

```
stcmd{Ex} remove-label –lbl “labelName”
```

Parameter	Description
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue

Parameter	Description

to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.

Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.
- A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.
- A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view’s root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile

The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .

The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.

Parameter	Description

Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.

In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.

The following is the syntax of the commands that can be used to store an encrypted password.

Use the following syntax to be prompted for the password that will be encrypted and stored in a file.

```
stcmd store-password -epwdfile "filePath"
```

Use the following syntax to include the encrypted password in the command as clear text.

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password "password"
```

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile` "`filePath`" as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

This is an optional parameter. If not specified, then the platform default is not to compress.

-encrypt

Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks.

This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server.

The full syntax is: `-encrypt encryptionType`.

The types of encryption are:

RC4	RSA RC4 stream cipher (fast).
RC2_ECB	RSA RC2 block cipher (Electronic Codebook).
RC2_CBC	RSA RC2 block cipher (Cipher Block Chaining).

These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.

Note: For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the

Parameter	Description
	user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.
-lbl	Specifies the label name on which to perform the action. This option can be used more than once. The application action is for all of the labels on the specified file or revisions.

Example

The following example removes the label `Beta` from the `StarDraw` view.

Use the `-p` with `remove-label` or the stateful `set` command to set the context of the project/view/parent folder.

```
stcmd remove-label -lbl "Beta"
```

Remove Files: remove

Use `remove` to remove files from version control. The specified files and their revision histories no longer appear in StarTeam unless you roll back the project view to a time before they were removed.

Syntax

The syntax for this command is:

```
stcmd{Ex} remove [-p "projectSpecifier" [-epwdfile "filePath"] [-cmp] [-csf]
[-encrypt encryptionType] ] [-is] [-rp "folderPath" | -fp "folderPath"]
[[ -active | [-cr | -req | -task ] processItemPath] [-filter "fileStatus"]
[-df] [-q|-pf “filterName”] [-ofp “resultsOutputFilePath”] [files...]
```

Parameter	Description
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.  **Note:** If the clear text password contains the @ or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The @ or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

Parameter	Description

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.
- A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.
- A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view’s root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile

The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .

The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.

Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.

In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.

The following is the syntax of the commands that can be used to store an encrypted password.

Use the following syntax to be prompted for the password that will be encrypted and stored in a file.

```
stcmd store-password -epwdfile "filePath"
```

Use the following syntax to include the encrypted password in the command as clear text.

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password "password"
```

Parameter	Description

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

This is an optional parameter. If not specified, then the platform default is not to compress.

-csf

When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder.

The default is that StarTeam folders are not differentiated based on the case of letters in their names.

With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.

-encrypt

Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks.

This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server.

The full syntax is: `-encrypt encryptionType`.

The types of encryption are:

RC4	RSA RC4 stream cipher (fast).
RC2_ECB	RSA RC2 block cipher (Electronic Codebook).
RC2_CBC	RSA RC2 block cipher (Cipher Block Chaining).

These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.

Note: For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not

Parameter	Description
	exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.
-is	Applies the command to all child folders. Without this option, the command applies only to the specified folder. When this option is used with add-folder, you can add an entire branch of folders to the StarTeam folder hierarchy. When used with `add` or `ci`, the command recursively visits all modified files in all sub-folders and checks them in.

-rp	Overrides the working folder or working directory for the StarTeam view's root folder.
	
	While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways.
	
	While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project.
	
	The UNIX shell interprets a backslash (`\`) as an escape character when it precedes certain characters, such as quotation marks. As a result, an error occurs in the following example:
	`stcmd ci -p "xxx" -rp "C:\" "*"`
	
	which is interpreted as:
	`stcmd ci -p "xxx" -rp "C:" *"`
	
	To avoid a situation like this, escape the final character in "`C:\`" as follows:
	`stcmd ci -p "xxx" -rp "C:\\" "*"`
	
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in "`C:\orion \`":
	`stcmd ci -p "xxx" -rp "C:\orion" "*"`
	
	The full syntax is: `-rp "folderName"` .
	
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-fp	Overrides the specified StarTeam folder's working folder or working directory. This is equivalent to setting an alternate working path for the folder. While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways. While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project. A backslash (`\`) is interpreted as an escape character when it precedes quotation marks. As a result, an error occurs in the following example: `stcmd ci -p "xxx" -fp "C:\" "*"`

Parameter	Description

	which is interpreted as:
	`stcmd ci -p "xxx" -fp "C:" *"`
	To avoid a situation like this, escape the final character in "C:\" as follows:
	`stcmd ci -p "xxx" -fp "C:\\" "*"`
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in `C:\orion\`:
	`stcmd ci -p "xxx" -fp "C:\orion" "*"`
	The full syntax is: `-rp "folderName"`.
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-lbl	Specifies the label name on which to perform the action. This option can be used more than once. The application action is for all of the labels on the specified file or revisions.
-active	The active process item.
-cr, -req, -task	Complete path from the project view's root folder to the change request, requirement, or task number to be used as a process item. Use the forward slash (`/`) as a delimiter between folder names. For out-of-view process items, specify the project name and view name in front of the complete folder path. For example: `-cr MyProject/RootView/RootFolder/SourceCode/37` This specifies change request `37` in the `SourceCode` folder (under the root folder) of the `ChildView` view in the `MyProject` project. **Note:** For in-view process items, as long as the change request, requirement, or task numbers are the unique primary descriptors of their types (true by default), it is sufficient simply to specify the number, with no path. The project and view names are assumed from `-p`. If a process item is specified, then the files being checked in are attached to the process item and follow the project process rules. `-cr`, `-req` or `-task` are mutually exclusive. If any one of them is specified, `-filter`/`-f` are ignored.
-filter	Specifies a string of one or more characters, each of which represents a file status. Never include spaces or other white space in this string. Only files that currently have the specified status(es) will be actioned. Does not apply to files that are `Not In View`. • `C = Current` • `M = Modified` • `O = Out of date` • `N = Not In View` • `I = Missing` • `G = Merge` • `U = Unknown` For example, using `CM` applies a command only to files with a status of `Current` or `Modified`. `-filter` takes precedence over `-f NCI`. If you use `G`, `O`, or `U`, you must also specify `-I` or `-o`. Otherwise the `G`, `O`, or `U` is ignored.

Parameter	Description
	`-filter` also takes precedence over `-f NCO`. If you use `G`, `M`, or `U`, you must also specify `-o` to force the checkout operation. Otherwise, the `G`, `M`, or `U` is ignored.
-df	Deletes the user's working file. Without this option the working file remains in the working folder on your workstation.
-q	Enables quiet mode. The `-q` option is retained for backward compatibility with the old command line. If `-q` is specified, then `-pf` cannot be specified. The command will return no results.
-pf	Specifies the filter name whose associated filter properties produce the columns in the output matrix. Each command returns a result matrix. `-pf` determines the matrix columns. See `-ofp` for more information. If not specified, the primary descriptor property of the Type is returned as the command output. `-pf` does not apply to the select query command.
-ofp	Provides a file name with a fully qualified path into which to write the command output. By default, a `"|"` character separates each column in the output. A new line separates each row. The first row is the command name. The second row has the property names. All subsequent rows contain the data. If the file already exists, the output is appended to the end of the file. It is possible to override the `"|"` character separator by specifying `separator = fieldSeparator` as a parameter to the connect command. For example, `separator = ;;` specifies two adjacent semicolons (`;`) as the column separator.
files...	Specifies the files to be used in the command by name or by file name-pattern specification, such as `"*.c"`. All options are interpreted using the semantic conventions of UNIX instead of Windows because UNIX conventions are more specific. This means that `"*"`, rather than `"*.*"` means "all files." The pattern `"*.*"` means "all files with file name extensions." For example, `"star*.*"` finds `starteam.doc` and `starteam.cpp`, but not `starteam`. To find all of these, you could use `"star*"`. Without this option, the default is `"*"`. When used, this option must always be the last option. Any options after it are ignored. If you use `*`, rather than `"*"` to indicate all files, a UNIX shell expands it into a series of items and passes this series as a group of options to the `stcmd` command. This can cause problems, for example, when you are checking out missing files, so it is best to use `"*"` to avoid unwanted complications. If you use a set of file patterns, each pattern should be enclosed in its own set of quotation marks. For example, you can use `"*.bat" "*.c"`, but you cannot use `"*.bat *.c"`.  **Note:** Always enclose this option in quotation marks. Failure to do so can result in unpredictable consequences on all supported platforms. Several special characters can be used in the file specification: ***** Matches any string including the empty string. For example, `*` matches any file name, with or without an extension. "xyz*" will match "xyz" and "xyz.cpp" and "xyzutyfj". **?** Matches any single character. For example, `"a?c"` will match `"abc"` but NOT `"ac"`. **[...]** Matches any one of the characters enclosed by the left and right brackets.

Parameter	Description
	- A pair of characters separated by a hyphen (`-`) specifies a range of characters to be matched. If the first character following the right bracket (`[`) is an exclamation point (`!`) or a caret (`^`), the rest of the characters are not matched. Any character not enclosed in the brackets is matched. For example, `"x[a-d]y"` matches `"xby"` but not `"xey"`. `"x[!a-d]y"` matches `"xey"` but not `"xby"`. A hyphen (`-`) or right bracket (`]`) may be matched by including it as the first or last character in the bracketed set. To use an asterisk (*), question mark (?), or left bracket (`[`) in a pattern, you must precede it with the escape character (which is the backslash (\).

Example

The following example uses `remove` to remove all `.hm` files from `SourceCode`, a child of the root folder `StarDraw` (in the `StarDraw` view of the `StarDraw` project), as well as all files in child folders of `SourceCode`. It also deletes the working files.

Use the `-p` with `remove` or the stateful `set` command to set the context of the project/view/parent folder.

```
stcmd remove -rp "StarDraw/StarDraw/SourceCode" -is -df "*.hm"
```

Remove Project: remove-project

Use the `remove-project` command to delete a project from the StarTeam Server.

Syntax

The syntax for this command is:

```
stcmd{Ex} remove-project –p “project specifier”
```

Parameter	Description
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands. **Note:** If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

Parameter	Description

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.
- A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.
- A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view’s root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile

The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .

The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.

Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.

In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.

The following is the syntax of the commands that can be used to store an encrypted password.

Use the following syntax to be prompted for the password that will be encrypted and stored in a file.

```
stcmd store-password -epwdfile "filePath"
```

Use the following syntax to include the encrypted password in the command as clear text.

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password "password"
```

Parameter	Description

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

This is an optional parameter. If not specified, then the platform default is not to compress.

-csf

When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder.

The default is that StarTeam folders are not differentiated based on the case of letters in their names.

With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.

-encrypt

Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks.

This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server.

The full syntax is: `-encrypt encryptionType`.

The types of encryption are:

RC4	RSA RC4 stream cipher (fast).
RC2_ECB	RSA RC2 block cipher (Electronic Codebook).
RC2_CBC	RSA RC2 block cipher (Cipher Block Chaining).

These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.

Note: For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam

Parameter	Description
	Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.

Example

The following example removes the project `StarDraw` from the server.

```
stcmd remove-project -p "Administrator:Administrator@localhost:49201/StarDraw"
```

Remove View: remove-view

Use the `remove-view` command to delete a view from the StarTeam Server.

The root (default) view of a project cannot be deleted. You need to delete the project itself.

A view with child views cannot be deleted until all child views have been individually deleted.

Syntax

The syntax for this command is:

```
stcmd{Ex} remove-view –p “project specifier”
```

Parameter	Description
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.

Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.

Parameter	Description

- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.
- A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.
- A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view's root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile

The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .

The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.

Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.

In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.

The following is the syntax of the commands that can be used to store an encrypted password.

Use the following syntax to be prompted for the password that will be encrypted and stored in a file.

```
stcmd store-password -epwdfile "filePath"
```

Use the following syntax to include the encrypted password in the command as clear text.

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password "password"
```

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile` "`filePath`" as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

Parameter	Description
-cmp	Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place. Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data. This is an optional parameter. If not specified, then the platform default is not to compress.
-csf	When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder. The default is that StarTeam folders are not differentiated based on the case of letters in their names. With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.
-encrypt	Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks. This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server. The full syntax is: `-encrypt encryptionType`. The types of encryption are: **RC4** RSA RC4 stream cipher (fast). **RC2_ECB** RSA RC2 block cipher (Electronic Codebook). **RC2_CBC** RSA RC2 block cipher (Cipher Block Chaining). These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it. **Note:** For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.

Example

The following example removes the view `Release 1.0 Maintenance` from the `StarDraw` project on the StarTeam Server.

```
stcmd remove-view -p "Administrator:Administrator@localhost:49201/StarDraw/
Release 1.0 Maintenance"
```

Select: select

The select command may be executed directly through the SDK's CommandProcessor class, or indirectly through either stcmd or stcmdEx. The syntax of the command is

Syntax

The syntax for this command is:

```
select * | all | access-rights | changes | linked-items | changed-files |
enhanced-links | lifecycle | links | workrecords |
agile-breakdown | scope-change | differences | historical-revisions |
attached-labels |
unlabeled-revisions | merge-counts | attachments |connections-log | duplicate-
shares | missing-artifacts | non-tip-artifacts | behavior | {propertyName,
propertyName,...} | filter = 'filtername'
from 'typeName'  | starteam {history} {deleted} {backlog} {workspace}
{rolldown}
{between-labels labelname1 and labelname2}
{attached-to-label labelname }
{at [label = "label" | promotionstate = "promotion state" | datetime = "date"
{-pattern "pattern"}]
into fileName { separator 'field separator' } {newline newlineSeparator}
{headers 'on' | 'off' | sqlnames} {toExcel}
{-pattern "datetimepattern"} {-locale "localecode"}
where
{{ attached-label = 'labelName' } | { query = 'myquery' } |
propertyName relation value and/or propertyName relation value and/or...}
{for} {folder = 'myfolder' {recurse} or folder = 'myfolderhierarchy'
{recurse} or folder = . {recurse}} or ...}
order by {propertyName, propertyName,...} | orderFilter = 'myOrderFilter'
[–epwdfile “passwordfilepath”] [-p "userName:password@hostName:endpoint/
projectName/
[viewName/][folderHierarchy/]"] | [-s "userName:password@hostName:endpoint]
```

Use the `select` command to invoke StarTeam meta-queries. The combination of options determines the type of query, which could be over a file, folder, change request or etc., and the saved filters for the type. Values that contain spaces should be enclosed in double quotes. This command has been modeled on the standard SQL SELECT syntax. Cross type joins are not supported. If folder identification clauses are not specified, the tool assumes the `folderHierarchy` is set through the `setProject` command or the root folder of the view, with recurse ON (that is, all descendants or depth == -1). The WHERE clause is constrained to a query and a possible set of folders. Folders may be combined with an OR, but cannot be joined with an AND. Folders act as a further constraint to a query. Folders potentially reduce the subset of results obtained from the query to the items that reside within the specified folders. When a folder hierarchy is specified in the WHERE clause of a select, update or delete statement, the path must start with the root folder and traverse the folder tree all the way down to the leaf folder of interest. It must be explicitly terminated by a \. However, the root folder path must not start with a \. / and \ are interchangeable.

Simple dynamic queries support either chained OR clauses or chained AND clauses. However, they do not support a mix of OR and AND conditions. Complex queries are supported but only as saved queries, for example: where query = 'mySavedQueryname'.

If a property name in the where clause identifies a text property, the relation is '=' and the value starts with a "*", then the * is treated as a wildcard targeted for expansion. This query returns all items whose property values end with the text of the query value. For example, the following returns all files in the view whose names end in .doc.

Only * is supported as a wildcard, and can exist contextually at the start of, at the end of or surrounding a phrase. In other words, "*.doc", "doc*", or "*.*" will all expand the * out to mean - any set of characters.

No other wildcard characters are supported. "." itself is treated as a literal character, no different than (say) a, b or c.

Parameter	Description
*** (| all)**	Specifies all defined properties for the type. (This is a superset of propertyName1, propertyName2, ...) `propertyName` specifies the subset of properties for the type. `relation` may be any one of the following: `= < <= > >= <> != in (,)`. Alternative equivalents are `eq, lt, lte, gt, gte, neq`. `in` represents a multi-value set relationship. For example, `changenumber in (1,2,3,4)`. All other relationships are single valued. For example, `changenumber = 1`, `changenumber > 2`, or `changenumber < 5`.
access-rights	Overrides properties. If specified, it generates an access rights report. The columns are the set of available permissions. The rows are the Securable's (or Container's) for which access rights exist.
agile-breakdown	Overrides properties. If specified, it generates an agile breakdown report.

Sprints For sprints, it is a denormalized join or view of all stories, tasks, and workrecords attached to the selected sprints. Each row represents the break down to the workrecord detail. The columns are broken down as:

Columns	Description
1-10	Sprint details
11-30	Story details
31-55	Task
56-60	WorkRecord

Plans For plans, it is a denormalized join or view of all plans and requirements attached to the selected plans. The columns are broken down as:

Columns	Description
1-6	Plan details
7-30	Requirement details

historical-revisions	`historical-revisions` is a report that identifies and lists all the revisions at which the items are actually attached to the specified label. if `historical-revisions` is specified, then the where clause is ignored.
attached-labels	Produces a report combining all historical revisions of the selected item(s) with the labels attached to each of those revisions. The rows of this report match the label tab details in the StarTeam Cross-Platform Client.
unlabelled-revisions	Produces a report which is the converse of `attached-labels`. It lists only those historical revisions of the selected item(s) which are *unlabeled* as of the last build.
changes	Overrides properties. If specified, it generates a change package changes report. The columns identify revision details of each attached item to the change package. The rows identify the attached items. The report can span multiple change packages.
changed-files	Overrides properties. If specified, it generates a software lines-of-code count report. The rows identify the attached files. The columns identify revision details of all attached files to the process item. The report can span multiple process items. Each process item may be linked to multiple revisions of the same artifact (file). Each row provides a count of the

Parameter	Description
	number of lines added, number of lines changed, or number of lines deleted in the context of the checked in revision.
scope-change	Overrides properties. Generates a scope change report that is targeted at measuring the total scope (cost) of the sprint. It includes the history of the sprint including all active/deleted stories associated with the sprint and the history of active stories, with their estimated points.
differences	Overrides properties. If specified, it generates a `differences` report for files (spanning all revisions) between two labels.  **Note:** `differences` can only be used in conjunction with the *file* type. If the `differences` keyword is specified, then `between-labels 'labelname1' and 'labelname2'` must be specified, (and overrides the `where` clause)
linked-items	Overrides properties. If specified, it generates a process item report. The columns identify revision details of all attached items to the process item. The rows identify the attached items. The report can span multiple process items. Each process item may be linked to multiple revisions of the same artifact (file, folder).
enhanced-links	Overrides properties. If specified, it generates an enhanced links report, which provides trace visibility across all views in the project for which trace the queried items shares exist. The columns identify revision details of all attached traces to the queried item. The rows identify the attached queried items. The report can span multiple items.
links	Overrides properties. If specified, it generates a links report. The query follows all the traces which lead to or away from the specified item and describes the details of the items at the other end point.
lifecycle	Overrides properties. If specified, it generates a report that tracks the item through its lifecycle (from creation, through moves and shares, to possible deletion). The rows identify all history and revisions per item, covering item changes, moves and deletes, ordered by modification time - from most recent to oldest. The report can span multiple items. The first column identifies the item by item id.
workrecords	Workrecords can only be specified in conjunction with the *Task* type. When `workrecords` is specified, a `workrecords` report will be produced.
merge-counts	Overrides properties and can only be specified in conjunction with the File type. When specified, the resultant report records a count of the number of times a file has been merged from any other view. The counts are generated per view.
attachments	Produces a report identifying all the attachments by name, ID, size and md5 for each item in the select list.
connections-log	Overrides properties. Generates a connections report. It queries the server log, parses each connection out, and identifies the users who have logged in, logged out, their license types (fixed v/s floating) and the total number of consumed licenses. The connections-log report is a metadata query that requires the type to be 'starteam'.
duplicate-shares	Overrides properties. If specified, it generates an exception report of all items which are shares of each other in the selected view.
missing-artifacts	Overrides properties. If specified, it generates an exception report of all items which are not attached to he label (specified by attached-to-label) in the selected view.
non-tip-artifacts	Overrides properties. If specified, it generates an exception report of all items which are attached to the label (specified by attached-to-label) in the selected view, but not at the tip revision.

