

Unix Tricks

Brian Mays

July 2000

Topics

- Using Windows Keys
- Preprocessing \LaTeX
- Ssh Tunnelling for POP

Using Windows Keys

The Problem Most PCs today come with those stupid little windows keys. With a clever trick, you can use these keys in X.

Using Windows Keys

The Problem Most PCs today come with those stupid little windows keys. With a clever trick, you can use these keys in X.

The Trick Assign useful *keysyms* to the appropriate *keycodes*. The keycodes in question are

0x73 the “Windows” key

0x75 the “Menu” key

Using Windows Keys

The Problem Most PCs today come with those stupid little windows keys. With a clever trick, you can use these keys in X.

The Trick Assign useful *keysyms* to the appropriate *keycodes*. The keycodes in question are

0x73 the “Windows” key

0x75 the “Menu” key

I map these keys to the following *keysyms*:

the “Windows” key → F13

the “Menu” key → Menu

To take advantage of this, the *xmodmap* utility must be used.

- Create a file with the following *xmodmap* expressions:

```
keycode 0x73 = F13
```

```
keycode 0x75 = Menu
```

To take advantage of this, the *xmodmap* utility must be used.

- Create a file with the following *xmodmap* expressions:

```
keycode 0x73 = F13
```

```
keycode 0x75 = Menu
```

- Modify your *.xsession* or *.xinitrc* file to run *xmodmap* on this file. For example,

```
case "$DISPLAY" in
```

```
:*) xmodmap $HOME/.Xmodmap-linux ;;
```

```
esac
```

To take advantage of this, the *xmodmap* utility must be used.

- Create a file with the following *xmodmap* expressions:

```
keycode 0x73 = F13
keycode 0x75 = Menu
```

- Modify your *.xsession* or *.xinitrc* file to run *xmodmap* on this file. For example,

```
case "$DISPLAY" in
:* ) xmodmap $HOME/.Xmodmap-linux ;;
esac
```

- Configure your window manager to use these new keys. For *Windowmaker*, a useful configuration is

```
RootMenuKey      = Menu;
WindowListKey    = F13;
```

Summary

- By using *xmodmap*, one can remap any key on one's keyboard to any useful purpose.
- Examples include window manager functions, the compose key, and special editor commands.
- X11 provides *keysyms* for special purpose keys, foreign language support, and 35 function keys.
- *Xkeycaps* can be used as a convenient way to generate *xmodmap* entries.
- Remember that X is portable and network independent. Try to keep keyboard remapping appropriate to the X display.

Preprocessing L^AT_EX

The Problem Some input formats are annoying, with structures that require repetition of input. For example, L^AT_EX environments present problems:

The `{\LaTeX}` book says that,

```
\begin{quote}
```

```
    The standard {\LaTeX} document styles provide  
    environments for producing several different  
    kinds of displays.
```

```
\end{quote}
```

- *Begin/end* are verbose.
- Does not lend itself to “brace matching.”
- Changing the environment type means changing *both* ends.

Compare with *troff's me* macros:

The LaTeX book says that,

```
.(q
```

```
    The standard LaTeX document styles provide  
    environments for producing several different  
    kinds of displays.
```

```
.)q
```

Note that

- More terse and well defined.
- The parentheses can be matched.

The Trick *Preprocess* the input file before passing it to \LaTeX . This can be done using existing tools or specially written programs.

The Trick *Preprocess* the input file before passing it to \LaTeX . This can be done using existing tools or specially written programs.

Aside My \LaTeX documents usually include *pic* drawings.

.PS

pic source

.PE

Tpic is already preprocessing the document. *Pic* definitions and macros are lost across multiple input files, however.

The Trick *Preprocess* the input file before passing it to \LaTeX . This can be done using existing tools or specially written programs.

Aside My \LaTeX documents usually include *pic* drawings.

```
.PS  
  pic source  
.PE
```

Tpic is already preprocessing the document. *Pic* definitions and macros are lost across multiple input files, however.

The *soelim* utility can be used to combine (source) multiple input files in the preprocessing stage.

```
.so part1.tex  
.so part2.tex
```

Problem: *Soelim* leaves lines beginning with “.lf”, however.

The Preprocessor

Design Implement a small preprocessor in *flex* to help.

