Clipboard

Jump to bottom

Nicholas Marriott edited this page on Jun 15, 2022 · 53 revisions

The clipboard

It is common to want to have text copied from tmux's copy mode or with the mouse in tmux synchronized with the system clipboard. The tools offered to tmux by terminals to do this are quite blunt and not consistently supported. This document gives an overview of how things work and some configuration examples.

There are two posible methods:

- OSC 52 and the `set-clipboard` option.
- Piping to an external tool like `xsel`.

Note that tmux should be restarted entirely (run `tmux kill-server`) after making changes to `.tmux.conf`.

The `set-clipboard` option

How it works

Some terminals offer an escape sequence to set the clipboard. This is one of the operating system control sequences so it is known as OSC 52.

To skip the details and read quick step-by-step instructions on configuring `set-clipboard`, skip to this section.

The way it works is that when text is copied in tmux it is packaged up and sent to the outside terminal in a similar way to how tmux draws the text and colours and attributes. The outside terminal recognises the clipboard escape sequence and sets the system clipboard.

tmux supports this through the `set-clipboard` option. The big advantage of this is that it works over an *ssh(1)* connection even if X11 forwarding is not configured. The disadvantages are that it is patchily supported and can be tricky to configure.

For `set-clipboard` to work, three things must be in place:

1. The `set-clipboard` option must be set to `on` or `external`. The default is `external`.
2. The `Ms` capability must be available to tmux when it looks at the *terminfo(5)* entry specified by `TERM`. This is present by default for some terminals and if not is added with `terminal-overrides` or `terminal-features` (see the next section).
3. The feature must be enabled in the terminal itself. How this is done varies from terminal to terminal. Some have it enabled by default and some do not.

The following two sections show how to configure `set-clipboard` and `Ms`; later sections cover support in different terminals.

Changing `set-clipboard`

The tmux `set-clipboard` option was added in tmux 1.5 with a default of `on`; the default was changed to `external` when `external` was added in tmux 2.6.

The difference is that `on` both makes tmux set the clipboard for the outside terminal, and allows applications inside tmux to set tmux's clipboard (adding a paste buffer). `external` only makes tmux set the clipboard and forbids applications inside from doing so.

Any of these commands in `.tmux.conf` or at the command prompt will set the three states:

```
set -s set-clipboard on
set -s set-clipboard external
set -s set-clipboard off
```

Setting the `Ms` capability

By default, tmux adds the `Ms` capability for terminals where `TERM` matches `xterm*`. `TERM` can be checked with this command run outside tmux:

```
$ echo $TERM
```

To see if `Ms` is already set, run this from *inside* tmux:

```
$ tmux info|grep Ms:
 180: Ms: (string) \033]52;%p1%s;%p2%s\a
```

If `Ms` is shown like this, it does not need to be added. If it shows `[missing]`, then it must be added with `terminal-overrides` or `terminal-features`. Check what `TERM` is outside tmux:

```
$ echo TERM
rxvt-unicode-256color
```

Then add an appropriate `.tmux.conf` line. For tmux 3.2 or later, this looks like this (change `rxvt-unicode-256color` to the appropriate name from `TERM`):

```
set -as terminal-features ',rxvt-unicode-256color:clipboard'
```

Or for older tmux versions:

```
set -as terminal-overrides ',rxvt-unicode-256color:Ms=\E]52;%p1%s;%p2%s\007'
```

Multiple similar lines may be added to `.tmux.conf` for different values of `TERM` (`-a` means to append to the option). It also supports wildcards, so using `rxvt-unicode*` would apply to both `rxvt-unicode` and `rxvt-unicode-256color`.

Security concerns

If `set-clipboard` is set to `external`, only tmux can set the clipboard. If set to `on` and tmux is version 2.6 or newer, any application running inside tmux can create a tmux paste buffer and set the system clipboard. It doesn't matter if the command is run with *su(1)* or *sudo(1)* - if a command can write text to a tmux pane, it can set the clipboard.

This means that with `set-clipboard` set to `on`, great care must be taken with untrusted commands run inside tmux.