Parameter	Description
behavior	Produces a behavior report. This report identifies the branch state and configuration characteristics of the selected items (from the where clause) and matches the StarTeam Cross-Platform Client **Advanced Behavior** dialog box.
Filter	Overrides properties. It identifies a saved filter (by name) for the specified type, and expands into a subset of properties for the type.
typeName	Specifies the item type. Types are mutually exclusive, and include any one of the stock or custom component (if any) types. Type name '`starteam`' is a special case which currently only supports the '`connections-log`' query. If '`starteam`' is the specified type, then -s is sufficient to identify the server; i.e. -p is not required.
at	describes a rolled back view configuration. It may be one of label, promotion state or datetime. If specified, the query is run at the rolled back configuration. datetime may be further qualified by a pattern. The pattern must match any valid pattern supported by the java JDK in `java.text.SimpleDateFormat.applyLocalizedPattern`(String) and ensure that embedded date/ time strings strictly adhere to that pattern. If a pattern is specified, a 2 character country code locale may also be optionally specified.
attached-label	Specifies a label to which the items of the specified type have been attached. The items to be selected are the ones attached to the label.
deleted	Specifies the result set. If specified, only deleted items are returned.
history	Specifies a qualifier on the result set. For example, if the result set contains ten items of a given type, history returns all revisions of each item.
backlog	Can only be specified in the context of the Story type. For example, if the result set contains ten items of a given type, history returns all revisions of each item.
workspace	Acts as a constraint on file queries. If specified, it locates *not-in-view* folders and files from the file system (the view path mapped to working folders on disk) and includes them in the report.
rolldown	Can only be specified for tree item types. When specified, the children of the selected tree items (specifically, the entire tree of descendants) are added to the result set.
Folder	Specifies the StarTeam folder name in the current view. If there are multiple folders with the same name, the command returns all folders with that name. Or specify the folder hierarchy in the "`/`" format. The folder path musts start and end with "`/`". Start from the root folder and end in a branch folder. For example: `/StarDraw/SourceCode/On-line Help/`.
recurse	Designates all descendants from the folder specified.
.	Implies the current working folder, requiring the tool to find StarTeam folders with paths mapping to the current working folder. The Command processor must be running inside the StarTeam folder hierarchy.
into	"fileNameWithPath" specifies a file to which the output of the query will be written. The generated output is tabular in format, each row separated by a carriage return/line feed, specific to the platform the query is run on.

Parameter	Description
newline	If newline separator is specified, then embedded new lines inside text fields are replaced by the provided separator. If not specified, then embedded new lines are replaced by their character string equivalents, specifically, "\r" and/or "\n".
headers	By default, `headers = on`. If `headers = off`, output files are written without headers. A special case of headers turned on is to substitute the property internal names for the display names by using the syntax `sqlnames`.
toExcel	Produces an output file which is in CSV format. The columns are separated by a comma and embedded new lines in text are replaced by line feeds. This is a special format which allows Microsoft Excel to successfully import these types of files directly into a spreadsheet while retaining the tabular structure of the data.
myQuery	Specifies a saved StarTeam query name for the type. The suported property value types are:

Text	Literal string.
Integer	A string in the form of an integer like "1234".
Double	A string in the form of a double like "1234.5678".
Long	A string in the form of a long like "1234567890".
Boolean	The string "true" or "false" - case insensitive.
Date	String format `yyyy-mm-dd`, 4 digit year, 1 <= mm <= 12, 1 <= dd <= 31.
DateTime	If -pattern "pattern" is specified, then it is parsed using `java.text.SimpleDateFormat`, localized pattern set to "pattern". See http://docs.oracle.com/javase/7/docs/api/java/text/SimpleDateFormat.html. If -pattern is not specified, attempt to match patterns using java.text.DateFormat {SHORT, MEDIUM, LONG, FULL} in that order. If all else fails, the engine will attempt ISO8601 parsing e.g.: yyyy-mm-ddThh:mm:ssZ . DateTime patterns work two ways. When specified, they are used to parse input date/time strings in the query itself. Additionally, if datetime properties are output, they are formatted to fit the specified pattern.
TimeSpan	String format [ws][-][d. |d]hh:mm:ss[.ff][ws], items in brackets optional. see `com.starteam.util.TimeSpan`. ws whitespace, d days, ff fractional second, hh hours, mm minutes 0 <= mm <= 59, ss seconds 0 <= ss <= 59.
Enumerated	String. Enumerated value specified may be internal name, display name, or string representation of integer enumeration code. If the Enumerated property is multi-selectable, two or more enums may be specified. In this case, they must be separated by a period. For example: 101.102.103.
Object	**User** — Value specified may be user name or string representation of integer user id. **Group** — Value specified may be group name or string representation of integer group id. **Label** — Value specified may be label name or string representation of integer label id.

Parameter	Description
	LinkValue A string in the form of an integer like "1234" which represents the viewmemberid of the source or target link of a trace.
order by	Specifies a default sort order for the output result set.
OrderFilter	Specifies the saved StarTeam filter name for the type. It expands into a subset of properties for the type, which is used for sorting. `myFilter` and `myOrderFilter` can be different. The properties specified in `MyOrderFilter` should not be set for grouping. If a set of property names is specified instead of the order filter, the sort criteria default to ascending order, sort by text is set for text properties, and sort by date is set for Date/DateTime Properties. If you need more specific sorting, specify an existing saved filter.

-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.
	Note: If the clear text password contains the @ or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The @ or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.
	The full syntax is:
	`stcmd -p "userName:password@hostName:endpoint/projectName/[viewName/][folderHierarchy/]"`
	For example:
	`stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/SourceCode/”`
	• If the user name is omitted, the current user name is used.
	• If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
	• If the host name is omitted, the default is localhost.
	• Entering an endpoint (port number) is required. The default is 1024.
	• The project name is always required.
	• A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the

Parameter	Description
	view name. Doing this is not recommended, however, because another view with that name could be created at a later date. • A folder hierarchy should be used to identify the folder. Use the forward slash (/) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view's root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .
	The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.
	Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.
	In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.
	The following is the syntax of the commands that can be used to store an encrypted password.
	Use the following syntax to be prompted for the password that will be encrypted and stored in a file.
	`stcmd store-password -epwdfile "filePath"`
	Use the following syntax to include the encrypted password in the command as clear text.
	Note: This action does not access the network with the clear value.
	`stcmd store-password -epwdfile "filePath" -password "password"`
	After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:
	`stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"`
	Important:
	If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:
	`-p user@hostname:port/projectName.viewName -epwdfile "pathToPasswordFile"`

-cmp	Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place. Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data. This is an optional parameter. If not specified, then the platform default is not to compress.

Parameter	Description
-csf	When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder. The default is that StarTeam folders are not differentiated based on the case of letters in their names. With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.
-encrypt	Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks. This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server. The full syntax is: `-encrypt encryptionType`. The types of encryption are: **RC4** RSA RC4 stream cipher (fast). **RC2_ECB** RSA RC2 block cipher (Electronic Codebook). **RC2_CBC** RSA RC2 block cipher (Cipher Block Chaining). These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it. **Note:** For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.
-s	Identifies the StarTeam Server. The full syntax is: `-s "userName:password@host:portNumber"` For example: `-s "JMarsh:password@orion:49201"` If the user name is omitted, the current user name is used. The user name in the example is "`JMarsh`". If the password is omitted, the user is prompted to enter the password. The password in the example is "`password`". If the host name is omitted, the default is `localhost`. The host name in the example is "`orion`". The port number is required. The default port number, `49201`, is used in the example.

In general, the select grammar is positional. It is best to follow the position of the syntax as outlined in the command description above.

Note: For the Change Request type, the AddressedInBuild property can be used in the where clause of a query using the special value "Next Build" or "-2".

Important: Note for use with stcmdEx. Enclose the greater than ">" and less than "<" characters within double quotes. By design, stcmdEx interprets the > as a redirect to a file so, without "" you get an error "The system cannot find the file specified".

Examples

```
select * from File where query = "Status = Current" order by
orderfilter = "All Files By Status" -p
"Administrator:Administrator@localhost:49201/StarDraw/Release 1.0
Maintenance"
```

The following example shows how to use the differences keyword. It selects all properties of all change requests and writes them into a file called QueryOutput.txt.

```
select differences from file between-labels 14.0.3.21 and 14.0.3.27
into "c:/temp/differencesReport.txt" separator | -p
"username:password@hostname:port/project/view"
```

```
select * from changerequest into "c:/temp/QueryOutput.txt"
```

```
select linked-items from ChangeRequest where AddressedIn = "Next Build"
or
select linked-items from ChangeRequest where AddressedIn = "-2"
```

The example below selects three properties, Name, Status, and File Time Stamp at check in, for all files, which satisfies the built in query Files to Checkin.

```
select Name, Status, Modified from file where query = "Files to CheckIn"
```

The example below selects all tasks from the Sales Materials folder or the Marketing Materials folder and its descendants. It returns a result set containing only the task properties described by the "By Status and Responsibility" filter. select * from task where filter = "By Status & Responsibility"

```
select filter = "By Status and Responsibility" from task where folder =
"Sales Materials" or folder = "Marketing Materials" recurse
```

The following examples show how to use select with change requests and change packages.

```
select linked-items from ChangeRequest into fullyQualifiedPathToOutputFile
where ChangeNumber = 1234 -p "username:password@host:port/project/view"
```

```
select changes from ChangePackage where name = "Workspace Changes on
2013-10-15@22-43-00Z"
```

This example shows how to use the select command with the lifecycle parameter.

```
select lifecycle from File into fullyQualifiedPathToOutputFile
where FileName = Server.java -p "username:password@host:port/project/view"
```

This example shows how to generate a workrecords report:

```
select workrecords from task where StTaskNumber = 88
```

The following example shows how to user the select command with the rolldown parameter:

```
stcmd (ex)   select {propertyname,(s)…} from task rolldown where {…} –p “…”
```

This examples shows how to generate an agile-breakdown report:

```
select agile-breakdown from sprint into c:/temp/agileReport.txt
separator "|" where SprintID = 1234 -p "user:password@host:port/project/view"
```

The following example shows you how to select rows in a range:

```
select "*" from changerequest into "c:/temp/stout.txt" where
changenumber ">" 50000 and changenumber "<" 50100 -p
"user:password@host:port/project/view"
```

The next example uses a date range on the ModifiedTime property:

```
select "*" from ChangeRequest into "c:/temp/stout.txt" pattern =
"M/d/y" where modifiedtime ">" 1/1/2014 and modifiedtime "<"
8/30/2014 -p "user:pwd@host:port/project/view"
```

The following example produces history results similar to the deprecated hist command:

```
select revisionnumber, viewid, modifieduserid, modifiedtime,
comment from file history
-p "Username:Password@host:port/project/view/[folderhioerarchy]/"
where name = "filename"
```

The next example shows the use of historical-revisions

```
select historical-revisions from file attached-to-label "label
name" -p "user:password@host:port/project/view"
```

The next two examples show the use of wildcards and fully qualified folder paths

```
select folder, name from file where name = "*.doc" for folder =
"StarDraw\Source Code\External Resources\"
```

```
select viewmemberid, name, description from file where name =
"*.doc"
```

The next few examples show different usage of property relation values:

```
select * from ChangeRequest pattern = "M/d/y" where ModifiedTime
> 12/14/1999 and modifiedtime < "2/16/1999"
```

```
select * from ChangeRequest where ChangeNumber in (10298, 10310,
10316, 10320)");
```

```
select * from ChangeRequest where ChangeNumber = 10298
select * from ChangeRequest where ChangeNumber ">=" 10298 and
ChangeNumber "<=" 10320
select * from File where name = "GNUmakefile.build*"
select * from File where name = "*.buildinfo"
select * from File where name = "*ranching release
views.doc*"
select * from ChangeRequest where Responsibility = "Alan
Kucheck"
```

Note that ">=" can be replaced by gte without the double quotes around it,

= can be replaced by eq, etc.

This example shows the use of date patterns driving the format of output date time property values:

```
select name, modifiedtime from file -pattern "yyyyMMddHHmmss"
where name = "*.java" order by name -p
"user:pwd@host:port/project/view"
```

This example queries the path of all files which are exclusively locked by a user:

```
select Path from file where exclusivelocker <> ""
```

```
select connections-log from starteam into
"c:/temp/connections.log" -s "user:password@host:port"
```

This example reports on custom properties, and uses them to drive the query.

```
select changenumber, usr_projecttype, usr_dateclosed from
changerequest into custom.cr.output.txt headers off pattern =
```

```
"M/d/y" where USR_DATECLOSED gte 1/1/2015 and USR_DATECLOSED lt
1/1/2016
```

This same query with greater precision:

```
select changenumber, usr_projecttype, usr_dateclosed from
changerequest into custom.cr.output.txt headers off pattern =
"M/d/y H:m:s" where USR_DATECLOSED gte "1/1/2015 0:0:0" and
USR_DATECLOSED lt "12/31/2016 23:59:59"
```

Set Personal Options: set-personal-options

Use `set-personal-options` to set and list parameter values. Command options are saved in the StarTeam Cross-Platform Client `Options` file. Any option used by subsequent commands are read from the file `Options` specified in the command override options within StarTeam Cross-Platform Client `Options`.

Conversely, if the option is not specified on the command, but found in StarTeam Cross-Platform Client Options, then that option value is used.

Options are specified using the form `key=value`. If only the key is specified, then the option is assumed to have the default value `true`.

Syntax

The syntax for this command is:

```
stcmd{Ex} set-personal-options parameter = value
```

Parameter	Description
OptionName	A parameter such as `-pf`. When saved to the personal options file, the `-` is stripped off from the `-pf`.
OptionValue	A default value assigned to the option.

Set Project and View: set

After connecting to the server using the `connect` command, use the `set` command to designate the project and view.

Note: This command is part of the stateful model and only works with `stcmd`, not `stcmdEx`.

Syntax

The syntax for this command is:

```
stcmd set project=projectName
[view=viewName | viewHierarchy=viewName:viewName:…]
[folderHierarchy=folderName/folderName/....]
[-cfgl "label" | -cfgp "promotion state" | -cfgd "date"]
[-pattern "date-pattern"]
```

Parameter	Description
project	Specifies the project name.
view	Specifies the view name.
viewHierarchy	Specifies the view hierarchy. Use ":" to separate view name within hierarchy.

Parameter	Description
folderHierarchy	Specifies the folder hierarchy. Use "/" to separate each folder name within the hierarchy. Start with the root folder and end with the branch folder.
-cfgl	Configures the view using the specified label. Without `-cfgl`, `-cfgp`, or `-cfgd`, the view's current configuration is used.
-cfgp	Configures the view using the specified promotion state.
-cfgd	Configures the view as of the specified date/time. Examples include: "12/29/13 10:52 AM" "December 29, 2013 10:52:00 AM PST" "Monday, December 29, 2013 10:52:00 AM PST"
-pattern	Qualifies the datetime. It can be specified wherever a date-time is specified, such as `-cfgd`, `-vd`, etc. The pattern must match any valid pattern supported by the java JDK in `java.text.SimpleDateFormat.applyLocalizedPattern(String)`. The pattern may be localized. For every command that takes a `-pattern` parameter, a `-locale` parameter is optionally available. This is the "two character country code".

Example

The following example connects to the server using the `connect` command and designates the project of `StarDraw` and the view of `StarDraw` using the `set` command. After connecting and designating project and view, execute all command for that project and view without reconnecting each time. When command are complete, use the `disconnect` command to disconnect from the project, view, and server. Alternatively, use the `set` command to switch to a different project/view without disconnecting. You can then execute all command in the new project/view context.

```
stcmd connect "JMarsh:password@Orion:1024"
stcmd set project = 'StarDraw' view ='StarDraw'
```

Share: share

Use share to share any StarTeam artifact from one view to another view, from a folder to another folder in the same view, and etc.

The share command returns the item ID of the newly created share.

Syntax

The syntax for the command is:

```
share –p “user:pwd@host:port/project/view” [ -epwdile “pathToPasswordFile” ]
[ -csf ] –type “typeName” –id itemID [ –tv “targetViewName” ] [ -tfp “target
folder path” ] [ -boc  true|false ] [ -vctip | -vclbl labelid | -vcps
promotionstateid | -vcdttm “datetimestring” –pattern “datetimepattern” ]
```

Parameter	Description
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be

Parameter	Description

migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.

Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.
- A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.
- A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view’s root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile

The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .

The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.

Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.

In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.

Parameter	Description

The following is the syntax of the commands that can be used to store an encrypted password.

Use the following syntax to be prompted for the password that will be encrypted and stored in a file.

```
stcmd store-password -epwdfile "filePath"
```

Use the following syntax to include the encrypted password in the command as clear text.