- Keep a format similar to *groff/pic/grap*.
 - Preprocessor commands occupy a line and are replaced by a line.
 - Preprocessor commands begin with a period (“.”) followed by two characters.
 - *Troff ms* and *me* macro package conventions are followed.
 - Arguments are separated by whitespace and may be quoted.
- Replace *begin/end* with preprocessor commands.
- Track environments on a stack to make nesting of environments convenient.
- Support the “.lf” (line and file) commands.

Examples Thus, our example becomes

.(e quote

The standard `{\LaTeX}` document styles provide environments for producing several different kinds of displays.

.)e

Examples Thus, our example becomes

```
.(e quote
```

```
  The standard {\LaTeX} document styles provide  
  environments for producing several different  
  kinds of displays.
```

```
.)e
```

Multiple environments may be specified with one command:

```
.(e center bfseries large
```

```
  This is a large, bold, centered line.
```

```
.)e
```

Examples Thus, our example becomes

```
.(e quote
```

```
  The standard {\LaTeX} document styles provide  
  environments for producing several different  
  kinds of displays.
```

```
.)e
```

Multiple environments may be specified with one command:

```
.(e center bfseries large
```

```
  This is a large, bold, centered line.
```

```
.)e
```

becomes

```
\begin{center}\begin{bfseries}\begin{large}
```

```
  This is a large, bold, centered line.
```

```
\end{large}\end{bfseries}\end{center}
```

Environments that take arguments are also supported. For example,

```
.(E tabular {111} t
```

becomes

```
\begin{tabular}[t]{111}
```

(“.)e” is still used to close the environment.)

Environments that take arguments are also supported. For example,

```
.(E tabular {111} t
```

becomes

```
\begin{tabular}[t]{111}
```

(“.)e” is still used to close the environment.)

The line and file commands are transformed into T_EX messages:

```
.lf 1 part1.tex
```

becomes

```
\message{ } (part1.tex}
```

Notes

- The “.ES”/“.EE” commands are equivalent to “. (e”/“.) e”.
- The stack is emptied at the end of processing; so *end* environment commands may be omitted from the end of the file (including “\end{document}”).
- Arguments to “.) e”/“.EE” are passed verbatim directly after “\end{environment}”.

Notes

- The “.ES”/“.EE” commands are equivalent to “. (e”/“.) e”.
- The stack is emptied at the end of processing; so *end* environment commands may be omitted from the end of the file (including “\end{document}”).
- Arguments to “.) e”/“.EE” are passed verbatim directly after “\end{environment}”.
- The “. (v”/“.) v” commands turn off preprocessing.
- The “.CS”/“.CE” commands also put L^AT_EX in *verbose* mode.

Notes

- The “.ES”/“.EE” commands are equivalent to “. (e”/“ .) e”.
- The stack is emptied at the end of processing; so *end* environment commands may be omitted from the end of the file (including “\end{document}”).
- Arguments to “.) e”/“.EE” are passed verbatim directly after “\end{environment}”.
- The “. (v”/“ .) v” commands turn off preprocessing.
- The “.CS”/“.CE” commands also put L^AT_EX in *verbose* mode.
- Other features are available including preprocessing commands for handling sections and paragraphs.
- Unrecognized commands are passed untouched; useful for other preprocessors (*pic*, *grap*, etc.).

Using the Preprocessor

The *make* utility can greatly simplify using the preprocessor. A few useful rules are

```
.SUFFIXES: .dvi .tex .t
```

```
%.tex: %.t
```

```
soelim "$<" | latexpp | grap | pic -t > $*.tex
```

```
%.dvi: %.tex
```

```
latex "$<"
```

```
latex "$<"
```

- Input files for *latexpp*, the preprocessor, are denoted by a “.t” extension.
- Explicit input files for \LaTeX are generated and given the “.tex” extension.
- More complicated rules, useful for incorporating \BibTeX , are possible.

Summary

- Using *flex*, small, useful preprocessors can be quickly, and easily written. (*Latexpp* is less than 400 lines.)
- Undesirable input formats may be replaced with more convenient ones.
- The preprocessor can be used to automatically track information, such as nesting levels, and environments.
- T_EX's free-form input lends itself well to preprocessing.
- A line-based preprocessor format simplifies implementation and clearly separates preprocessor input from non-preprocessor input.
- A clever preprocessor format can make use of existing tools.