The same applies to any commands run without tmux if OSC 52 is enabled in the terminal.

Terminal support - tmux inside tmux

If tmux is run inside tmux, the inner tmux's outside terminal is tmux:

- `set-clipboard` and `Ms` must be configured for the inner tmux as for any other terminal. `TERM` will be `screen` or `screen-256color` or `tmux` or `tmux-256color`.

- The outer tmux must have `set-clipboard` set to `on` rather than `external` and it must be configured with `Ms` for its outside terminal, whatever that is.
- The outside terminal must have OSC 52 enabled.

Terminal support - xterm

xterm supports OSC 52 but it is disabled by default. It can be enabled by putting this in `.Xresources` or `.Xdefaults`:

```
XTerm*disallowedWindowOps: 20,21,SetXprop
```

Terminal support - VTE terminals

VTE terminals (GNOME terminal, XFCE terminal, Terminator) do not support the OSC 52 escape sequence.

Most will ignore it, but some versions will not and will instead print it to the terminal - this appears as a large set of letters and numbers covering any existing text. To fix this problem, turn `set-clipboard` off:

```
set -s set-clipboard off
```

Terminal support - Kitty

Kitty does support OSC 52, but it has a bug where it appends to the clipboard each time text is copied rather than replacing it.

This bug can be worked around by modifying the `kitty.conf` file to add `no-append`:

```
clipboard_control write-primary write-clipboard no-append
```

Terminal support - rxvt-unicode

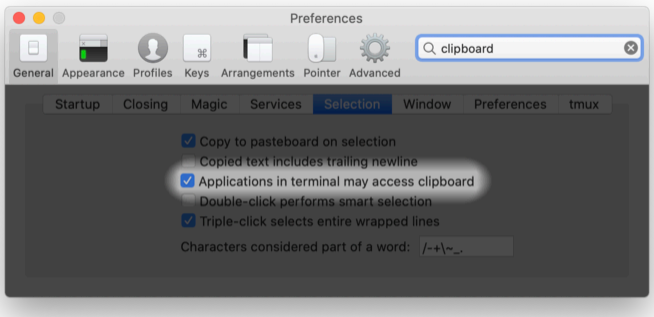
rxvt-unicode does not support OSC 52 natively. There is an unofficial Perl extension here.

Terminal support - st

st supports OSC 52 but versions before 0.8.3 have a length limit on the amount of text that can be copied, so text may be truncated.

Terminal support - iTerm2

iTerm2 supports OSC 52 but it has be enabled from the preferences with this option:



Quick summary

In summary, to configure `set-clipboard`, follow these steps:

1. Make sure `set-clipboard` is set in tmux:

```
$ tmux show -s set-clipboard
external
```

   If it is not `on` or `external`, add this to `.tmux.conf` and restart tmux (use `on` rather than `external` before tmux 2.6):

```
set -s set-clipboard external
```

2. Make sure `Ms` is set. Start tmux and run:

```
$ tmux info|grep Ms
180: Ms: [missing]
```

   If it is `[missing]`, get the value of `TERM` outside tmux:

```
$ echo $TERM
rxvt-unicode-256color
```

Then add an appropriate `terminal-features` or `terminal-overrides` line to `.tmux.conf` and restart tmux. For tmux 3.2 or later:

```
set -as terminal-features ',rxvt-unicode-256color:clipboard'
```

Or for older tmux versions:

```
set -as terminal-overrides ',rxvt-unicode-256color:Ms=\E]52;%p1%s;%p2%s\007'
```

Then start tmux and check it has worked by running this inside tmux:

```
$ tmux info|grep Ms:
180: Ms: (string) \033]52;%p1%s;%p2%s\a
```

3. Enable support in the terminal options if necessary, or use a terminal where it is enabled by default.

External tools

Available tools

An alternative to using `set-clipboard` is to use an external tool to set the clipboard. tmux has a method to pipe copied text to a command rather than only creating a paste buffer. The copy key bindings can be changed to do this.