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password "password"
```

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

This is an optional parameter. If not specified, then the platform default is not to compress.

-csf

When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder.

The default is that StarTeam folders are not differentiated based on the case of letters in their names.

With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.

-encrypt

Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks.

This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server.

The full syntax is: `-encrypt encryptionType`.

The types of encryption are:

RC4	RSA RC4 stream cipher (fast).

Parameter	Description
	RC2_ECB RSA RC2 block cipher (Electronic Codebook).
	RC2_CBC RSA RC2 block cipher (Cipher Block Chaining).

These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.

Note: For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.

-type	Specifies the type of item. The type is one of the stock type names, such as `changerequest`, `task`, `requirement`, `sprint`, `story`, `plan` or any custom type name that is applicable to the command.
-id	Specifies the unique item (view member) ID of the item. Look in the property lists of the CPC or query using the select command to find the View Member IDs. `-id` can also specify the primary descriptor of the item; e.g. file name, folder name, change request number.
-tv	Describes an optional specific target view into which to move. If not specified (and `-tp` is not specified), then the item will be moved into the source view described by `-p`.
-tfp	Describes the folder path to which to move. If a folder name is specified, then this name must be unique across the folder tree. Otherwise, a fully qualified folder path is required, starting from the root folder of the view down to the leaf folder into which the move must be created. If `-tfp` is not specified, then the move will be created in the root folder of the target view `-tv`.
-tp	Describes an optional specific target project into which to move. If not specified, then the item will be moved into the source view described by `-tv`, assuming that is specified. If neither is specified, then the move will occur into the original view as specified by `-p`. If `-tp` is specified, and `-tv` is not, then the move will occur into the default (root) view of the project described by `-tp`.

The remaining parameters are optional. If not specified, they default to the view level behavior.

Parameter	Description
-boc	Set to `true` for branch on change ON, `false` for branch on change OFF.
-vctip	Specifies that the new share floats.
-vclabel	Specifies that the new share is pinned to the specified label. A label is identified by either the unique integer ID or the label name.
-vcps	Specifies that the new share is pinned to the specified promotion state. A promotion state is identified by either the unique integer promotion state ID or the promotion state name.
-vcdttm	Specifies that the new share is pinned to the specified point in time. An optional `-pattern` describes the syntax of the date time string, just like in other commands.

Store Password: store-password

Use the `store-password` command to store the password in an encrypted file.

Syntax

The syntax for this command is:

```
stcmd{Ex} store-password –password “password” -epwdfile “passwordFilePath”
```

Parameter	Description
-password	Specifies the password.

-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .
	
	The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.
	
	Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.
	
	In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.
	
	The following is the syntax of the commands that can be used to store an encrypted password.
	
	Use the following syntax to be prompted for the password that will be encrypted and stored in a file.
	`stcmd store-password -epwdfile "filePath"`
	
	Use the following syntax to include the encrypted password in the command as clear text.
	
	Note: This action does not access the network with the clear value.
	`stcmd store-password -epwdfile "filePath" -password "password"`
	
	After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:
	`stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"`
	
	Important:
	
	If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:
	`-p user@hostname:port/projectName.viewName -epwdfile "pathToPasswordFile"`

Synchronize: sync

Use the `sync` command to synchronize changes on your local hard disk (working folders or workspace) to your view and vice versa. For instance, if you deleted a file or a folder on disk, `sync` deletes that file or

folder from the view. If you have a new file or not-in-view folder (with files) on your local hard disk, `sync` adds them to the view. `sync` does not synchronize views with each other.

`sync` supports the same syntax as the `ci` command (check in) for process items and reasons for checking in files. If the project requires a reason for adding/checking in files, specify one with `-d` or `-r`. If the project requires a process item, specify one with `-cr`, `-req` or `-task`. You cannot use `-p` with `sync`. It is one of the new class of stateful commands.

Syntax

The syntax for this command is:

```
stcmd{Ex} sync workspace|view|both [-o] [-nivf][-q] [-cmp]
[-l | -u | -nel] [-vl "labelName"]
[-nomove] [-d|-r "comment"  | -rf "fileName"]
[[-active | [-cr | -req | -task] processItemPath] -pi typename:"path" [-mark]]
```

Parameter	Description
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.

Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp \pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.
- A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the

Parameter	Description
	view name. Doing this is not recommended, however, because another view with that name could be created at a later date. • A folder hierarchy should be used to identify the folder. Use the forward slash (/) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view's root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .
	
	The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.
	
	Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.
	
	In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.
	
	The following is the syntax of the commands that can be used to store an encrypted password.
	
	Use the following syntax to be prompted for the password that will be encrypted and stored in a file.
	`stcmd store-password -epwdfile "filePath"`
	
	Use the following syntax to include the encrypted password in the command as clear text.
	
	Note: This action does not access the network with the clear value.
	
	`stcmd store-password -epwdfile "filePath" -password "password"`
	
	After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:
	`stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"`
	
	Important:
	If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:
	`-p user@hostname:port/projectName.viewName -epwdfile "pathToPasswordFile"`

-o	Forces a check-in/check-out depending on which command is used. `-o` is supported with `-filter` and `-f NCD`, but not with `-f NCO`.
-nivf	If `-nivf` is included, then files in `Not in View` folders are also included in the action.
-q	Enables quiet mode. The `-q` option is retained for backward compatibility with the old command line. If `-q` is specified, then `-pf` cannot be specified. The command will return no results.

Parameter	Description
-cmp	Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place. Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data. This is an optional parameter. If not specified, then the platform default is not to compress.
-csf	When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder. The default is that StarTeam folders are not differentiated based on the case of letters in their names. With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.
-encrypt	Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks. This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server. The full syntax is: `-encrypt encryptionType`. The types of encryption are: **RC4** RSA RC4 stream cipher (fast). **RC2_ECB** RSA RC2 block cipher (Electronic Codebook). **RC2_CBC** RSA RC2 block cipher (Cipher Block Chaining). These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.  **Note:** For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.
-l | -u | -nel	Locks the item(s). `-l` is exclusive lock, `-u` is unlocked, and `-nel` is non exclusive lock. These items are mutually exclusive and an optional parameter.
-vl	Specifies a label (created using `stcmd label`) to be applied to the checked-in files. The label is enclosed in double quotation marks. This option can appear in the command more than once. The label can be either a view or revision label, but it must already exist in the application.

Parameter	Description
-nomove	Do not move labels if already attached.
-d | -r	Description or reason for check-in. If `-d` or `-r` is used, `-rf` cannot be used. The reason should be enclosed in double quotation marks and should not exceed 254 characters in length.
-rf	Specifies the file name that contains the check-in reason. This is useful if the same reason should be applied to all check-ins across multiple command line runs.
-active	The active process item.
-cr, -req, -task	Complete path from the project view's root folder to the change request, requirement, or task number to be used as a process item. Use the forward slash (`/`) as a delimiter between folder names. For out-of-view process items, specify the project name and view name in front of the complete folder path. For example: `-cr MyProject/RootView/RootFolder/SourceCode/37` This specifies change request `37` in the `SourceCode` folder (under the root folder) of the `ChildView` view in the `MyProject` project. **Note:** For in-view process items, as long as the change request, requirement, or task numbers are the unique primary descriptors of their types (true by default), it is sufficient simply to specify the number, with no path. The project and view names are assumed from `-p`. If a process item is specified, then the files being checked in are attached to the process item and follow the project process rules. `-cr`, `-req` or `-task` are mutually exclusive. If any one of them is specified, `-filter/-f` are ignored.
-pi	The type name is the internal name of a component that supports being a process item type. for example, change request, task, requirement, story, or custom component name. For a type to be a process item type, it must have a status property for check in. The syntax to use is `-pi typename:"path"`.
-mark	Indicates that, if all the files are successfully added, the process item's status will be changed to fixed (for a change request), finished (for a task), or complete (for a requirement). The files are pinned to the revision with the new status. The item is not marked as fixed, finished, or complete unless all the files are successfully added.

Example

The following is a simple `sync` example:

```
stcmd connect "User:PW@server:port"
stcmd set project=projectname view=viewname
stcmd sync {view | workspace | both} -nivf
```

When you specify view, `sync` applies to the view all the changes you have made to your workspace (that is, working folders on disk). When you specify workspace, `sync` applies to your workspace all the changes made to your view. When you specify both, it does both of the above.

Trace: trace

A trace is a link between any two StarTeam items. It expresses a join relationship. Use the `trace` command to create or to find and update a trace in a project/view described by the `-p` parameter (or preceding `connect/set` commands). A trace will only be created if its endpoints are guaranteed to exist.

Syntax

The syntax for this command is:

```
stcmd{Ex} trace [-id traceID]
[-sourceType sourceTypeName -sourceID sourceItemID
[-sourceView sourceViewName]]
[-targetType targetTypeName -targetID targetItemID
[-targetView targetViewName]]
[-pinSource] [-pinTarget] [-suspect]
[-traceTypeID nnn]
[-p "user:password@host:port/project/view"]
```

Parameter	Description
-id traceID	Optional. If specified, an existing trace (identified by `traceID`) is updated, otherwise, a new trace is created.
[-sourceType sourceTypeName -sourceID sourceItemID [-sourceView sourceViewName]]	Optional. If specified, the source endpoint is set to the item which is identified by `sourceItemID`, `sourceTypeName` (e.g. changerequest, task, story, sprint, etc) and source view name. Source view name is optional. If not specified, the source item must resolve to an item in the same view as the trace.
[-targetType targetTypeName -targetID targetItemID [-targetView targetViewName]]	Optional. If specified, the target endpoint is set to the item which is identified by `targetItemID`, `targetTypeName` (e.g. changerequest, task, story, sprint, etc) and target view name. Target view name itself is optional. If not specified, the target item must resolve to an item in the same view as the trace.
[-pinSource]	Optional. The default behavior is for the trace to float at the source end. If specified, the trace is pinned to the source tip.
[-pinTarget]	Optional. The default behavior is for the trace to float at the target end. If specified, the trace is pinned to the target tip.
[-suspect]	Optional. The default behavior is for a trace not to be marked suspect. If specified, the trace is marked as suspect (it's suspect flag is set to `true`).
[-traceTypeID]	If specified, the numeric id must be a valid enum value from the set of trace types: • `UNDEFINED 0` • `REQUIREMENT_DEPENDENCY 1` • `TASK_DEPENDENCY 2` • `WORK_ITEM 3` • `RESOLUTION_POINT 4` • `VALIDATION_TEST 5` • `MANUAL 6` • `PROCESS_ITEM 7` • `COPY 8` • `BREAKDOWN 9`

Parameter	Description
	• EXECUTION 10 • DEFECT_STREAM 11 • REQUIREMENT_STREAM 12 • BACKLOG_STREAM 13 • CARRY_OVER 14 • WHITEBOARD 15 • DEFINE 16 • REQUIREMENT_SYNC 17

-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.
	
	Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.
	
	The full syntax is:
	`stcmd -p "userName:password@hostName:endpoint/projectName/[viewName/][folderHierarchy/]"`
	
	For example:
	`stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/SourceCode/”`
	
	• If the user name is omitted, the current user name is used.
	• If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
	• If the host name is omitted, the default is localhost.
	• Entering an endpoint (port number) is required. The default is 1024.
	• The project name is always required.
	• A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.

Parameter	Description
	• A folder hierarchy should be used to identify the folder. Use the forward slash (/) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view's root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .
	
	The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.
	
	Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.
	
	In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.
	
	The following is the syntax of the commands that can be used to store an encrypted password.
	
	Use the following syntax to be prompted for the password that will be encrypted and stored in a file.
	`stcmd store-password -epwdfile "filePath"`
	
	Use the following syntax to include the encrypted password in the command as clear text.
	
	Note: This action does not access the network with the clear value.
	
	`stcmd store-password -epwdfile "filePath" -password "password"`
	
	After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"'` as parameters. For example:
	`stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"`
	
	Important:
	
	If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:
	`-p user@hostname:port/projectName.viewName -epwdfile "pathToPasswordFile"`

-cmp	Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place. Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

Parameter	Description
	This is an optional parameter. If not specified, then the platform default is not to compress.
-csf	When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder. The default is that StarTeam folders are not differentiated based on the case of letters in their names. With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.
-encrypt	Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks. This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server. The full syntax is: `-encrypt encryptionType`. The types of encryption are: **RC4** RSA RC4 stream cipher (fast). **RC2_ECB** RSA RC2 block cipher (Electronic Codebook). **RC2_CBC** RSA RC2 block cipher (Cipher Block Chaining). These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it. **Note:** For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.

Example

The following will create a trace in the `bar` view, source endpoint `ChangeRequest (ID 1234)` also from the `bar` view, target endpoint `Story (ID 5678)` from the `foo` view, source floating, target pinned:

```
stcmd trace -sourceType changerequest -sourceID
1234 -targetType story -targetID 5678 -targetView "foo"
-pinTarget -p "user:pwd@host:port/project/bar"
```

Transfer Traces: transfer-traces

The transfer-traces command is used to convert a multi-configuration environment into a Federated Tracing solution.

It requires each configuration to have the `FederatedTraceServer` config option to reference the GUID of the one configuration which will store all the traces.

This command should be run once for each participating configuration, the target always pointing to the federated trace configuration. The command moves each cross configuration trace it discovers over to the federated configuration.

Syntax

The syntax for this command is:

```
stcmd transfer-traces -s "Source Server" -t "Target Server" [-q] [-ofp
outputfilepath}
```

Parameter	Description
-s	Identifies the StarTeam Server. The full syntax is: `-s "userName:password@host:portNumber"` For example: `-s "JMarsh:password@orion:49201"` If the user name is omitted, the current user name is used. The user name in the example is "`JMarsh`". If the password is omitted, the user is prompted to enter the password. The password in the example is "`password`". If the host name is omitted, the default is `localhost`. The host name in the example is "`orion`". The port number is required. The default port number, `49201`, is used in the example.
-t	Represents the federated trace server to be written to.
-q	Enables quiet mode. The `-q` option is retained for backward compatibility with the old command line. If `-q` is specified, then `-pf` cannot be specified. The command will return no results.
-ofp	Provides a file name with a fully qualified path into which to write the command output. By default, a `"|"` character separates each column in the output. A new line separates each row. The first row is the command name. The second row has the property names. All subsequent rows contain the data. If the file already exists, the output is appended to the end of the file. It is possible to override the `"|"` character separator by specifying `separator = fieldSeparator` as a parameter to the connect command. For example, `separator = ;;` specifies two adjacent semicolons (`;`) as the column separator.

-q and -ofp are mutually exclusive.

Example

```
stcmd transfer-traces -s "Administrator:Administrator@bel-151traces:49201" -t
"bel-151traces:49203" [-q] [-ofp c:/temp/transferred.txt}
```

Update: update

Use `update` to update all items that satisfy the `where` clause. The syntax of the where clause is identical for select, update and delete, and is fully described in the `select` command section. Values that contain spaces should be enclosed in double quotes. This command has been modeled on the standard SQL `update` syntax.

Note: Only user modifiable properties can be specified for a value update. Run the describe type command to identify the set of user modifiable properties.

Syntax

The syntax for this command is:

```
stcmd{Ex} update type
[{set
propertyName = value,
propertyName = value,
where {{ attached-label = 'labelName' } | { query = 'myquery' }
| propertyName relation value and/or propertyName relation value and/or...}
{{for} {folder = 'myfolder' {recurse} or folder = 'myfolderhierarchy'
{recurse} or folder = . {recurse} or ...}} |
{ ( propertyName1, propertyName2, . . . propertyNamen ) } from fileName
{ join propertyName }] {output* | {propertyName,...} |
filter='myFilter' into "outputFilePath"
{ separator ‘fieldSeparator’} {-pattern "pattern"}}
[–epwdfile “passwordfilepat”]
[-p "userName:password@hostName:endpoint/projectName/[viewName/]
[folderHierarchy/]"]
```

Relation in {=, <, <=, >, >=, <>, !=}.

As an alternative to the `set … where …` syntax, you can use the `(propertyName 1..n ) from filename { join propertyName }` syntax. This is useful for updating types with values from a comma separated file on disk (see the Examples below).

A special null value is recognized as an allowed value of an Enumerated Property This permits the syntax:

```
update ChangeRequest set status = null where status = New -p
"user:pwd@host:port/project/view"
(In this example, an assigned status is unassigned).
```

```
update ChangeRequest set status = New where status = null -p
"user:pwd@host:port/project/view"
(In this example, an unassigned status is re-assigned).
```

Parameter	Description
myFilter	Specifies a filter by name, whose properties are written to the output file.
myFolder	Specifies the StarTeam folder name in the current view. If there are multiple folders with the same name, the command performs the action on all folders with that name.
myFolderHierarchy	Specifies the folder hierarchy in the "`/`" format. Start from the root folder and end in a branch folder. For example: `/StarDraw/SourceCode/On-line Help/`.
output	Turns on logging of the command to a log file specified by `INTO`.

Parameter	Description
	The `INSERT`, `DELETE`, and `UPDATE` commands log the selected properties of the inserted items to a log file. The property values are separated by the specified `fieldSeparator`, or `"|"` if a separator is not specified.
recurse	Designates all descendants from the folder specified.
.	Implies the current working folder, requiring the tool to find StarTeam folders with paths mapping to the current working folder. The Command processor must be running inside the StarTeam folder hierarchy.
query='myQuery'	Specifies the saved StarTeam query name for the type. It acts as the equivalent of a compound where clause of a SQL statement, such as combinations of relations and operators. If no query name is specified, the command performs the action on all objects of the type.
propertyName	Specifies the subset of properties for the type.
type	Specifies the StarTeam item type by name. Types are mutually exclusive.
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.

Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.

Parameter	Description
	• A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date. • A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view's root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .
	
	The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.
	
	Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.
	
	In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.
	
	The following is the syntax of the commands that can be used to store an encrypted password.
	
	Use the following syntax to be prompted for the password that will be encrypted and stored in a file.
	`stcmd store-password -epwdfile "filePath"`
	
	Use the following syntax to include the encrypted password in the command as clear text.
	
	Note: This action does not access the network with the clear value.
	
	`stcmd store-password -epwdfile "filePath" -password "password"`
	
	After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:
	`stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"`
	
	Important:
	If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:
	`-p user@hostname:port/projectName.viewName -epwdfile "pathToPasswordFile"`

Parameter	Description
-cmp	Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place. Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data. This is an optional parameter. If not specified, then the platform default is not to compress.
-csf	When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder. The default is that StarTeam folders are not differentiated based on the case of letters in their names. With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.
-encrypt	Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks. This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server. The full syntax is: `-encrypt encryptionType`. The types of encryption are: **RC4** RSA RC4 stream cipher (fast). **RC2_ECB** RSA RC2 block cipher (Electronic Codebook). **RC2_CBC** RSA RC2 block cipher (Cipher Block Chaining). These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it. **Note:** For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.

Property Values

The following are the property values for the command:

Property Type	Value
Text	Literal string.
Integer	A string in the form of an integer like `"1234"`.
Double	A string in the form of a double like `"1234.5678"`.
Long	A string in the form of a long like `"1234567890"`.
Boolean	The string `"true"` or `"false"` - case insensitive.
Date	String format `yyyy-mm-dd`, 4 digit year, `1 <= mm <= 12`, `1 <= dd <= 31`.
DateTime	If `-pattern "pattern"` is specified, then it is parsed using `java.text.SimpleDateFormat`, localized pattern set to `"pattern"`. See *http://docs.oracle.com/javase/7/docs/api/java/text/SimpleDateFormat.html*. If -pattern is not specified, attempt to match patterns using `java.text.DateFormat {SHORT, MEDIUM, LONG, FULL}` in that order. See *http://docs.oracle.com/javase/7/docs/api/java/text/DateFormat.html*. If all else fails, try ISO8601 parsing e.g.: `yyyy-mm-ddThh:mm:ssZ` (ignore fractional content after seconds).
TimeSpan	String format `[ws][-][d. |d]hh:mm:ss[.ff][ws]`, items in brackets optional. See `com.starteam.util.TimeSpan`. `ws` whitespace, `d` days, `ff` fractional second, `hh` hours, `mm` minutes `0 <= mm <= 59`, `ss` seconds `0 <= ss <= 59`.

Enumerated	String. Enumerated value specified may be internal name, display name, or string representation of integer enumeration code. If the `Enumerated` property is multi-selectable, two or more enums may be specified as values. In this case, they must be separated by a period. For example: `101.102.103`. Here are some examples:
	`stcmd insert into story (name, tag) values ("This is a story name", 101.103)`
	`stcmd update story set tag = 102.103 where viewmemberid = 1234`

Object	**User string** — Value specified may be user name or string representation of integer user id. **Group string** — Value specified may be group name or string representation of integer group id. **Label string** — Value specified may be label name or string representation of integer label id. **LinkValue** — A string in the form of an integer like "1234" which represents the viewmemberid of the source or target link of a trace.

Note: The update command can be used to assign a new revision comment to the tip revision of a selected set of items of a given type using the special property keyword `revisionComment`. `revisionComment` should not be used in conjunction with any other property updates. For example:

```
stcmd update changeRequest set revisionComment = "Now is the time for all good men" where query = "Status = Open" -p …
```

Simple Example

The following example sets the synopsis to the value `foo` for all change requests with an `Open` status.

```
stcmd connect localhost:49201 // OR
stcmd connect localhost:49201 // attempts an autologon via the toolbar & cached credentials
stcmd set project = ‘StarDraw’ view = ‘StarDraw’
```

```
stcmd update changerequest set synopsis = "foo" where query = "Status = Open"
disconnect
```

Example Using a Text File for Changes

The following example will update the set of all change requests in the file `crsForUpdate.txt`, properties as specified in the comma separated list, values in the file spread across several lines, 1 per change request, order of the values matching the order of the properties in the command syntax.