Ssh Tunnelling for POP

The Problem

- POP is a common mail-retrieval protocol.
- Normal POP sessions are insecure, since the password is sent as cleartext.
- Secure methods, such as APOP, are not widely supported.

Ssh Tunnelling for POP

The Problem

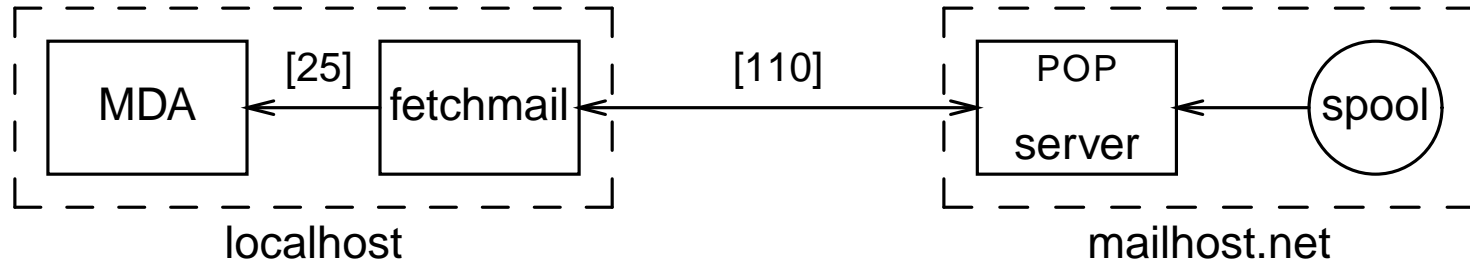
- POP is a common mail-retrieval protocol.
- Normal POP sessions are insecure, since the password is sent as cleartext.
- Secure methods, such as APOP, are not widely supported.

The Trick

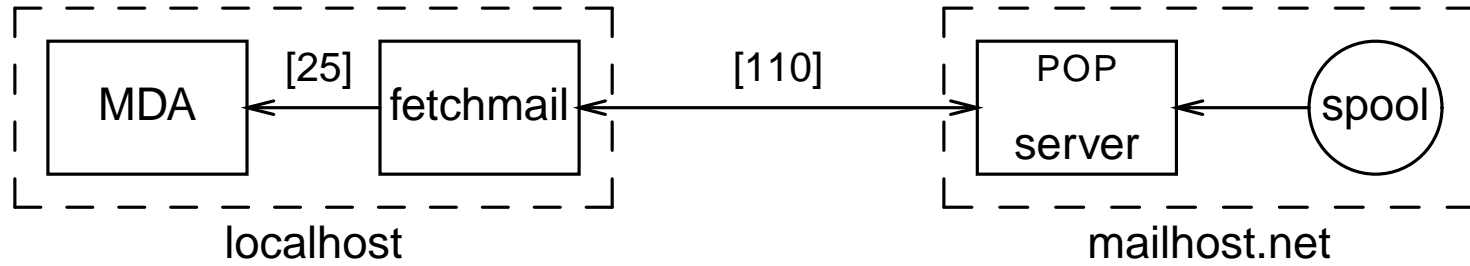
- Use port forwarding of *ssh* to encrypt the POP session.
- I use *fetchmail* to automate this process.

Aside This is overkill, since only the *password* needs to be encrypted, not the *content*.