The available tools are:

- On Linux and *BSD, there are the *xsel(1)* and *xclip(1)* tools, usually available as packages.
- macOS has a builtin tool called *pbcopy(1)*.

These tools talk to the *X(7)* server (or equivalent) directly so without additional configuration they only work on the local computer.

How to configure - tmux 3.2 and later

tmux 3.2 introduced an option called `copy-command` to set a command to pipe to for all key bindings. This is used when `copy-pipe` is called with no arguments which is now the default. If the option is empty, the copied text is not piped.

To pipe to *xsel(1)*:

```
set -s copy-command 'xsel -i'
```

The more complex configuration in the next section also works with tmux 3.2 and later versions.

How to configure - tmux 2.4 to 3.1

To use these tools with tmux before tmux 3.2, the copy key bindings must be changed. The equivalent command to the default `copy-selection-and-cancel` is `copy-pipe-and-cancel`; if using `copy-selection` instead use `copy-pipe`, or for `copy-selection-no-clear`, `copy-pipe-no-clear`.

The copy key bindings are:

- `C-w` and `M-w` with *emacs(1)* keys (`mode-keys` set to `emacs`).
- `C-j` and `Enter` with *vi(1)* keys (`mode-keys` set to `vi`).
- `MouseDragEnd1Pane` for copying with the mouse.

These must be changed for the key table in use. For *emacs(1)* keys:

```
bind -Tcopy-mode C-w               send -X copy-pipe-and-cancel 'xsel -i'
bind -Tcopy-mode M-w               send -X copy-pipe-and-cancel 'xsel -i'
bind -Tcopy-mode MouseDragEnd1Pane send -X copy-pipe-and-cancel 'xsel -i'
```

Or for *vi(1)* keys:

```
bind -Tcopy-mode-vi C-j               send -X copy-pipe-and-cancel 'xsel -i'
bind -Tcopy-mode-vi Enter             send -X copy-pipe-and-cancel 'xsel -i'
bind -Tcopy-mode-vi MouseDragEnd1Pane send -X copy-pipe-and-cancel 'xsel -i'
```

How to configure - tmux 2.3 and earlier

In tmux 2.4, copy mode key bindings were completely changed so that tmux commands could be bound in copy mode instead of a limited set of copy-mode-only commands. The configuration for older versions for *emacs(1)* keys looks like this:

```
bind -temacs-copy C-w               copy-pipe 'xsel -i'
bind -temacs-copy M-w               copy-pipe 'xsel -i'
bind -temacs-copy MouseDragEnd1Pane copy-pipe 'xsel -i'
```

Or for *vi(1)* keys:

```
bind -tvi-copy C-j               copy-pipe 'xsel -i'
bind -tvi-copy Enter             copy-pipe 'xsel -i'
bind -tvi-copy MouseDragEnd1Pane copy-pipe 'xsel -i'
```

`set-clipboard` and `copy-pipe`

If the `copy-pipe` method is used with a terminal that also supports `set-clipboard`, the two can conflict. It is best to disable `set-clipboard` in that case:

```
set -s set-clipboard off
```

Common issues - `DISPLAY`

Because the *xsel(1)* and *xclip(1)* tools need to talk to the *X(7)* server, they need the `DISPLAY` environment variable to be set. This is not normally a problem, but if it is missing (for example if tmux is started outside *X(7)*), it can be set with something like:

```
$ tmux setenv -g DISPLAY :0
```

Common issues - *xclip(1)*

xclip(1) has a bug where it does not correctly close `stdout` so tmux doesn't know it has finished and won't respond to any further key presses. The easiest fix is to redirect `stdout` to `/dev/null`:

```
xclip >/dev/null
```

Common issues - wrong clipboard

X(7) has several clipboards. If copied text isn't available, look at:

- The `-p`, `-s` and `-b` flags for *xsel(1)*
- The `-selection` flag for *xclip(1)*.

- Contributing
- Control Mode
- FAQ
- Formats
- Getting Started
- Installing
- Modifier Keys
- Recipes

Clone this wiki locally

```
https://github.com/tmux/tmux.wiki.git
```