```
stcmd update ChangeRequest (ChangeNumber, Synopsis, usr_SomeText, Component)
from c:\\somepath\\crsForUpdate.txt separator ,
-p “Administrator:Administrator@localhost:49201/TestUpdate”
```

If the property names are not specified in the command syntax, they must be specified as the first line of the file. The default separator for the command line is the `|` symbol. Command authors can override the separator by providing a different separator in the syntax, e.g. separator `,` meaning, the file uses comma as the property value separator.

One and only one column in the file must be the column used to match each row to an item in StarTeam. This match will be made on the primary descriptor (e.g. CR Number) or the `viewmemberID` (the default), provided they are also specified in the file.

If neither property is found, then the command author must specify the join `propertyName` in the syntax. In this case, the join property column is expected to be in the file.

The file content (property values) must match the order of the properties specified in the update syntax, and separated by a `,`. Each change request must be on a separate line.

The property `ChangeNumber` is the *key* that matches the data in the file to change requests in the server. It can behave as a key because it is a read-only property, and it is also the primary descriptor for the change request type. You could equally well have used `ViewMemberID` as the key, specified that instead of `ChangeNumber`, and provided view member id's in the file.

Properties `Synopsis` and `Component` are user modifiable text properties.

Property `usr_SomeText` is a custom property added to the type by the customer (that's why it's name starts with `usr_`).

By definition, all custom properties on a type are user modifiable.

Type and property names are case insensitive and all command line syntax is expected to be in lower case however, project, view, folder, file names may or may not be case sensitive, depending upon the platform.

Matching the example above, the file `crsForUpdate.txt` would have the format:

```
1234,"this is a 1234 foo","special text for the custom property
usr_SomeText","software component" 5678,"this is a 5678 bar","other text,
that is relevant, to a bar","hardware component"
```

From the example, `1234` & `5678` are change numbers for which change requests must already exist in the server, in the selected project and view.

If you specified a different separator (for example: `|`), then the data in the file would be separated by `|`.

```
1234|"this is a 1234 foo"|"special text for the custom property
usr_SomeText"|"software component"
```

You may specify any consecutive data values in the file, provided that the order and type of the data values match the order and type of the property names specified in the command, and that these properties are user modifiable.

Update File Status: update-status

When you update the status of a file, StarTeam compares the working file with the revision you checked out and the tip revision. For example, your **File** list may say that the file is `Current`, but someone else has just checked in a copy of it, so the status of your file is actually is `Out Of Date`.

Updating file statuses is not the same as updating files. If a file is not in your working folder, updating the status lets you know that the file's status is `Missing`, but will not check out the file for you. Normally, you update file status to determine whether a file should be checked in, checked out, added, or ignored.

For example, you may want to:

- Check in a file if its status is `Out Of Date`, `Missing`, or `Merge`.
- Check out a file if its status is `Modified` or `Merge`.
- Add a file to the application if its status is `Not In View`. However, the `update-status` command never lists files that have the status `Not In View` because they are not stored in the repository.

Use `stcmd{Ex} update-status` to display the filename, its status before the command, and its status after the command. A sample line of output might be:

```
x.cpp: status is Current (was Unknown)
```

Syntax

The syntax for this command is:

```
stcmd{Ex} update-status [-p "projectSpecifier" [-epwdfile "filePath"]
[-cmp] [-csf] [-encrypt encryptionType]][-is]
[-vb] [-rp "folderPath" | -fp "folderPath"] [-filter "fileStatus"]
[-cfgl "labelName" | -cfgp "stateName" | -cfgd "asOfDate"]
[-pattern "pattern"] [-q|-pf “filterName”]
[-ofp “resultsOutputFilePath”] [files...]
```

Parameter	Description
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.

Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

Parameter	Description

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.
- A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.
- A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view’s root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

-epwdfile

The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .

The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.

Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.

In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.

The following is the syntax of the commands that can be used to store an encrypted password.

Use the following syntax to be prompted for the password that will be encrypted and stored in a file.

```
stcmd store-password -epwdfile "filePath"
```

Use the following syntax to include the encrypted password in the command as clear text.

Note: This action does not access the network with the clear value.

```
stcmd store-password -epwdfile "filePath" -password "password"
```

Parameter	Description

After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile "filePath"` as parameters. For example:

```
stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/
SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"
```

Important:

If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:

```
-p user@hostname:port/projectName.viewName -epwdfile
"pathToPasswordFile"
```

-cmp

Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place.

Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data.

This is an optional parameter. If not specified, then the platform default is not to compress.

-csf

When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder.

The default is that StarTeam folders are not differentiated based on the case of letters in their names.

With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.

-encrypt

Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks.

This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server.

The full syntax is: `-encrypt encryptionType`.

The types of encryption are:

RC4	RSA RC4 stream cipher (fast).
RC2_ECB	RSA RC2 block cipher (Electronic Codebook).
RC2_CBC	RSA RC2 block cipher (Cipher Block Chaining).

These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it.

Note: For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam

Parameter	Description
	Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.
-is	Applies the command to all child folders. Without this option, the command applies only to the specified folder. When this option is used with add-folder, you can add an entire branch of folders to the StarTeam folder hierarchy. When used with `add` or `ci`, the command recursively visits all modified files in all sub-folders and checks them in.
-vb	If specified, then the output reports all identified files, whether or not their status changed as a consequence of the call to update-status. When not specified (the default behavior), the output only describes those files whose status changed by virtue of this command.

-rp	Overrides the working folder or working directory for the StarTeam view's root folder.
	While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways.
	While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project.
	The UNIX shell interprets a backslash (\) as an escape character when it precedes certain characters, such as quotation marks. As a result, an error occurs in the following example:
	`stcmd ci -p "xxx" -rp "C:\" "*"`
	which is interpreted as:
	`stcmd ci -p "xxx" -rp "C:" *"`
	To avoid a situation like this, escape the final character in "`C:\`" as follows:
	`stcmd ci -p "xxx" -rp "C:\\" "*"`
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in "`C:\orion\`":
	`stcmd ci -p "xxx" -rp "C:\orion" "*"`
	The full syntax is: `-rp "folderName"` .
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-fp	Overrides the specified StarTeam folder's working folder or working directory. This is equivalent to setting an alternate working path for the folder. While this option allows you to use a different working folder than the one specified by the StarTeam view, its critical importance is to provide cross-platform compatibility. For example, UNIX and Microsoft Windows systems specify drive and directory path names in incompatible ways. While the path `D:\MYPRODUCT\DEVELOPMENT\SOURCE` is understood on a Microsoft Windows platform, it is not understood on a UNIX platform. Use this option to define the working path if your platform does not understand the path specified in the StarTeam project. A backslash (\) is interpreted as an escape character when it precedes quotation marks. As a result, an error occurs in the following example: `stcmd ci -p "xxx" -fp "C:\" "*"`

Parameter	Description

	which is interpreted as:
	`stcmd ci -p "xxx" -fp "C:" *"`
	To avoid a situation like this, escape the final character in "C:\" as follows:
	`stcmd ci -p "xxx" -fp "C:\\" "*"`
	Or avoid it as follows when the `-rp` path doesn't end with the root folder as in `C:\orion\`:
	`stcmd ci -p "xxx" -fp "C:\orion" "*"`
	The full syntax is: `-rp "folderName"`.
	Folder is the Microsoft Windows term and appears in the StarTeam user interface. Directory is the correct term for the UNIX platform.

-filter	Specifies a string of one or more characters, each of which represents a file status. Never include spaces or other white space in this string. Only files that currently have the specified status(es) will be actioned. Does not apply to files that are `Not In View`. • `C = Current` • `M = Modified` • `O = Out of date` • `N = Not In View` • `I = Missing` • `G = Merge` • `U = Unknown` For example, using `CM` applies a command only to files with a status of `Current` or `Modified`. `-filter` takes precedence over `-f NCI`. If you use `G`, `O`, or `U`, you must also specify `-I` or `-o`. Otherwise the `G`, `O`, or `U` is ignored. `-filter` also takes precedence over `-f NCO`. If you use `G`, `M`, or `U`, you must also specify `-o` to force the checkout operation. Otherwise, the `G`, `M`, or `U` is ignored.
-cfgl	Configures the view using the specified label. Without `-cfgl`, `-cfgp`, or `-cfgd`, the view's current configuration is used.
-cfgp	Configures the view using the specified promotion state.
-cfgd	Configures the view as of the specified date/time. Examples include: "12/29/13 10:52 AM" "December 29, 2013 10:52:00 AM PST" "Monday, December 29, 2013 10:52:00 AM PST"
-pattern	Qualifies the datetime. It can be specified wherever a date-time is specified, such as `-cfgd`, `-vd`, etc. The pattern must match any valid pattern supported by the java JDK in `java.text.SimpleDateFormat.applyLocalizedPattern(String)`. The pattern may be localized. For every command that takes a `-pattern` parameter, a `-locale` parameter is optionally available. This is the "two character country code".
-q	Enables quiet mode. The `-q` option is retained for backward compatibility with the old command line. If `-q` is specified, then `-pf` cannot be specified. The command will return no results.

Parameter	Description
-pf	Specifies the filter name whose associated filter properties produce the columns in the output matrix. Each command returns a result matrix. `-pf` determines the matrix columns. See `-ofp` for more information. If not specified, the primary descriptor property of the Type is returned as the command output. `-pf` does not apply to the select query command.
-ofp	Provides a file name with a fully qualified path into which to write the command output. By default, a "|" character separates each column in the output. A new line separates each row. The first row is the command name. The second row has the property names. All subsequent rows contain the data. If the file already exists, the output is appended to the end of the file. It is possible to override the "|" character separator by specifying `separator = fieldSeparator` as a parameter to the connect command. For example, `separator = ;;` specifies two adjacent semicolons (`;`) as the column separator.
files...	Specifies the files to be used in the command by name or by file name-pattern specification, such as `"*.c"`. All options are interpreted using the semantic conventions of UNIX instead of Windows because UNIX conventions are more specific. This means that `"*"`, rather than `"*.*"` means “all files.” The pattern `"*.*"` means “all files with file name extensions.” For example, `“star*.*”` finds `starteam.doc` and `starteam.cpp`, but not `starteam`. To find all of these, you could use `"star*"`. Without this option, the default is `"*"`. When used, this option must always be the last option. Any options after it are ignored. If you use *, rather than `"*"` to indicate all files, a UNIX shell expands it into a series of items and passes this series as a group of options to the `stcmd` command. This can cause problems, for example, when you are checking out missing files, so it is best to use `"*"` to avoid unwanted complications. If you use a set of file patterns, each pattern should be enclosed in its own set of quotation marks. For example, you can use `"*.bat" "*.c"`, but you cannot use `"*.bat *.c"`. **Note:** Always enclose this option in quotation marks. Failure to do so can result in unpredictable consequences on all supported platforms. Several special characters can be used in the file specification: ***** Matches any string including the empty string. For example, * matches any file name, with or without an extension. "xyz*" will match "xyz" and "xyz.cpp" and "xyzutyfj". **?** Matches any single character. For example, `"a?c"` will match `"abc"` but NOT `"ac"`. **[...]** Matches any one of the characters enclosed by the left and right brackets. **-** A pair of characters separated by a hyphen (-) specifies a range of characters to be matched. If the first character following the right bracket (`[`) is an exclamation point (`!`) or a caret (`^`), the rest of the characters are not matched. Any character not enclosed in the brackets is matched. For example, `"x[a-d]y"` matches `"xby"` but not `"xey"`. `"x[!a-d]y"` matches `"xey"` but not `"xby"`. A hyphen (-) or right bracket (`]`) may be matched by including it as the first or last character in the bracketed set. To use an asterisk (*), question mark (?), or left bracket (`[`) in a pattern, you must precede it with the escape character (which is the backslash (\).

Example

The following example uses `update-status` to verify that each file in the working folder for the `StarTeam` folder named `SourceCode` has an accurate status. `SourceCode` is a child of the root folder `StarDraw` (in the `StarDraw` view of the `StarDraw` project).

Use the `-p` with `update-status` or the stateful `set` command to set the context of the project/view/parent folder.

```
stcmd update-status -rp "/StarDraw/StarDraw/SourceCode" "*"
```

Update Property: update-property

Updates the display name of any StarTeam property on any type.

The display name change is specific to the selected configuration.

In a multi-configuration server, all other configurations will have the original display name specified in the Locale .XML in the server install folders.

Note: This command requires administrative privileges.

Syntax

The syntax for this command is:

```
update-property -type {typeName} -property {propertyName} -
sortEnumsAlphabetically
-displayName {newDisplayName} –locale {localeName} -s
"user:password@host:port"
```

Parameter	Description
-type	Specifies the type of item. The type is one of the stock type names, such as `changerequest`, `task`, `requirement`, `sprint`, `story`, `plan` or any custom type name that is applicable to the command.
-property	The name of the enumerated property.
-sortEnumsAlphabetically	Optional. If specified, the property must be an enumerated property. If specified, then all the enums in the enumerated property will be sorted alphabetically, and saved, so that they show up in the StarTeam Cross-Platform Client alphabetically.
-displayName	A name for a type that can be translated to the supported languages.
-locale	Optional. It may be one of `en`, `de`, `fr`, `ja`, `pt`, or `zh`. If specified, then the display Name is updated for the specified locale. If not specified, then the display Name is updated for the current locale.
-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` . The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.

Parameter	Description

	Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.
	
	In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.
	
	The following is the syntax of the commands that can be used to store an encrypted password.
	
	Use the following syntax to be prompted for the password that will be encrypted and stored in a file.
	`stcmd store-password -epwdfile "filePath"`
	
	Use the following syntax to include the encrypted password in the command as clear text.
	
	Note: This action does not access the network with the clear value.
	
	`stcmd store-password -epwdfile "filePath" -password "password"`
	
	After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile` "`filePath`"' as parameters. For example:
	`stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"`
	
	Important:
	If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:
	`-p user@hostname:port/projectName.viewName -epwdfile "pathToPasswordFile"`

-cmp	Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place. Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data. This is an optional parameter. If not specified, then the platform default is not to compress.
-csf	When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder. The default is that StarTeam folders are not differentiated based on the case of letters in their names.

Parameter	Description
	With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.
-encrypt	Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks. This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server. The full syntax is: `-encrypt encryptionType`. The types of encryption are: **RC4** RSA RC4 stream cipher (fast). **RC2_ECB** RSA RC2 block cipher (Electronic Codebook). **RC2_CBC** RSA RC2 block cipher (Cipher Block Chaining). These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it. **Note:** For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.
-s	Identifies the StarTeam Server. The full syntax is: `-s "userName:password@host:portNumber"` For example: `-s "JMarsh:password@orion:49201"` If the user name is omitted, the current user name is used. The user name in the example is "`JMarsh`". If the password is omitted, the user is prompted to enter the password. The password in the example is "`password`". If the host name is omitted, the default is `localhost`. The host name in the example is "`orion`". The port number is required. The default port number, `49201`, is used in the example.

Update User: update-user

Use the `update-user` command to update a user properties on the StarTeam Server.

Either `-logonname` or `-userid` are required to identify the user. All other parameters are optional.

If specified, the new property will be applied to the user.

Only users with administrative privileges can run this command on behalf of any user.

Syntax

The syntax for this command is:

```
stcmd{Ex} update-user –s “username:password@host:port” –logonname “logon
name” | -userid “userid”
[–password “password”] [-name “user full name” ]
[-phone “phone number”] [-email “email address”] [-address “postal address”]
[-voicemail “voice mail number”] [-pager “pager number”] [-fax “fax number”][-
distinguishedname “distinguished name”]
[-directoryservicevalidation] [-fixedlicense]
```

Parameter	Description
-s	Identifies the StarTeam Server. The full syntax is: `-s "userName:password@host:portNumber"` For example: `-s "JMarsh:password@orion:49201"` If the user name is omitted, the current user name is used. The user name in the example is “`JMarsh`”. If the password is omitted, the user is prompted to enter the password. The password in the example is “`password`”. If the host name is omitted, the default is `localhost`. The host name in the example is “`orion`”. The port number is required. The default port number, `49201`, is used in the example.
-logon name	Name used for logging in to the Server.
-userid	Unique integer user ID of the user on the StarTeam Server.
-password	User's new password.
-name	User's full name.
-phone	User's phone number.
-email	User's email address.
-address	User's postal address.
-voicemail	User's voicemail number.
-pager	User's pager number.
-fax	User's fax number.
-distinguishedname	User's directory service distinguished name
-directoryservicevalidation	If specified, turns on directory service validation.
-fixedlicense	If specified, sets the user to inherit a fixed license.

User Resource: user-resource

Use the `user-resource` command to add or remove a user resource from a Requirement or Task.

Note: If the user already exists as a resource/responsibility for this item, the command does nothing.

Syntax

The syntax for this command is:

```
stcmd{Ex} user-resource –p “user:pwd@host:port/project/view” [ -epwdile
“pathToPasswordFile” ] –type “typeName” –id itemID
-user userID [-rmv]
```

Parameter	Description
-p	Indicates the view or folder to be used. It also provides the user name and password needed to access the server. `-p` is retained for backward compatibility. Commands using `-p` continue to work, but are stateless. Each command opens a connection, executes the command syntax, and closes the connection. (New command line scripts may take advantage of the command line's stateful nature. See `connect` and `set` for examples. Old scripts may be migrated to the new command line syntax.) Stateless commands cause more client server traffic than stateful commands.

Note: If the clear text password contains the `@` or the `:` symbols, then it cannot be specified through `-p` using the syntax `username:password@host:port`. The `@` or `:`symbols will conflict with the syntax and cause the command to fail. In general, passwords with special characters in them such as `@, :, ,,` must be stored in the password file using the `store-password` command. Additionally, the password, when specified for storage in the encrypted file, must be quoted. For example: `stcmd store-password -password "foo@bar" -epwdfile c:\tmp\pwdfl`. Passwords stored in an encrypted password file can be used in conjunction with `-p` or the `connect` command as documented.

The full syntax is:

```
stcmd -p "userName:password@hostName:endpoint/projectName/
[viewName/][folderHierarchy/]"
```

For example:

```
stcmd -p “bsmith:rocketfive@orion:49201/StarDraw/StarDraw/
SourceCode/”
```

- If the user name is omitted, the current user name is used.
- If the password is omitted, the user is prompted to enter the password. When the user types a password, the characters are not displayed on the screen.
- If the host name is omitted, the default is localhost.
- Entering an endpoint (port number) is required. The default is 1024.
- The project name is always required.
- A view hierarchy should be used to identify the view. Use the colon (`:`) as a delimiter between view names. The view hierarchy should always include the root view. For example, "`StarDraw:Release 4:Service Packs`" indicates that the view to be used is the `Service Packs` view, which is a child of the `Release 4` view and a grandchild of the `StarDraw` root view. If the view name is omitted, the root view is used. If the view is the only view in that project with that name, you can use only the view name. Doing this is not recommended, however, because another view with that name could be created at a later date.
- A folder hierarchy should be used to identify the folder. Use the forward slash (`/`) as a delimiter between folder names. The folder hierarchy never includes the root folder. Omit the folder hierarchy if the file is in the view’s root folder. For example, if the root folder of the view is `StarDraw`, and the hierarchy to your files is `StarDraw/SourceCode/Client`, use only "`SourceCode/Client`".

Parameter	Description

-epwdfile	The `-epwdfile` keyword specifies the path to the file that contains the encrypted password. Like `-pwdfile` , `-epwdfile` replaces the password being used as part of the `-p` or `-s` option, preventing others from seeing the user's password on the command line. The full syntax is: `-epwdfile "filePath"` .
	
	The `-pwdfile` is supported for backward compatibility. Un-encrypted passwords stored using older versions of `stcmd` are read. However, passwords cannot be stored to files using `-pwdfile` anymore.
	
	Note: When `-epwdfile` is used, a password should not be specified as part of the `-p` or `-s`parameter.
	
	In this case, the syntax of `-p` or `-s` reduces to `-p "username@hostname:port/... -epwdfile "fullyQualifiedPathToPasswordFile""`.
	
	The following is the syntax of the commands that can be used to store an encrypted password.
	
	Use the following syntax to be prompted for the password that will be encrypted and stored in a file.
	`stcmd store-password -epwdfile "filePath"`
	
	Use the following syntax to include the encrypted password in the command as clear text.
	
	Note: This action does not access the network with the clear value.
	
	`stcmd store-password -epwdfile "filePath" -password "password"`
	
	After an encrypted password is stored, other `stcmd` commands can specify `-epwdfile` "`filePath`" as parameters. For example:
	`stcmd delete-local -p "JMarsh@Orion:1024/StarDraw/StarDraw/SourceCode" -epwdfile "C:\estuff\myfile.txt" -filter "N" "*"`
	
	Important:
	
	If `-p` or `-s` and `-epwdfile` are used together, then the parameter `:password` must be omitted from `-p`. For example:
	`-p user@hostname:port/projectName.viewName -epwdfile "pathToPasswordFile"`

-cmp	Compresses all the data sent between the workstation and the server and decompresses it when it arrives. Without this option, no compression takes place. Compression speeds transmission across the network, but it takes time on the front end to compress the data and at the back end to decompress the data. This is an optional parameter. If not specified, then the platform default is not to compress.
-csf	When the command maps the folder specified in the `-p` option to the underlying StarTeam folder, using `-csf` causes the command to differentiate StarTeam folders based on the case-sensitive spelling of their names This option does not apply to the case-sensitivity of filenames in the folders. For example, with `-csf`, StarTeam folders named `doc` and `Doc` are recognized as different folders. Without this option, either folder could be recognized as the `doc` folder. The default is that StarTeam folders are not differentiated based on the case of letters in their names.

Parameter	Description
	With or without `-csf`, if folder names are ambiguous, an error occurs. For example, when you use `-csf`, the names of two folders are ambiguous if both a `Doc` and `doc` folder exist. When you do not use `-csf`, folder names are ambiguous if they are spelled identically.
-encrypt	Encrypts all data sent between the workstation and the server and decrypts it when it arrives. Without this option, no encryption takes place. Encryption protects files, data and other project information from being read by unauthorized parties over unsecured networks. This is an optional parameter. If not specified, then the server and the command line negotiate the encryption required by the server. The full syntax is: `-encrypt encryptionType`. The types of encryption are: **RC4** RSA RC4 stream cipher (fast). **RC2_ECB** RSA RC2 block cipher (Electronic Codebook). **RC2_CBC** RSA RC2 block cipher (Cipher Block Chaining). These encryption types are ordered from fastest to slowest. Each of the slower encryption types is safer than the one preceding it. **Note:** For platforms other than Microsoft Windows, the public and private keys used in the encryption process are not created automatically. They are stored in a file in the user's home directory. This options file is named `.starteam`. It contains a variable or shell variable called *keyfile*. The *keyfile* variable specifies the location of the file that contains the public and private keys. If you do not specify the *keyfile* variable, an error occurs. When you specify the *keyfile* variable, but the file does not exist, the StarTeam Cross-Platform Client generates a random pair of keys, creates the file, and stores the keys in it. Be sure to secure this file. For example, in UNIX, only its owner should be able to read it.
-type	Specifies the type of item. The type is one of the stock type names, such as `changerequest`, `task`, `requirement`, `sprint`, `story`, `plan` or any custom type name that is applicable to the command.
-id	Specifies the unique item (view member) ID of the item. Look in the property lists of the CPC or query using the select command to find the View Member IDs. `-id` can also specify the primary descriptor of the item; e.g. file name, folder name, change request number.
-userID	Type either the unique integer userID of the User, the case sensitive User Full Name, or the User Logon Name. **Note:** If the user logon name is specified, the command must be run by an Administrator.
-rmv	If specified, remove the user.

Version: version

Use `version` to show the SDK version number that you have installed.

Syntax

The syntax for this command is:

```
stcmd{Ex} version
```

Examples

The following command returns an SDK version number such as `14.0.1`, for example.

```
stcmd version
```

starteamserver Command Parameters

The following topics contain the parameters for the `starteamserver` command with examples of their uses.

-access

Registers the StarTeam Server as a licensed version. Use this option with the `-serial` option. The first time you start the StarTeam Server, you must register the application as either a licensed version or an evaluation copy.

Use with: *-serial*.

See also: *-serial*, *-license*, and *-eval*.

Syntax

```
-access Key
```

Example

```
starteamserver -serial 1234 -access 5678
```

-all

Used in conjunction with the `-start` (or `-restart`) or `-stop` options. The `-start -all` options start all server configurations that have a status of `Ready` in the `starteam-server-configs.xml file`. The `-stop` and `-all` options stop all server configurations that have a status of `Running`.

Use with: *-start*, *-stop*, and *-restart*.

Syntax

```
-all
```

Example

```
starteamserver -stop -all
```

-autorecover

The `-autorecover` option instructs the StarTeam Server to attempt to make limited repairs where necessary during the verification process.

Use with: *-start*.

See also: -stoponerrors.

Syntax

```
-autorecover
```

Example

```
starteamserver -start MyServer -autorecover
```

-dbport

This is an optional parameter. If this value is not supplied, the default port is assumed. For Microsoft SQL Server the default port is 1433. For Oracle, the default port is 1521. Use this parameter only if the database is not running on the default ports.

-dbserver

Specifies the database connection information. Enter the existing `Database Server Name`.

In releases 5.1 and 5.2, Oracle databases were accessed using the Oracle net service name that is stored in `$ORACLE_HOME/network/admin/tnsnames.ora`. This is no longer the case.

The value you specify for "DatabaseServerName" is stored in the `starteam-serverconfigs.xml` file. You can review or modify the database connection information by using:

- The `-view` and `-edit` options from the command line.
- Database tab of the **StarTeam Server Configuration** dialog box in **StarTeam Administration**.
- Database tab of the **<Server configuration> Properties** dialog box in **Server Administration**.

Modifications take effect the next time you start the server configuration.

Use with: *-new*, *-edit*, *-start*, and *-restart*.

See also: *-t*, *-p*, and *-u*.

Syntax

```
-dbserver  "DatabaseServerName"
```

Example

```
starteamserver -edit MyServer -dbserver
```

-dbservicename

Use for Oracle to identify the Oracle service on the host machine. Use either `-dbservicename` or use `-dbsid`.

-dbsid

User for Oracle to identify the Oracle service on the host machine. Use either `-dbservicename` or use `-dbsid`.

-edit

Use with: `-name`, `-dsn`, `-u`, and `-p`.

Edits the session options for the specified server configuration. You can edit the following options: `-name, -dsn, -u, -p`. If the server configuration is running, you must shut it down before you can make any edits.

Syntax

```
-edit ConfigurationName
```

Example

```
starteamserver -edit MyServer -name Portable -dbserver RemoteServer -u
StarTeamAdmin -p password
```

-eval

Extends the evaluation period for an evaluation copy of the StarTeam Server. The first time you start the StarTeam Server, you must register the application as either a licensed version or an evaluation copy.

See also: *-serial*, *-access*, and *-license*.

Syntax

```
-eval Number
```

Example

```
starteamserver -eval 01234567890
```

-help

Displays a message describing all of the command options.

Syntax

```
-help
```

Example

```
starteamserver -help
```

-licenses

Displays license and registration information. If you are running an evaluation copy of the application, the system displays a message informing you of this. Otherwise, the system displays your serial number.

See also: *-serial*, *-access*, and *-eval*.

Syntax

```
-licenses
```

Example

```
starteamserver -licenses
```

-list

Lists the StarTeam Server configurations defined in the `starteam-server-configs.xml` file and the status of each one. A StarTeam Server configuration can have one of the following statuses at any given point in time: `Ready`, `Starting`, `Running`, `Disabled`, and `Stopping`.

Syntax

```
-list
```

Example

```
starteamserver -list
```

The StarTeam Server displays a message similar to the following:

```
Configuration Status MyServer Ready StarDrawRepository Running Portable Ready
```

-mb

This is an optional parameter used when creating a new server configuration. Use the following values to set the type of message broker:

- `0 = None`
- `1 = StarteamMPX`
- `2 = ActiveMQ MPX`

If this value is not specified, the new configuration is configured with ActiveMQ message broker.

-name

Renames a StarTeam Server configuration. This option is used in conjunction with the `-edit` option. The new StarTeam Server configuration name will take effect the next time you start the StarTeam Server configuration.

Use with: *-edit*, *-start*, and *-restart*.

Syntax

```
-name ConfigurationName
```

Example

```
starteamserver -edit MyServer -name NewTeamServer
```

-new

Creates a hive named `DefaultHive` for the new server configuration with the specified name and settings. This configuration uses a Native-II vault. This option produces the same result as selecting **New** on the **Server Administration Tool** menu, and using the wizard to create a new configuration.

A number of options can only be specified with `-new`. These are: `-c`, *-r*, and *-t*.

Syntax

```
-new ConfigurationName
```

Example

```
starteamserver -new NewServer1 -r "c:\new server\" -t 1 -database
RemoteServer -dbtype 1 -dbusername amin -dbuserpassword admin
-u Admin -p password
```

-p

Specifies the password used to access the database. The value you specify for `DBUserPassword` is stored in the `starteam-server-configs.xml` file. Ensure that the password you specify is the correct one for the database user name. You can review or modify the password and user name using the `-view` and `-edit` options from the command line. Any modifications you make will take effect the next time you start the server configuration.

Use with: *-new*, *-edit*, *-start*, and *-restart*.

See also: *-t*, and *-u*.

Syntax

```
-p DBUserPassword
```

Example

```
starteamserver -edit MyServer -u JodyK -p password
```

-r

Specifies the repository path for a new StarTeam Server configuration. If the repository path you specify does not exist, the system will create the appropriate folders the first time you start this StarTeam Server configuration.

The value you specify for `RepositoryPath` is stored in the `starteam-serverconfigs.xml` file. You can review the repository path using the `-view` option from the command line or in the application on the **General** tab of the **StarTeam Server Configuration** tool in the **Server Administration Tool**.

Caution: Do not use the StarTeam Server home folder/directory as a StarTeam Server configuration repository path because the StarTeam Server configuration will not start.

Use with: *-new*.

Syntax

```
-r RepositoryPath
```

Example

```
starteamserver -new NewServer1 -r "c:\new server\" -t 1 -dbserver NewServer -
u Admin -p password
```

-remove

Deletes the specified StarTeam Server configuration from the `starteam-server-configs.xml` file.

Syntax

```
-remove ConfigurationName
```

Example

```
starteamserver -remove MyServer
```

-restart

Stops and restarts the specified StarTeam Server configuration. Use this option after you make changes to a StarTeam Server configuration and want those changes to take effect. If the StarTeam Server configuration fails to restart, check the StarTeam Server log file for more information.

You can restart a StarTeam Server configuration and modify a number of its options at the same time.

The following options can be used with the `-restart` option: *-all*, `-attach`, *-dbservername*, *-name*, *-p*, *-tcpip*, and *-u*. You cannot use both the `-all` and the specific configuration name at the same time.

Syntax

```
-restart ConfigurationName
```

Example

```
starteamserver -restart MyServer -tcpip StarTeamTCPIP -u SuperUser -p
SuperUserPassword
```

-serial

Registers the StarTeam Server as a licensed version. Use this option with the `-access` option. The first time you start the StarTeam Server, you must register the application as either a licensed version or an evaluation copy. The serial and access numbers in the example below would be replaced with actual serial and access numbers.

See also: *-access*, *-license*, and *-eval*.

Syntax

```
-serial Number
```

Example

```
starteamserver -serial 1234567890 -access 9999999
```

-start

Starts the specified StarTeam Server configuration. `starteamserver` updates the StarTeam Server configuration entry in the `starteam-server-configs.xml` file to `Status=Running` and `PID=nnn` where `nnn` would be replaced with the actual `PID` number.

You can start a StarTeam Server configuration and modify a number of its options at the same time.

The following options can be used with the `-start` parameter: `-attach`, *-dbservername*, *-name*, *-p*, *-tcpip*, and *-u*.

See also: *-all* and *-stop*.

Syntax

```
-start  ConfigurationName
```

Example

```
starteamserver -start MyServer -tcpip StarTeamTCPIP -u SuperUser -p
SuperUserPassword
```

-stop

Shuts down the specified StarTeam Server configuration. After the StarTeam Server configuration stops running, `starteamserver` updates the entry in the `starteam-server-configs.xml` file to `Status=Ready` and `PID=0`.

Note: For enterprise advantage users: If you are running the StarTeam Server as a service and StarTeam Notification Agent as a dependent service, you cannot shut down the StarTeam Server unless the StarTeam Notification Agent service is shut down first.

See also: *-all* and *-start*.

Syntax

```
-stop ConfigurationName
```

Example

```
starteamserver -stop MyServer
```

-t

Specifies the database type. This option can be used only when you are creating a new StarTeam Server configuration. Use one of the following numbered values to indicate the type of database:

- `2 = Microsoft SQL Server or SSE`
- `3 = Oracle`

The value you specify for `DBType` is stored in the `starteam-server-configs.xml` file. You can review the database type using:

- The `-view` option from the command line.
- In **StarTeam Administration Tool** on the **Database** tab of the **StarTeam Server Configuration** tab.
- In **Server Administration** on the **Database** tab of the `<server configuration="">` **Properties** dialog box

See also: *-dbservername*, *-p*, *-u*.

Use with: *-new*.

Syntax

```
-t DBType
```

Example

```
starteamserver -new NewServer1 -r "c:\new server\" -t 2 -dbserver NewServer -
u Admin -p password
```

-tcpip

Sets the endpoint for the TCP/IP (Sockets) protocol. Also enables or disables the protocol. Use `up` to enable and `down` to disable. You can both set the endpoint and enable or disable it using `up` or `down` followed by a colon and the endpoint.

The value you specify for the endpoint is stored in the database used by this StarTeam Server configuration.

You can modify this information using the *-start* (or *-restart*) and -tcpip options from the command line or in the application on the **Protocol** tab of the **StarTeam Server Configuration** tab.

Use with: *-start*, *-restart*.

Syntax

```
-tcpip  Endpoint | up[:Endpoint] | down[:Endpoint]
```

Example

```
starteamserver -start MyServer -tcpip 49201 starteamserver -start MyServer -
tcpip up
```

-u

Specifies the user name that the StarTeam Server configuration uses to access the database. The value you specify for `DBUserName` is stored in the `starteam-server-configs.xml` file. You can review or modify the database user name using the `-view` or `-edit` options from the command line. Be sure to also specify the password for this user account. Any modifications you make will take effect the next time you start the StarTeam Server configuration. Ensure that the user name and password you specify using the `starteamserver` command is a valid account in the database. The StarTeam Server configuration will fail to start if the user account is missing in the database.

Use with: *-new*, *-edit*, *-start*, *-p*, and *-restart*.

See also: *-t*, and *-dbservername*.

Syntax

```
-u DBUserName
```

Example

```
starteamserver -edit MyServer -u SuperUser -p SuperUserPassword
```

-version

Displays the version and build number for the StarTeam Server.

Syntax

```
-version
```

Example

```
starteamserver -version
```

The StarTeam Server displays a message similar to the following:

```
StarTeam Server Version: x.x Build number: x.x.xxx
```

-view

Lists the session properties of the specified StarTeam Server configuration.

Syntax

```
-view ConfigurationName
```

Example

```
starteamserver -view StarDraw
```

Vault Verify Command-line Options

Below are descriptions of the command-line options for the Vault Verify utility.

In general, you can run Vault Verify from the command line as follows: `VaultVerify [options] "configuration"` .

Based on the default or given `-check` options, integrity checks are performed on the vault archive files for the specified StarTeam "server configuration". If you specify the `-repair` option, Vault Verify attempts to correct problems found. Vault Verify opens the database for the server configuration but does not modify it. Valid options for Vault Verify are described in the following table.

-check {missing | corrupt | stray | all}

Determines which integrity checks to perform:

missing	Checks for missing files by comparing the database against archive files actually present.
corrupt	Checks the integrity of existing archive files (MD5, name, folder, and .gz file format).
stray	Checks for extraneous files based on the database. This option cannot be used if the server configuration is in use.
all	Performs all integrity checks.

Multiple `-check` options can be specified. Also, see the `-repair` option.

–cf <folder path>

Path name of the *corrupt file folder*, where problem files found by the `corrupt` check are moved when `-repair` is specified. The default corrupt file folder is `C:\Temp\VVCorruptFiles`.

–dbhost <host>

Specifies the host name of the database for the specified <server configuration>. On Microsoft Windows, it is only meaningful when `–dbinstance` is also provided. On Microsoft Windows and Linux, use this option only when the database server executes on a different host than this one.

-dbname <name>

Specifies the database name for the specified <configuration>. On Microsoft Windows, this parameter is only meaningful when `–dbinstance` is also specified, and it is only needed when the database name is different than the database server name. On Linux, use this option only if `–dbinstance` is not used and the Oracle service name is different than the server name or SID.

-dbinstance <name>

This option is only meaningful on Windows. When used, it causes `VaultVerify` to open the database directly instead of via the database server name specified in the configuration file. For Microsoft SQL Server, the `<name>` must be the instance name (for example, `'SSE2005_ST'`).

Note: The default Instance name for Microsoft SQL Server is 'MSSQLSERVER' and for Microsoft SQL Server Express, it is 'SQLEXPRESS'. For Oracle, should be the service name, (for example, 'ORCL').

`-dbinstance` must be used with `-dbhost` when the database server executes on a different host. For SQL Server, `-dbname` should also be used if the database name is different than the Database Server Name. For Oracle, `-dbname` is ignored if `-dbinstance` is specified.

-dbpassword <password>	Specifies the database logon password. If not specified, a blank password is used. (The password stored in the configuration is encrypted and cannot be used by `VaultVerify`.) On server configurations running against Oracle, this option must be specified since the Oracle password is never empty.
-dbport <port>	Specifies the TCP/IP port to use to connect to the database server. This parameter is only used on non-Microsoft Windows platforms when a different port is used than the vendor's default database port (for example, 1521 for Oracle).
-dbuser <user>	Specifies the logon ID used to connect to the database. If specified, this parameter overrides the user specified in the StarTeam <configuration>. The only valid user to use with this option is the user that owns the StarTeam tables.
-help (or -h or -?)	Displays this usage information.
–path <folder path>	Specifies the folder path of the `starteam-server-configs.xml` file. This file must exist and contain the specified <server configuration>. By default, this file is opened in the parent folder of the current working directory if it is not found in the current working directory.
-nosharereport	Suppresses the reporting of share information. Normally, all share paths of each corrupt file is reported. This option suppresses the share path information, which can speed up application execution and substantially reduce the report size.
-repair	Specifies that an attempt should be made to correct archive file problems. 'Corrupt' archives are moved to the '`corrupt file folder`' (see the `-cf` option). If they correspond to valid file revisions, they are then treated as missing. Missing archive recovery is attempted from other vault files and, if the `-useca` option is specified, from a Cache Agent. Stray archives are moved to the 'stray file folder' (see the `-sf` option). **Note:** `-repair` is ignored if the StarTeam <configuration> is in use.
-sf <folder path>	Path name of the '`stray file folder`', where extraneous files found by the 'stray' check are moved when `-repair` is specified. The default 'stray file folder' is `C:\Temp\VVStrayFiles`.
–t	Displays elapsed time information when the verification finishes.
-useca <host>:<port>	If `-repair` is specified, this option enables attempts to recover missing files from the specified MPX Cache Agent. The `<host>` and `<port>` must designate a remote MPX Cache Agent because it maintains an independent cache.
-verbose	Displays additional status information as the verification proceeds.
“configuration”	Specifies the configuration name. The configuration name passed to `VaultVerify` is case-sensitive, and if it includes spaces, you must pass the configuration name to Vault Verify in quotation marks.

VCM Command-line Utility

Overview of the VCM Command-line Utility (VCMUtility)

The `VCMUtility` is a command-line utility that compares a StarTeam source view to a target view, and optionally merges the differences into the target view.

You can start a View Compare/Merge session from the command line and finish it in the StarTeam Cross-Platform Client in the View Compare/Merge UI. For example, you can use the `VCMUtility` to create a VCM session, perhaps using its `DefaultAction` option, but do not let it commit. It will automatically save the VCM session with any alternate name you choose if needed. You can then open that VCM session in the StarTeam Cross-Platform Client, review and make adjustments, then commit the changes to the repository.

Syntax Conventions

The syntax for the command-line uses the following conventions.

Curly braces {}	Encloses required syntax elements.
Square brackets []	Encloses optional elements
Angle brackets < >	Encloses a word or phrase that must be replaced with an appropriate value, or set of values. For example, `<file name>` would be replaced by an actual filename or path, and `<userid>` would be replaced by an actual user ID. However, many of the words or phrases in angle brackets can be expanded into more complicated syntax. For example, `<change requests>` can be replaced by `CR`, `CRs`, `ChangeRequest`, `ChangeRequests`, `ChangeRequest *4277`, and so on. When you are not sure what can be used, see the `VCMUtility` “Cheat Sheet” topic in the `Reference/CompareMerge` section of the documentation. The “Cheat Sheet” provides the full syntax for phrases like `<change requests>`.
Vertical bar |	Separates alternate elements.
Prefixed with an asterisk *	Indicates the following element can be repeated.

Note: All options are case-insensitive (for example, `Server` is the same as `server`).

VCMUtility Command