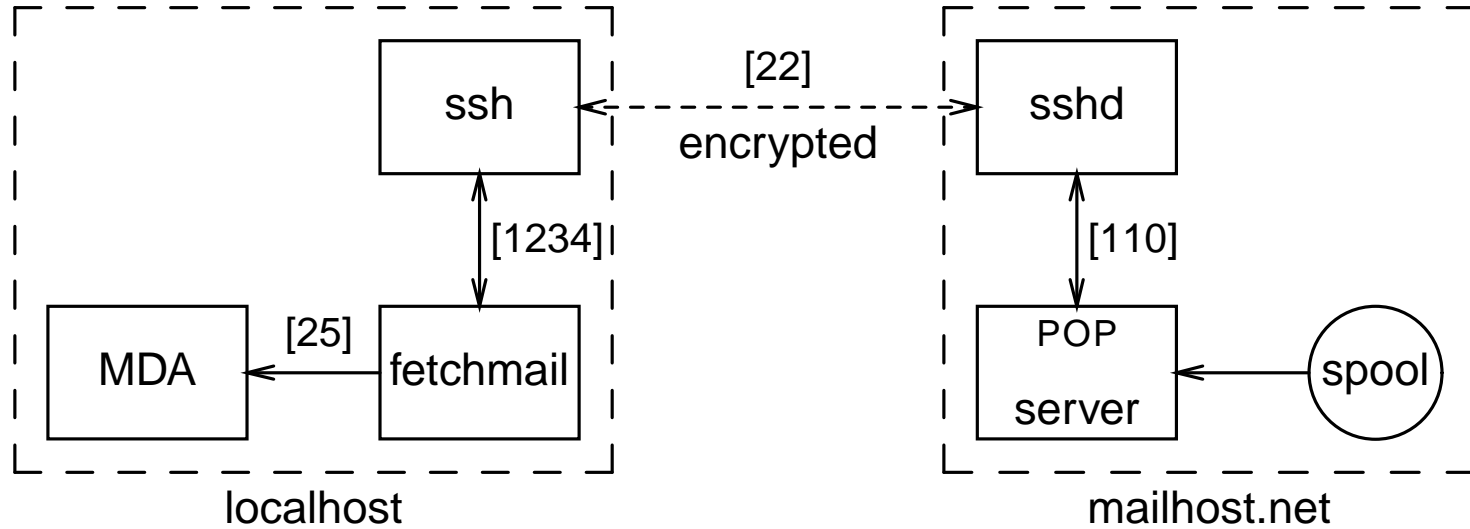
Fetchmail



Fetchmail



Ssh Tunnelling



Notes

- Unless a protocol is specified, *fetchmail* will automatically query the server to pick one.
- *Fetchmail* can run an MDA (e.g., *sendmail* or *procmail*) directly instead of forwarding to port 25.
- *Ssh* needs to run *something*. Usually, this is *sleep*.
- The local forwarded port number (here “1234”) is arbitrary.
- Over slow connections, such as modem lines, *ssh*’s compression may be used.

Executing Ssh

To start, run the following:

```
ssh -C -f -L 1234:mailhost.net:110 \  
mailhost.net sleep 20
```

Executing Ssh

To start, run the following:

```
ssh -C -f -L 1234:mailhost.net:110 \  
mailhost.net sleep 20
```

-C – Use compression (optional).

-f – Fork to background after establishing the forwarding connection.

mailhost.net – The POP server.

-L 1234:mailhost.net:110 –

Forward local port 1234 to port 110 on mailhost.net.

sleep 20 – The remote command. Sleep long enough to setup a connection.

Using Fetchmail

The run control file is *.fetchmailrc*.

- Uses a free-format, token-oriented syntax.
- Keywords may be abbreviated.
- Whitespace is ignored unless surrounded by quotes (e.g., "*string with spaces*").
- Lots of extra ignored tokens (noise keywords).
- Punctuation (:, ;, and ,) is ignored.
- Comments begin with #.
- Basic format is

```
poll servername protocol protocol username name  
password password
```

Running Ssh from Fetchmail

An example *.fetchmailrc* entry is

```
poll mailhost.net via localhost port 1234  
    with proto pop3:  
preconnect "ssh -f -L 1234:mailhost.net:110  
    mailhost.net sleep 20 </dev/null  
    >/dev/null";
```

Running Ssh from Fetchmail

An example *.fetchmailrc* entry is

```
poll mailhost.net via localhost port 1234  
    with proto pop3:  
preconnect "ssh -f -L 1234:mailhost.net:110  
    mailhost.net sleep 20 </dev/null  
    >/dev/null";
```

Notes

- IMAP may be used instead of POP, with *pop3* → *imap* and *110* → *143*.
- Forwarding for *mailhost.net* can be specified in the *ssh* configuration file.
- *Ssh* will prompt for a password (if needed). This step can be eliminated by using *ssh-agent* and *ssh-add*.

Summary

- Security is important on the internet.
- The *email* does not need to be encrypted, the *password* used to access email does.
- *Ssh* can provide security to many applications, not just remote shell sessions.
- Communication through any port can be passed through a secure channel, including X11, POP, and IMAP.
- *Fetchmail* provides hooks that can be used to automate *ssh* forwarding.

THE END