```
VCMUtility [<options file>] [options] [options] [epwdfile file]
[DifferenceReport file][UpdateReport file]
```

You can provide options in the specified `<options file>` (as the first parameter), command-line arguments, or both. Command-line arguments override any options found in the `<options file>`. In the `<options file>`, the option name should begin as the first character on a new line and exclude the leading `-`.

VCMUtility Options File

You can specify `VCMUtility` options in an options file whose name is passed as the first parameter of the `VCMUtility` command.

Example:

```
VCMUtility c:\VCMconfig.txt
```

Each option in the file must begin on a new line. Option names must begin in column 1 and be followed by at least one white space character. An option's value can flow onto multiple lines by starting each continuation line with a blank or tab character. Blank lines are ignored. You provide comments by prefixing them with a double forward-slash (`//`).

Example:

```
// This is a comment
server jsmith:mypw@somehost:49201
type Rebase
include "/Cygnus/StarTeam/<StarTeam Core>/Server/Common/*.h" +ALL
  *.cpp    *.rc Makefile    // long value continued on a second line

// The line above was blank
save
 my—rebase-session               // value provided on a separate line
```

Command-line Parameters

`VCMUtility` options can be passed as command-line parameters by placing a dash in front of the option name. For example, the `Server` option can be provided as a command-line parameter `-server`. If an option has secondary "value" tokens, they must immediately follow the option name (without a dash).

Mixing Input Sources

`VCMUtility` options can be provided in an options file, with command-line parameters, or with a mixture of both. For example, commonly-used or "static" option values can be placed in the configuration file while "dynamic" values can be provided in command-line parameters.

A command-line parameter may specify the same option as defined in the configuration file. When a command-line argument specifies the same option as in the options file, the command-line option value overrides the configuration file option value. For example, if the configuration file specifies `Source View1` but the command-line specifies `-Source View2`, then `View2` is used as the source view.

Option Values with Unicode Characters

The encoding of option values passed as command-line arguments is controlled by the launching environment (for instance, command shell). Consequently, on systems where option values must be passed to the `VCMUtility` that require characters not expressible by the launching environment, those options must be passed by way of the options file.

When the options file does not begin with a byte-order mark (BOM), it is opened with the system default character set (for example, ANSI [Windows-1252] on Windows, UTF-8 on Linux). If the options file begins with a BOM, it is interpreted with the corresponding encoding. UTF-8 and UTF-16 encodings allow the full set of Unicode characters to be provided in the options file.

For Reference, the BOM sequences are:

0xEFBBBF	UTF-8
0xFEFF	UTF-16 BE (big-endian)
0xFFFE	UTF-16 LE (little-endian)

Boolean Options

The default for all Boolean options (whose value can be `True` or `False`) is `False`. However, specifying a Boolean option without an option value is the equivalent to specifying the value `True`. Thus, a Boolean option can be enabled by simply including it. Example:

```
// Set these options to True
AutoLogon
BreakLocks
```

Abbreviations

In addition to their "long form" (shown in this document), most command and option names have one or more "short forms" or abbreviations. These alternate spellings help shorten `VCMUtility` command tails with lots of options. The full lists of abbreviations can be achieved by using the command `-Help abbreviations`. Example abbreviations are:

- `Help`: `H` or `?`.
- `ActiveProcessItem`: `ActivePI` or `API`.
- `SourceLabel`: `SrcLabel` or `SL`.

In most cases, a syntactic item spelled with mixed-case in this document can be abbreviated to its "capitals only" short form. For example, `ManualMergeFiles` can be `MMF`, or `AutoMergeProperties` can be `AMP`, and so on.

Exit Codes

The VCM utility will return the following exit codes to indicate the results of its execution:

0 No errors occurred.

1 A fatal error occurred.

2 Partial success. This result is returned when the compare phase was performed, but the commit could not be performed due to unresolvable conflicts.

VCMUtility Log Files

During its execution, the `VCMUtility` writes informational, warning, and error messages to the console window (standard out). For most operations, the `VCMUtility` also creates a log file that summarizes its operation. As with console window output, the log file is more detailed when the `Verbose` option is enabled. The log file is created for new VCM sessions and for the `Import`, `Open`, `Replay`, and `Resume` commands. However, the log file is not started unless command-line parameters and the options file, if used, have been parsed without errors. A log file is not created for the `Help` or `Delete` commands.

The VCMUtility log file is created in the user's home directory (what Java identifies as *user.home*) with the following file name:

```
VCMUtility-YYYY-MM-DD_hh-mm-ss.log
```

`YYYY-MM-DD` and `hh-mm-ss` are the current date and time in the local time zone. The full path name of the log file is written to the console window when the log file is started.

VCMUtility Support for Change Packages

The `VCMUtility` supports change packages for any StarTeam configuration that has been upgraded to the 2009 release. Because change packages are persistent objects stored on the StarTeam Server, they offer many advantages and over VCM session (`.vcms`) and VCM export (`.vcmx`) files. Therefore, for StarTeam configurations that have been upgraded, change packages are preferred over session and export files for saving and resuming sessions. Correspondingly, the `Save` option without a parameter and the `Open` command are preferred over the `Save` option with a parameter, the `Resume` command, the `Export` command, and the `Import` command. However, for backward compatibility, the 2009 `VCMUtility` still

supports commands that use VCM session files. See the `Open` command and the `Save` option for more information.

VCMUtility Commands

This section defines `VCMUtility` functionality in terms of its utility execution commands. Each `VCMUtility` execution performs one command.

VCMUtility Command

```
VCMUtility [<options file>] [options] [epwdfile file] [DifferenceReport file]
[UpdateReport file]
```

You can provide options in the specified `<options file>` (as the first parameter), command-line arguments, or both. Command-line arguments override any options found in the `<options file>`. In the `<options file>`, start option names in column 1 and exclude the leading "`-`"

Note: `epwdfile` and `pwdfile` are mutually exclusive. `DifferenceReport` and `ReportDiffs` are mutually exclusive. `UpdateReport` and `ReportUpdates` are mutually exclusive.

VCMUtility Command Types

This section contains the `VCMUtility` command types. The default command type is a new VCM session.

New Session Command

By default, each `VCMUtility` execution begins a new VCM session unless the `Help`, **Open** , `Replay`, `Resume`, `Delete`, or `Import` command is explicitly given.

Help Command

?

```
Help [<option>]
```

Displays the `VCMUtility` Help. If you provide an `<option>`, help specific to that topic is displayed. For example, `Help MMF` would provide help on the `ManualMergeFiles` option.

Delete Command

```
Delete <VCM session file>
```

Specifies that the session stored in the specific `<VCM session file>` is to be deleted. All intermediate files (for example, merged result files) and the session file itself are deleted. However, if the session was previously saved as an uncommitted change package in the target view, the change package object is not deleted.

Import Command

```
Import <VCM exchange file>
```

The `Import` command is identical to the `Resume` command except that the `<VCM exchange file>` passed to it must be a VCM exchange file (`.vcmx`) previously created by an `Export` command. The imported VCM session is resumed where it left off:

- The compare phase is performed if it has not yet successfully completed.
- Manual merging is performed if `ManualMergeFiles` is specified and existing file merge conflicts exist.
- The target view "merge preview" is checked out if `CheckoutPreview` is specified and commit has not yet been performed.

- The differences report is generated if `ReportDiffs` is specified and commit has not yet been performed.
- The commit phase is performed if `CommitMerge` is `True` and commit has not yet been performed.
- The update report is generated if `ReportUpdates` is specified and commit has been performed.

`Export` and `Import` can be used together to "transport" a VCM session from one workstation to another. For example, one user could create a new VCM session, resolve all conflicts, then `Export` the session. The resulting archive file could then be transferred to a test machine, where the `Import` command can be used with the `CheckoutPreview` option (with `CommitMerge` set to False) to check out, build, and test the target merge "preview". If tests succeed, the test machine could then execute a `Resume` command and set `CommitMerge` to `True`.

Note: Sessions resumed by way of the `Resume` or `Import` command are automatically saved if they are not committed. If the `Save` option is specified, the session is saved in the specified `<VCM session file>`. Otherwise, the VCM session file specified by a `Resume` command is used; an automatically-generated VCM session filename is used for an `Import` command.

Open Command

`Open <Change Package name>`

Resumes a VCM session previously saved as a change package with the given name. This option is only available on servers that support change packages. The specified name must be the default or user-specified name of a saved, uncommitted change package belonging to the specified `Project` and `TargetView`, which are required. Also, the session must not be locked by another user, which typically indicates that it has already been opened by that user.

For additional information, see the `Name`, `Save`, `Import`, and `Resume` commands.

Replay Command

Replay `<Change Package name>`

Creates a new VCM session by "replaying" a previously-committed change package to a new target view. This command is only available when the server supports change packages. The named Change Package must belong to the project specified by the `Project` option and the view identified by the `SourceView` option. (Since committed change packages "belong" to the target view they update, the target view of the change package to be replayed is always the source view for the new session.)

When the `Replay` command is used, the `TargetView` should be specified, allowing the `MergeType` of the new session to be chosen automatically based on the relationship between the two views:

- If the target view is a child of the source view, a Rebase session is performed.
- If the target view is the parent of the source view, a Promote session is performed.
- Otherwise, a Replicate session is performed.

Alternatively, you can specify a `MergeType` of Promote, in which case the target view is not needed.

A replay VCM session attempts to make the same changes in the new target view that were made in the specified change package. This means that the source scope of the new VCM session is automatically chosen. Consequently, the `Include` and `Exclude` options are not allowed. In a replay session, some changes made in the original change package might not be possible in the new target view (such as when a new version is already present). Some changes may need to be applied in a different way (for example, `Move-and-Merge` instead of `Merge`), and new conflicts could appear (such as `Merge` instead of `Repin`). The replay session can be committed only if no unresolved conflicts occur.

Resume Command

`Resume <VCM session file>`

Specifies that the session saved in the given `<VCM session file>` is to be resumed instead of creating a new session. This is typically used to perform the commit phase of a previous session for which only the

compare phase was performed. A session that has already been committed can also be resumed, but only to generate a difference report. For more information, see the `Export` option and the `Import` command.

VCMUtility Connection Options

This section defines `VCMUtility` functionality in terms of its connection options.

AutoLogon

AL

`AutoLogon [True] | [False]`

If a `<user>` is not specified in the `Server` option, `AutoLogon` requests an attempt be made to log on using the userid/password for the specified StarTeam Server, as stored by the StarTeam **Toolbar Utility**.

Encryption

`Encrypt`

`En`

`Encryption {NONE | RC4 | RC2_ECB | RC2_CBC | RC2_CFB}`

Specifies the encryption level of the server connection. The default is `NONE`. However, due to SDK behavior, if necessary, the `VCMUtility` will automatically upgrade the encryption level to the minimum value required by the StarTeam server.

EPwdFile

`EPF`

`EPwdFile <file name>`

Specifies a file that contains the encrypted logon password. `-EPwdFile` overrides the `<password>` if provided in the `Server` parameter.

PwdFile

`PF`

`PwdFile <file name>`

Specifies a file that contains the logon password. `-PwdFile` overrides the `<password>` if provided in the `Server` parameter.

Server

S

`Server [<user>[:<password>]@]<host>[:<port>]`

Specifies the StarTeam server to which the VCM Utility will connect.

- If `<user>` and `AutoLogon` are not specified, the logon `<user>` defaults to "Administrator."
- If `<password>` and `PwdFile` are not specified, the `VCMUtility` prompts for the password.
- If a `<user>` or `<password>` contains the characters "`:`" or "@", or a blank, it must be enclosed in single or double quotes.
- If a `<user>` or `<password>` is quoted, it can contain an embedded quote of the same type by escaping (preceding) it with a backslash (\).

- If a quoted <user> or <password> contains an embedded backslash, it must be escaped with another backslash. For example, a double backslash within a quoted token is interpreted as a single backslash.
- The server <host> can be a host name or IP address. The <host> is required if the Server option is specified.
- If the Server option is not specified, the <host> defaults to **localhost**. If not specified, the <port> defaults to 49201.

UseCA

UCA

```
UseCA {<host>:<port> | AutoLocate}
```

Specifies that file check-outs should attempt to use an MPX Cache Agent. The Cache Agent can be explicitly provided with a host name or address (<host> and port number (<port>), or the network-nearest Cache Agent can be automatically located (AutoLocate).

UseServerProfile

USP

```
UseServerProfile [True | False]
```

If true, specifies that the <host> name specified in the Server option should be interpreted as a server profile name. Server profiles are stored in the user's starteam-servers.xml file. A server profile specifies a StarTeam server host name, port number, encryption level, and compression setting. Consequently, when UseServerProfile is specified, the Server option must be specified but should not contain a port number, and the Encryption option should not be specified.

VCMUtility Session Options

Session Options are grouped into two sections: "New Session Options" and "Resumed Session Options".

New Session Options

This section contains the VCMUtility session options.

Session Options	Description
AutoMergeFiles	AMF `AutoMergeFiles [True | False]` If True, requests automatic merging of files found in the compare phase in a merge state. When an auto-merge is successful, the result file is retained as part of the VCM session. Otherwise, the result file is discarded and the affected files remain in an unresolved merge state. AutoMergeFiles is ignored for Compare sessions.
AutoMergeProperties	AMP `AutoMergeProperties [True | False]` If True, requests automatic merging of properties for items found in the compare phase in a merge state. If property auto-merging is successful, the merged item is retained as part of the VCM session. Otherwise, the merged item is discarded and the items are flagged as being in an unresolved property merge state. AutoMergeProperties is ignored for Compare sessions.

Session Options	Description

BreakLocks	`BL`
	`BreakLocks [True | False]`
	If `True`, requests that an attempt is made to break any lock found on items used in the compare phase. Breaking a source or target item lock is only required when the lock is owned by another user. Lock breaking requires a special permission and may not be successful. `BreakLocks` is ignored for Compare sessions.

CaseSensitiveFilenames	`CSF`
	`CaseSensitiveFilenames [True | False]`
	If `True`, considers file names different only by case as unequal for purposes of evaluating the `PreventDuplicateFilenames` option and for matching files between source and target views.

CheckoutPreview	`Checkout`
	`CP`
	`CheckoutPreview <files> [<check-out options>]`
	This option specifies that files within a "merge preview" are to be checked out to the client workspace. A "merge preview" is a simulation of the target view updated with all changes in the VCM session. The `<files>` syntax allows file names and/or patterns to be checked out from specified folders in the merge preview. The optional `<check-out options>` control options such as where files are to be checked-out and what status of files should be checked-out.
	When `CheckoutPreview` is specified, files are checked out after the compare phase, after auto- and manual-merging has occurred, but before a commit occurs. The check-out occurs only if the VCM session has no file content merge conflicts. If merge conflicts exist, an error is displayed, and no merge is performed, regardless of the `CommitMerge` option. If no merge conflicts exist and `CommitMerge` is `True`, the VCM session is committed after the check out is performed.
	Example:
	`CheckoutPreview /src/com/acme/*.java +cwf +eol LF +filter CGMIOU +o +ro +rp C:\BuildDir`

CommitMerge	`Commit`
	`CM`
	`CommitMerge [True | False]`
	Specifies whether or not the results of VCM session should be committed. `False` specifies that a commit will not be performed. This option can be used to produce a compare/report-only session. `True` specifies that the commit should occur only if there are no unresolved conflicts. `CommitMerge` is ignored for Compare sessions.

CustomDifferenceTypes (CDT)	If specifying this parameter, provide a custom VCM merge type name (created on the **Customize VCM** tab in the Server Administration Tool) in which individual difference types can be overridden at will.

Session Options	Description

	For instance, Modified In Source, Modified In Target has a default action of Merge. This can be overridden to OVERWRITE. Similarly, Target View has Multiple Floating Shares has a default action of Needs Review. This can be overridden to IGNORE.
	
	Using Custom Difference Types provides an alternative mechanism to specifying either Match States or Session Option Properties, such as Lock Source for Difference and Ignore Merge Points, for example. Specifically, the values of the difference type actions and session properties from Custom Difference Types override anything equivalent specified through the `vcmutility` command line.
	
	Besides supporting tracking all configuration options through the Server Administration Tool, using this parameter also has the added advantage of simplifying and minimizing the `vcmutility` parameter set.
	
	Below is an example of content in the `vcmutilityoptions.txt` file that specifies the following: run a promote from StarDraw Release 1.0 Maintenance to StarDraw in project StarDraw using a custom merge type called `vcmutilityConfig`. The custom merge type `vcmutilityConfig` was first created and saved using the Server Administration Tool.
	
	`Server Administrator:Administrator@localhost:49201`
	`Project StarDraw`
	`MergeType Promote`
	`TargetView StarDraw`
	`SourceView StarDraw/Release 1.0 Maintenance`
	`CustomDifferenceTypes vcmutilityConfig`
	`CommitMerge False`

DefaultAction	`DA`
	
	`DefaultAction [MergeType <merge type>] [ItemType <item type>] <match state> <action>`
	
	Specifies a default `<action>` for items that are compared and meet the conditions specified in the given `<match state>`. The VCM utility uses a rules-based "decision table" to determine what action, if any, should be taken when it finds item differences between the source and target views. The `DefaultAction` option allows the default rules to be overridden. This option can be specified multiple times to change the default action for multiple differences. However, the order of definition is important: if two overrides are both applicable to an item difference found in the compare phase, the last override specified takes precedence over the prior one.
	
	• If `MergeType` is specified, the `DefaultAction` only applies to VCM sessions of the specified `<merge type>`: Rebase, Promote, or Replicate.
	• If `MergeType` is not specified, the `DefaultAction` applies to the current VCM session.
	
	Specifying a `DefaultAction` with a different `<merge type>` than that of the current session allows rules used by different VCM sessions to be specified in a single options file.

Session Options	Description

	If `ItemType` is specified, the `DefaultAction` applies only to items of the specified `<item type>`: CRs, Files, Folders, Requirements, Tasks, or Topics. By default, a `DefaultAction` applies to items of all types.
	
	The `<match state>` determines the conditions that must be met by the source and/or target items during comparison. A `<match state>` consists of one or more source/target `<item condition>` definitions, each of which has a `<condition name>` (for example,`source.moved`), and a `<condition value>` (`True`, `False`, or `Unspecified`). The `<condition value>` is optional and defaults to `True`. A `<match state>` is the union of all the conditions defined for it.
	
	The `<action>` determines how to handle source/target item pairs whose differences match the `<match state>`. The `<action>` merely defines the default action for matching items; the actual action can be changed after compare in the StarTeam Cross-Platform Client.
	
	Some example `DefaultAction` definitions are shown below:
	
	`//When a source item has moved, but the target item has not,`
	`//ignore the move.`
	`DefaultAction source.moved target.moved false Ignore`
	
	`//In a Rebase, if a file is binary and has been modified in both the`
	`//source and target, overwrite the target with the source version.`
	`DefaultAction MergeType Rebase`
	`items.binaryfile`
	`source.modified`
	`target.modified`
	`Overwrite`
	
	`//In a Promote, if a CR has moved in both the source and target views`
	`//(to different folders), move the target item to the matching folder as`
	`//the source item, but only if the CRs are on the same branch.`
	`DefaultAction MergeType Promote ItemType CR`
	`source.moved`
	`target.moved`
	`items.branched false`
	`Move`
	
	`DefaultAction` is ignored for Compare sessions.

DefaultComment	`DC`
	
	`DefaultComment <comment>`
	
	Specifies the default revision comment to be used for new item revisions created in the target view. The `<comment>` is a free-form text string. Within the comment value, all white space sequences, including line breaks (CRs and LFs), blanks, and tabs, are converted into a single blank for each occurrence. By default, an auto-generated comment is used as the default revision comment for new item revisions. To disable the use of a default revision comment, specify the `DefaultComment` option with an empty value.

Session Options	Description
	`DefaultComment` is ignored for Compare sessions.
DifferenceReport	`DR <pathToReportfile>` Specifies a path and file name to which the difference report will be written

Exclude	Exclude the specified folders from the source scope. Only folders explicitly specified in `Exclude <files>` or `Exclude <folders>` are excluded. Consequently, an `Exclude <folders>` option can be used to "prune" unwanted folders from the source scope.
	For Example:
	`// suppose we decide to explicitly include CRs and files in all folders below /a/b/`
	`Include /a/b/ +all CRs Files`
	
	`//But we want to exclude CRs in folder /a/b/c/`
	`Exclude /a/b/c/ CRs  OR`
	`Exclude /a/b/c/ +all CRs`
	
	`//But if this CR is in folder /a/b/c/, it is still included`
	`Include CR 12345`
	Regardless of declaration order, `Exclude` options are processed after `Include` options.
	Note: `+all` specifies the inclusion or exclusion of all descendant folder trees. The folder path specified is always relative to the root folder, but DOES NOT contain the root folder name. For instance, if the root folder is `StarDraw` and the folder tree is `Source Code/External Resources`, then the `Include` or `Exclude` syntax must be specified as
	`/Source Code/External Resources/`
	It cannot be specified as
	`/StarDraw/Source Code/External Resources/`
	Attempting to do so will result in a syntax error being reported by `vcmutility`.

Include and Exclude Semantics	If no `Include` options are specified, the default VCM session scope is implicitly "all files in the source view". This is equivalent to explicitly specifying `include /* +all`. If at least one `Include` option is specified, the scope is explicitly limited to those items selected by `Include` statements. In both implicit and explicit scopes, all selected source items are pruned by any `Exclude` options. All `Include` and `Exclude` options must identify objects (labels, files, CRs, and so on) in the source view. Also, selection type names can be singular or plural (`RevLabel`, `CR`, and so on), even if multiple values are provided. **Note:** `Exclude` options are always processed after `Include` options, regardless of declaration order. Therefore,

Session Options	Description
	`Exclude /src/foo/bar/` followed by `Include /src/foo/ +all` causes folder `/src/foo/bar/` to be excluded.

Export	`Exp`
	`Export <VCM exchange file>`
	The `Export` option specifies that all the VCM session information, including merged result files, are to be combined and stored in the given `<VCM exchange file>`. The exchange file name is always suffixed with a `.vcmx` extension. A VCM exchange file allows the entire VCM session to be transported to another machine, allowing that machine to perform an `Import` command, which resumes the session. (See the `Import` command for more information.)
	If the `<VCM exchange file>` does not contain path information, it is saved in the user's home directory (what Java identifies as `user.home`).
	Note: The `Export` option always causes the VCM exchange file to be created, even when the session itself is not saved. See the `Save` option for more information.

FixFloatingChildShares	`FFCS`
	`FixFloatingChildShares [True | False]`
	Specifies whether, in Rebase and Replicate merge operations, each target view item found that is a floating share of a source view item should be “fixed” by pinning it. When a target view item is a floating child share of a source item (which implies that the target item has not branched), differences will not be detected between the source and target item during VCM sessions because changes to the source item immediately float to the child item. VCM best practices suggest that child shares should always be pinned, allowing changes to propagate from the source to target view in a controlled manner. This option allows floating child items found by VCM to be “fixed” by pinning them to the parent item revision. Specifying this option has a performance cost due to the extra commands required to check each target item examined during the compare phase.

IgnoreMergePoints	`IMP`
	`IgnoreMergePoints [True] | [False]`
	Specifies whether merge points should be ignored during the comparison phase. If `True`, items with merge conflicts use their branch point as the common ancestor instead of the source revision of the last merge point.

Include	`Inc`
	`Include {<change requests> | <files> | <folders> | <process items> | <requirements> | <revision labels | <tasks> | <topics> }`
	Includes the specified items in the source scope. The `Include` option can be provided multiple times, causing all selected items to be included. Only one item selection type (revision labels, change requests, and so on) can be specified with each `Include` option. The selection type keyword,

Session Options	Description

	which is optional for files and folders, can be singular or plural, for example, `ProcessItem` or `ProcessItems`.
	
	Examples:
	`Include CRs ALL`
	`Include /src/com/*.java +all *.jar +2 *.jpx Buildnumber.h`
	`Include Folders /docs/api/ +all`
	`Include ProcessItem CR 451`
	`Include Reqs 4515 4516`
	`Include RevLabel "Beta Fix 12.413"`
	`Include Topic 14512`
	`Include Task 413`

LockMergeConflicts	`LMC`
	
	`LockMergeConflicts {None | Source | Target | Both}`
	
	Specifies that items with unresolved conflicts are to be locked exclusively in either the Source, Target, or Both views. Locks are acquired in the compare phase. `None` is the default, which specifies that no locks are to be created for items with unresolved locks. Note that locks are only applied to source and/or target items for which differences are found. Locks are not applied to items that are compared for which no differences are found. Also, note that this option is not affected by the Project option **Require exclusive comment when files are checked in** nor the client workstation option **Exclusively lock files on check-out**. Those options are properly handled by the VCM engine. `LockMergeConflicts` is ignored for Compare sessions.

ManualMergeFiles	`MMF`
	
	`ManualMergeFiles [True | False]`
	
	If `True`, this causes the file merge tool configured for the workstation to be launched for each source/target file pair found in a content merge state.
	
	The `ManualMergeFiles` option can be used in conjunction with `AutoMergeFiles`:
	
	• If a merge conflict is detected and `AutoMergeFiles` is requested, an auto-merge attempt is made first.
	• If the conflict is resolved, the merged result file is saved, and a manual merge is not needed.
	• If the auto-merge is not successful, or if `AutoMergeFiles` has not been requested, then if `ManualMergeFiles` is `True`, a manual file merge is performed.
	
	Note: `ManualMergeFiles` is ignored (and a warning is displayed) if the workstation has no manual merge tool configured. Also, if the manual merge tool cannot be launched, or returns an error condition, the affected file remains in an unresolved conflict state. `ManualMergeFiles` is ignored for Compare sessions.

Match	`Match [Folder] *{<folder path> to <folder path>}`

Session Options	Description

	Specifies that for comparison purposes, the folder specified in the first <folder path>, which must reside in the source view, should match the second <folder path>, which must reside in the target view. The Match option is sometimes needed to prevent “ambiguous match” conditions, which can occur when one of the views is a non-derived view. Typically, the Match option is only needed to match the source and target view root folders. However, other folders can be matched to resolve other ambiguous match conditions reported by the compare phase.
	
	Both the source and target <folder path> must begin and end with a forward slash ("/").
	
	By convention, the root folder is represented by a single "/". This means that the root folder name should not be provided in folder paths. For example, if the root folder is named “StarDraw”, the folder path for the immediate child folder “Source Code” is simply "/Source Code/".
	
	Examples:
	`// Force the source and target root view folders to match.`
	`Match / to /`
	
	`//Force the source view folder "/Source Code" to match the target view`
	`//folder "/Modules/Materials/src".`
	`Match "/Source Code/" to "/Modules/Materials/src/"`

MergeType	Type MT MergeType {Compare | Rebase | Promote | Replicate} Specifies whether to perform a Compare session or a Rebase, Promote, or Replicate merge session. If only a SourceView is specified, MergeType defaults to Promote. If only a TargetView is specified, MergeType defaults to Rebase. If both SourceView and TargetView are specified, MergeType must be specified. For a Compare session, the source and target views can be the same.
Name	Na Name <Change Package name> Specifies the name of the change package associated with the VCM session. For servers that support change packages, a name is automatically chosen when a change package is created by saving or committing the session. This option allows a specific name to be used instead of the default name. However, the name must be unique from all other change package names already saved or committed for the target view, otherwise the save or commit action will fail. When the Name option is used in conjunction with the Open command, the opened change package is renamed to the given value. Also see the Save and CommitMerge options.
PostCommitLabel	PostCL PostCommitLabel <label>

Session Options	Description
	If the VCM session is committed, the given *view* `<label>` is created in the target view after all updates are performed. The label reflects the revisions of all target view items used during the compare phase, *modified* by the changes made by the commit phase. This means the label contains new items, new item revisions, and item moves, but items deleted by the commit will be detached from the label. The post-commit label is essentially identical to the "pre-merge view". `PostCommitLabel` is ignored for Compare sessions. By default, a post-commit view label is created with a default name. To disable the post-commit view label, specify `PostCommitLabel` with a blank value (that is, `" "`).

PostCommitRevLabel	`PostRL`
	
	`PostCommitRevLabel <label>`
	
	If the VCM session is committed, the given *revision* `<label>` is created in the target view, and all items modified by the VCM session, except for deleted items, are attached to it. Consequently, the label contains items that were added, moved, re-pinned, or updated in any other way (except for deletion) by the VCM session. `PostCommitRevLabel` is ignored for Compare sessions.
	
	By default, a post-commit revision label is not created.

PreCommitLabel	`PreCL`
	
	`PreCommitLabel <label>`
	
	The given view `<label>` is created in the target view, reflecting the snapshot used in the compare phase. The label reflects the revisions of all target items used during the compare phase. `PreCommitLabel` is ignored for Compare sessions.
	
	By default, a pre-comimt view label is not created.

PreCommitRevLabel	`PreRL`
	
	`PreCommitRevLabel <label>`
	
	If the VCM session is committed, the given revision `<label>` is created in the target view, and all non-ignored target view items are attached to it in their “before” state. That is, target view items to be modified by the session are attached to the revision label before they are modified. This means that items to be added (for example, shared) to the target view will **not** be attached, but items to be deleted **will** be attached. `PreCommitRevLabel` is ignored for Compare sessions.
	
	By default, a pre-commit revision label is not created.

PreventDuplicateFilenames	`PDF`
	
	`PreventDuplicateFilenames [True | False]`
	
	If `True`, it specifies that sharing a new file to the target view is not allowed if it results in two identically-named files to exist in the same folder.

Project	`Pro`
	
	`Project <project>`

Session Options	Description
	Specifies the project to be used in the VCM session. This option is required. The source and target views must belong to the same <project>. Project names are case-insensitive.
ReportDiffs	RD ReportDiffs [True | False] If True, causes a report to be generated listing item differences found in the compare phase. The difference report is generated in the user's home directory (what Java identifies as user.home) with the following title: VCMDiffReport-YYYY-MM-DD_hh-mm-ss.html where YYYY-MM-DD and hh-mm-ss are the current date and time in the local time zone.
ReportUpdates	RU ReportUpdates [True | False] If True, causes a report to be generated listing all changes made to the target view in the commit phase. The update report is generated in the user's home directory (what Java identifies as user.home) with the following title: VCMUpdateReport-YYYY-MM-DD_hh-mm-ss.html where YYYY-MM-DD and hh-mm-ss are the current date and time in the local time zone. ReportUpdates is ignored for Compare sessions.
Save	Save [<VCM session file>] Specifies that the VCM session is to be saved. By default, uncommitted VCM sessions are automatically saved to a VCM session (.vcms) file with a default name using the format: <user home>/VCMSession-YYYY-MM-DD_hh-mm-ss.vcms where YYYY-MM-DD_hh-mm-ss is the date and time when the session is saved. The folder <user home> is the user's home directory. If the Save option is specified with a <VCM session file> name, an uncommitted session is saved with the given file name instead of the default name. If needed, .vcms is appended to the name. If the given file name does not contain path information, the session file is stored in the user.home folder. A .vcms file contains VCM session metadata, but not the contents of merged files. Merged file contents are stored in a user-relative temporary folder, referenced by elements in the session file. Consequently, a .vcms file can only be used to resume the VCM session on the same workstation. (See the Resume command.) When the Save option is specified without a file name, an attempt is made to save an uncommitted VCM session as an active change package in the target view. The change package is saved with the default or user-specified name (see the Name option). A VCM session saved as a change package can later be resumed on any workstation using the Open option. However, if the server does not support change packages or a server-side

Session Options	Description
	save is unsuccessful, the session is instead saved to a `.vcms` file with a default file name as described above. When a commit is successfully performed, the `Save` option is ignored. If the server supports change packages, the committed session creates a Committed change package using the default or user-specified name (see the `Name` option). If a `.vcms` file was previously created, it is deleted along with all merged result files created by the VCM session. Also see the `Export` option.

SourceLabel	`SrcLabel`
	
	`SL`
	
	`SourceLabel <label>`
	
	Requests the source view to be used as of a given view label. Label names are case-insensitive. Only one of `SourceLabel`, `SourceState`, and `SourceTime` can be specified. If none of these options is specified, the option `SourceTime Now` is implicitly used.

SourceState	`SrcState`
	
	`SS`
	
	`SourceState <state>`
	
	Requests the source view to be used as of a given view promotion state. Promotion state names are case-insensitive. Only one of `SourceLabel`, `SourceState`, and `SourceTime` can be specified. If none of these options is specified, the option `SourceTime Now` is implicitly used.

SourceTime	`SrcTime`
	
	`ST`
	
	`SourceTime {<timestamp> | Now}`
	
	Requests the source view to be used as of a given timestamp. The keyword `Now` causes a snapshot of the current time to be used as configuration timestamp. Only one of `SourceLabel`, `SourceState`, and `SourceTime` can be specified. If none of these options is specified, the option `SourceTime Now` is implicitly used.

SourceView	`Source`
	
	`SV`
	
	`SourceView <view>`
	
	Specifies the source view to be used in the VCM session. If more than one view within the project has the same `<view>` name, a slash-separated "view path" can be provided (for example, `MainView/ChildView/GrandchildView`). If a view name contains embedded slashes, it must be enclosed in quotes.
	
	`SourceView` is optional for Rebase merges; if specified, it must be the parent of the target view.
	
	  **Note:** View names are case-insensitive.

Session Options	Description

TargetLabel	`TgtLabel`
	`TL`
	`TargetLabel <label>`
	Requests the target view to be used as of a given view label. `TargetLabel` can only be used for Compare sessions. Label names are case-insensitive. Only one of `TargetLabel`, `TargetState`, and `TargetTime` can be specified. If none of these options is specified, the option `TargetTime Now` is implicitly used.

TargetState	`TgtState`
	`TS`
	`TargetState <state>`
	Requests the target view to be used as of a given view promotion state. `TargetState` can only be used for Compare sessions. Promotion state names are case-insensitive. Only one of `TargetLabel`, `TargetState`, and `TargetTime` can be specified. If none of these options is specified, the option `TargetTime Now` is implicitly used

TargetTime	`TgtTime`
	`TT`
	`TargetTime {<timestamp> | Now}`
	Requests the target view to be used as of a given timestamp. `TargetTime` can only be used for Compare sessions. The keyword `Now` causes a snapshot of the current time to be used as configuration timestamp. Only one of `TargetLabel`, `TargetState`, and `TargetTime` can be specified. If none of these options is specified, the option `TargetTime Now` is implicitly used

TargetView	`Target`
	`TV`
	`TargetView <view>`
	Specifies the target view to be used in the VCM session. If more than one view within the project has the same `<view>` name, a slash-separated "view path" can be provided (for example, `MainView/ChildView/GrandchildView`). If the view name contains embedded slashes, it must be enclosed in quotes.
	`TargetView` is optional for Promote merges; if specified, it must be the parent of the source view. For Compare sessions, the target view can be the same as the source view.
	Note: View names are case-insensitive.

UpdateReport	`UR <pathToReportfile>` Specifies a path and file name to which the update report will be written

Resumed Session Options

So that the same options file can be specified for a `Resume` command, all options allowed for new sessions can also be specified for resumed sessions. However, most options, if re-specified, are ignored because they cannot be modified once the session has been started. The only exceptions are the options specifically outlined below:

Connection options	Since connection information (server address and port, userid, and password) are not persisted in the VCM session file, connection information must be re-specified for resumed sessions. However, a resumed session will fail if it is not reconnected to the same StarTeam server of if a different user is used that has permission conflicts with the views or items used in the VCM session.
CommitMerge	This option will commonly be specified to `True` in a resumed session. This allows the original VCM utility execution to be used as a compare-only run and a second VCM utility execution to be used as a commit run.
ReportDiffs	This option can be specified in a resumed session. If true, a difference report is created before the commit phase, if any.
ReportUpdates	This option can be specified in a resume session. If true and the commit phase is successfully performed, all changes made to the target view are reported.
CheckoutPreview	Normally, if `CheckoutPreview` was specified in the original VCM session, the “merge preview” check-out operation is performed in the resumed session with the same options as before. However, if `CheckoutPreview` is specified in the resumed session, it overrides the original option and causes files to be checked-out in the resumed session according to the new settings.
Description	If specified, this option overrides the default or previously-provided Change Package description text. The new description text is used for the new Change Package revision created when the VCM session is saved or committed.
ManualMergeFiles	Normally, if `ManualMergeFiles` was specified in the original VCM session, and the session is saved with unresolved file merge conflicts, the manual file merge phase will be performed again when the session is resumed. However, if `ManualMergeFiles` was not specified in the original VCM session, it can be specified as `True` in the resumed session to invoke the manual merge phase. Alternatively, it can be specified as `False` in a resumed session to prevent the manual merge phase.
PostCommitLabel, PostCommitRevLabel, PreCommitLabel, and PreCommitRevLabel	If any of these label options are specified in a resumed option, they override the previous value for the corresponding label. When a label option is set to a blank (" "), the corresponding label option is disabled and will not be created in the commit phase.

VCMUtility Miscellaneous Options

This section defines `VCMUtility` miscellaneous options that are not saved in view compare/merge sessions.

NetMon

`NM`

```
NetMon [True | False]
```

Enables the SDK net monitor feature. Each command issued by the `VCMUtility` to the StarTeam Server is logged to the console window (but not the `VCMUtility` log file).

Time

`T`

`Time [True | False]`

Causes timing information to be displayed for each phase of the VCM session performed. Timing information is written to both the console window and the `VCMUtility` log file.

Verbose

`vb`

`V`

`Verbose [True | False]`

Causes additional diagnostic and progress information to be displayed to the console (standard output) and to the `VCMUtility` log file during execution.

VCMUtility Examples

This topic presents examples of using the `VCMUtility` for various types of merges.

Hello World Rebase

Below are the options for the "Hello World" equivalent of a `VCMUtility` Rebase run:

```
Type     Rebase
Project     Hello
Target     World
```

Automatic Rebase

The options file below performs the same Rebase as in the previous example, but it commits if possible and provides detailed reporting on the results:

```
Type                Rebase
Project             Hello
Target              World
CommitMerge         True
LockMergeConflicts  Both

// All of these options are set to True:
AutoMergeFiles
BreakLocks
ReportDiffs
ReportUpdates
```

All files are auto-merged both in content and properties. Files that are in conflict but cannot be resolved are locked in both the source and target views. Existing lock conflicts are broken if possible. If no unresolved conflicts are encountered, the session is committed. Details of both the compare phase (differences) and commit phase (updates) are reported. If the commit is successful, all VCM session temporary files are deleted.

Promote by View Label: Compare Only

The options below perform a compare-only promote of files and CRs as of a view label, saving the session in a specific session filename:

```
// Connection settings
Server            MyUserid@ProdServer:4000
PwdFile            MyPassword.txt

// Merge type and view configuration
Type            Promote
Project            StarDraw
Source            "Beta Release"
SrcLabel        Build-4.0_142

// Select all files and CRs as source items
include            /* +all
include            / +all CRs

// Compare-only, report, and save with a specific session filename
CommitMerge        False
save            Build-4.0_142-Promote
ReportDiffs

//Miscellaneous options
AutoMergeFiles        True
AutoMergeProperties    False // leave these as conflicts and merge manually
LockMergeConflicts    Target
```

Promote by View Label: Merge

The VCM utility command-line below resumes the session saved in the previous example and commits it, assuming no new conflicts have occurred.

```
VCMUtility -resume Build-4.0_142-Promote -CommitMerge -ReportUpdates
```

Cheat Sheet

`VCMUtility` command-line syntax: `VCMUtility [<options file>] [*<option>]`

Within the `<options file>`, each `<option>` must begin in column 1 but can continue on subsequent lines if those lines begin with a space or tab character. When typing options in the command line, each `<option>` must be preceded with a "`-`".

Options

The information below lists all the `VCMUtility` command-line options and their syntax.

```
<option>
  <command> | <connection option> | <session option> | <miscellaneous option>
```

<command>

```
{{Help | H | ?} [<help topic>]} |
  {Delete <VCM session file>}|
  {Import <VCM archive file} |
  {Open <Change Package name>} |
  {Replay <Change Package name>} |
  {Resume <VCM session file>}
```

<connection option>

```
{{AutoLogon | AL} [True | False]} |
  {{Encryption | Encrypt | En} {None | RC4 | RC2_ECB | RC2_CBC
| RC2_CFB}} |
```

```
  {{PwdFile  | PF} <file name>} |
  {{Server | S} [<user>[:<password>]@]<host>[:<port>]} |
  {{UseCA | UCA} {<host>:<port> | AutoLocate}} |
  {{UseServerProfile | USP} [True | False]}
```

<session option>

```
{{AutoMergeFiles | AMF} [True | False]} |
  {{AutoMergeProperties | AMP} [True | False]} |
  {{BreakLocks | BL} [True | False]} |
  {{CaseSensitiveFilenames | CSF} [True | False]} |
  {{CheckoutPreview | check-out | CP} <files> [<check-out
opotions>]} |
  {{CommitMerge | Commit | CM} [True | False]} |
  {{DefaultAction | DA} [MergeType <merge type>] [ItemType
<item type>] <match state> <action>} |
  {{DefaultComment | DC} <comment>} |
  {{Description |} <description>} |
  {{Exclude | Exc} <folders>} |
  {{Export | Exp} <VCM archive file>} |
  {{FixFloatingChildShares | True | False} |
  {{IgnoreMergePoints | IMP} [True | False]} |
  {{Include | Inc} {<change requests> | <files> | <folders> |
<process items> |
     <requirements> | <revision labels> | <tasks> | <topics>}}
|
  {{LockMergeConflicts | LMC} {None | Source | Target | Both}}
|
  {{ManualMergeFiles | MMF} [True | False]} |
  {{Match [Folder] *{<folder path> to <folder path>}} |
  {{MergeType | Type | MT} {Compare | Rebase | Promote |
Replicate}} |
  {(Name |Na} <Change Package name>} |
  {{PostCommitLabel | PostCL} <label>} |
  {{PostCommitRevLabel | PostRL} <label>} |
  {{PreCommitLabel | PreCL} <label>} |
  {{PreCommitRevLabel | PreRL} <label>} |
  {{PreventDuplicateFilenames | PDF} [True | False]} |
  {{Project | Pro} <project>} |
  {{ReportDiffs | RD} [True | False]} |
  {{ReportUpdates | RU} [True | False]} |
  {Save [<VCM session file>]} |
  {{SourceLabel | SrcLabel | SL} <label>} |
  {{SourceState | SrcState | SS} <state>} |
  {{SourceTime | SrcTime | ST} {<timestamp> | Now}} |
  {{SourceView | Source | SV} <view>} |
  {{TargetLabel | TgtLabel | TL} <label>} |
  {{TargetState | TgtState | TS} <state>} |
  {{TargetView | Target | TV} <view} |
```

<miscellaneous option>

```
{{NetMon | NM} [True | False]} |
  {{Time | T} [True | False]} |
  {{Verbose | Vb | V} [True | False]} |
```

Other Syntax Elements

The table below lists other syntax elements in alphabetical order:

<action>

```
Delete | DeleteAndReverseShare | Fail | Ignore | Merge |
Move | MoveAndMerge | MoveAndRepin | NeedsReview |
Overwrite | Repin | RepinAndMove |
 ReverseShare | Share
```

<change requests>	{CR | CRs | ChangeRequests} {ALL | *<CR #>}
<Change Package name>	{A name consisting of one or more characters}
<check-out options>	[+cwf] [+eol {on | off | cr | lf}] [+filter {CGIMOU}] [+o] [+ro] [+rp <work folder path>]
<condition name>	items.binaryfile | items.branched | items.samecontent | source.childshare | source.deleted | source.floating | source.modified | source.moved | source.present | source.rootbranch | target.childshare | target.deleted | target.floating | target.modified | target.moved | target.present | target.parentdeleted | target.rootbranch
<condition value>	True | False | Unspecified
<files>	[File | Files] {ALL | *{<file name pattern>} [+<depth>]}}
<folder path>	{A slash followed by an optional series of folder names each ending with a slash}
<folders>	[Folder | Folders] {ALL | *{<folder path>} [+<depth>] *[<item type>]}}
<item condition>	<condition name> [<condition value>]
<item type>	{ChangeRequest | CR | ChangeRequests | CRs} | {File | Files} | {Folder | Folders} | {Requirement | Req | Requirements | Reqs} {Task | Tasks} {Topic | Topics}
<match state>	*<item condition>
<process items>	ProcessItems *{[View <view>] CR <CR #> | [View <view>] Req <Req #> | [View <view>] Task <Task #>}
<requirements>	{Requirement | Req | Requirements | Reqs} {ALL | *<Req #>}
<revision labels>	RevLabels *<label>
<task>	{Task | Tasks} {ALL | *<Task #>}
<timestamp>	Example formats: "3/11/06 1:32 PM" "Mar 11, 2006 1:32:38 PM" "March 11, 2006 1:32:38 PM PST" "Saturday, March 11, 2006 1:32:38 PM PST"
<topics>	{Topic | Topics} {ALL | *<Topic #>}
<VCM exchange file>	A .vcms file name

<VCM session file>	`A .vcms file name`

Syntax for VCMUtility Compound Options

<action>

Specifies the action to perform for a given source/target item difference. An `<action>` is one of the following mnemonics:

Mnemonic	Description
Delete	Delete the target item.
DeleteAndReverseShare	Equivalent to a `Delete` followed by a `ReverseShare`.
Fail	Synonym for `NeedsReview` (see below).
Ignore	Take no action.
MarkResolved	Create a merge point only that marks the source and target items as resolved.
Merge	Merge the source and target items.
Move	Move the target item to the equivalent folder as the source item.
MoveAndMerge	Equivalent to a `Move` followed by a `Merge`.
MoveAndOverwrite	Equivalent to a `Move` followed by an `Overwrite`.
MoveAndRepin	Equivalent to a `Move` followed by a `Repin`.
NeedsReview	Force a review before a commit. That is, do not allow commit while this action is selected. Item differences with this action are conflicts, therefore, their action must be changed to something else.
Overwrite	Overwrite the target with the contents of the source.
Repin	Change the revision to which the target is pinned to match the source item.
ReverseShare	Move the source item to the target view and share it back to the source view.
Share	Share the source item to the target view.

Note: Not every `<action>` is valid for every item difference For example, `Delete` is not valid when the target item is already deleted.

<check-out options>

The following section describes the syntax used for the compound `VCMUtility` option `<check-out options>`.

```
[+cwf] [+eol {on | off | cr | lf | crlf}] [+filter {CGIMOU}] [+o] [+ro] [+rp
<work folder path>]
```

Specifies non-default check-out options. The available check-out options are similar to those provided by the StarTeam command-line (stcmd), except that option names must be prefixed with a ‘ **+** ’ sign. The available options are detailed below.

+cwf	Requests the creation of working folders for all specified folders, even if they do not have files to be checked-out by this run. Only visible folders are created.

+eol <eol option>	Requests conversion of all end-of-line delimiters for text files to the specified format. An <eol option> of on uses the client-configured EOL format. off prevents any EOL conversion. cr, lf, and crlf cause each EOL to be converted to a carriage-return, line-feed, or carriage-return/line-feed pair, respectively. Note that text files with a "fixed" EOL format are always converted to the specified format.
+filter	+filter {CGIMOU} Specifies the status of files to consider for check-out: **C**urrent, mer**G**e, m**I**ssing, **M**odified, **O**ut-of-date, or **U**nknown. Multiple status flags can be combined. If +filter is not specified, the default filter is IO (**Missing** and **Out-of-date**). If **Merge**, **Merge**, or **Unknown** files are included without the +o option, a warning is generated for each such file, and the file is not checked out.
+o	Specifies that, in addition to **Missing** and **Out-of-date** files, files whose status is **Modified**, **Merge**, or **Unknown** are included. Furthermore, all files are overwritten without warning. If +filter is also specified, only the specified files are checked out.
+ro	Sets each file to read-only after check out. By default, checked-out files are read-write.
+rp	Specifies the root working folder of the "merge preview". Files are checked-out to child working folders relative to <work folder path>.

<change requests>

<change requests> {CR | CRs | ChangeRequests} {ALL | *<CR #>}

Specifies all change requests in the view, or individual change requests by change request number. CRs and ChangeRequests are synonyms. The singular form of each is also accepted.

<files>

[File | Files] {ALL | *{<file name pattern> [+depth>]}}

Specifies all files in the view or a set of specific files, given as a list of file names and/or patterns, each with an optional folder <depth>. The keyword File (or Files) is optional unless the keyword All is used. A <file name pattern> can be a specific file name (for example, foo.java), a file name pattern (for example, *.java), or a file name or pattern with a folder path (for example, (/src/com/acme/foo.java) or /src/com/acme/*.java).

Usage

Folder paths must use forward slashes; a single slash (/) is a synonym for the root folder. (Consistent with other StarTeam utilities, the root folder name, which typically matches the view name, should not be provided in path names.)

- If a filename or pattern is provided without a folder path, the implied folder is the same as the previous <file name pattern> parameter.
- If the first <file name pattern> parameter does not contain a folder path, the root folder is implied.
- If provided, the folder <depth> specifies the number of child folder levels below the specified folder to include; it can be a number or the keyword All.
- If a file or pattern name contains spaces, it must be enclosed in quotes.

Examples

Below are examples of <files> usage:

```
// all files in the view
```

```
include Files ALL

//foo.java and bar.java in folder /src/com/acme
include /src/com/acme/foo.java bar.java

// all .java files in folder /src/com/acme and below
include /src/com/acme/*.java +all

// all .txt files in the root folder, all .zip file in first-level
// child folders, and a specific readme.txt file
include *.txt *.zip +1 /docs/acme/readme.txt
```

<folders>

`[Folder | Folders] {ALL | *{<folder path> [+<depth>] *[<item type>]}}`

Specifies all folders in the view or specific folder paths, optionally indicating a folder depth and specific item types. The keyword `Folder` or (`Folders`) is optional unless the keyword `ALL` is used.

Usage

A valid `<folder path>` must begin and end with a forward slash (`/src/com/`). If provided, the `<depth>` specifies the number of child folder levels below the specified folder to include; it can be a number, or the keyword `All`.

- If a folder path contains spaces, it must be inclosed in quotes.
- If no `<item type>` parameters are provided, only files are included in the specified folder(s). Otherwise, all items of the specified item types are included.

Recognized item types are `CRs`, `Files`, `Folders`, `Tasks`, `Topics`, and `Requirements` (singular or plural).

Examples

Below are examples of <folder> usage:

```
// all folders in the view
include folders ALL

// all files in the folder /src/com/acme/ alone
include /src/com/acme/

// all files and tasks in  /src/ and below
include /src/ +all files tasks

// all CRs in the folder "/triage/" and all files in "/PR docs/"
// child folders two levels below it
include /triage/ CRs "/PR docs/" +2
```

By convention, the root folder is represented by a single "`/`". This means that the root folder name should not be provided in folder paths. For example, if the root folder is named "`StarDraw`", the folder path for the immediate child folder "`Source Code`" is simply `/Source Code/`.

<item type>

`<item type>`

Specifies an item type. Allowed values are `ChangeRequest` (or `CR`), `File`, `Folder`, `Requirement` (or `Req`), `Task`, and `Topic`. Item type names are case-insensitive and can be plural.

<match state>

`*<item condition>`

Defines a set of conditions that apply to source/target item differences. A `<match state>` is the union of each `<item condition>` defined for it. Each `<item condition>` has the form:

`<condition name> [<condition value>]`

<condition name>

The valid `<condition names>` and their meaning are:

<condition name>	Meaning
items.binaryfile	Indicates whether either of the items in question is a binary file.
items.branched	Indicates whether the source and target items are in different branches of the object version tree.
items.samecontent	Indicates whether the source and target items have the same user-modifiable properties and, for files, data content.
source.childshare	Indicates whether the source item is a child share of the target item.
source.deleted	Indicates whether the item in question is deleted in the source view.
source.floating	Indicates whether the source item has a floating configuration.
source.modified	Indicates whether the item in question is modified in the source view.
source.moved	Indicates whether the item in question is moved in the source view.
source.present	Indicates whether the item in question is present in the source view.
source.rootbranch	Indicates whether the source item is the root branch of its share tree.
target.childshare	Indicates whether the target item is a child share of the source item.
target.deleted	Indicates whether the item is question is deleted in the target view.
target.floating	Indicates whether the target item has a floating configuration.
target.modified	Indicates whether the item in question is modified in the target view.
target.moved	Indicates whether the item in question is moved in the target view.
target.parentdeleted	Indicates whether the target item's folder has been deleted.
target.present	Indicates whether the item in question is present in the target view.
target.rootbranch	Indicates whether the target item is the root branch of its share tree.

<condition value>

The valid `<condition value>`s are:

<condition value>	Meaning
`True`	The condition is true for the applicable item(s).
`False`	The condition is false for the applicable item(s).
`Unspecified`	The condition is unknown or not relevant for the applicable item(s).

The `<condition value>` is optional and defaults to `True`. For any given `<match state>`, all unspecified conditions are initially `Unspecified`.

An <item condition> can be defined as True or False to cause the corresponding condition to "participate" in matching the condition to actual item differences.

A condition can be defined as Unspecified, for example, to experimentally remove the condition from the matching criteria without deleting the condition from an options file.

Note: Some conditions are mutually exclusive: if defined together, they will never match any actual item differences. For example, a source item cannot be both present (source.present=true) and deleted (source.deleted=true).

<process item>

ProcessItems *{[View <view>] CR <CR #> | [View <view>] Req <Req #> | [View <view>] Task <Task #>}

Specifies a set of process items (change requests, tasks, and/or requirements) to be included. Specifying a process item causes items linked to it in the source view to be included as well. The keyword ProcessItems can be singular. The full names ChangeRequest and Requirement can be used in place of CR and Req respectively.

By default, a process item specified must reside in the source view. However, the optional prefix View <view> can be used to select a process item in a view other than the source view. When a non-source view process item is included, the process item is **not** included in the source scope, but those items linked to it in the source view are included. The specific revision of each source view item linked to the process item is included.

Examples

```
// Include CR #451 in the source view and its linked items
include ProcessItem CR 451

//Include the items in the source view that are linked to Task #909
//includle Requirement #518, both from view "Triage"
//include ProcessItem View Triage Task 909
                    View Triage Requirement 518
```

Note: If the view name contains spaces, it must be quoted ("Release 4.3"). If more than one view in the project has the same view name, the view name can be a slash-separated view path ("Apps/Releases/Release 4.3").

<requirements>

{Reqs | Requirements} {ALL | *{<Req #>}

Specifies individual requirements by requirement number. Reqs and Requirements are synonyms; the singular form of each is also accepted.

<revision labels>

RevLabels *<label>

Specifies all the items attached to each specified revision label (<label>). The keyword RevLabels can be singular. Revision labels are case-insensitive.

<tasks>

Tasks {ALL | *{<Task #>}

Includes the specified individual tasks by task number. The keyword Tasks can be singular.

<timestamp>

A `<timestamp>` must have one of the Java-recognized formats for date and time strings.

- Date formats are interpreted with the local date formatting conventions (for example, `3/11/06` is interpreted as March 11, 2006 in the United States.)
- Seconds are optional (for example, `1:32` and `1:32:00` are identical).
- The `AM/PM` indicator is required.
- The time zone indicator is optional; if omitted, the local time zone is assumed.
- The day of week, if provided, is ignored.

Examples:

```
"3/11/13 1:32 PM"
"Mar 11, 2013 1:32:38 PM"
"March 11, 2013 1:32:38 PM PST"
"Saturday, March 13, 2013 1:32:38 PM PST"
```

<topics>

`Topics {ALL | *{<Topic #>}`

Includes the specified individual topics by topic number. The keyword `Topics` can be singular.

Index

C

I

S

U

V