The moɔpTi*k*Z package

a library for typesetting computer networks

Stephan M. Günther

April 6, 2016

The moɔpTi*k*Z package provides shapes for typesetting computer networks, e. g. shapes for hubs, switches, routers, and more. It is in use since 2012 for the lecture materials of our basic course in computer networks due to shortcomings of some commercial products and (at that time) the lack of meaningful alternatives. For the first official release many shapes were redesigned and a couple of new features added, e. g. complete anchor support and custom colors. Currently the package provides the following shapes and commands:

- `hub`, `switch`, `router`, `client`, `server`, `nuc`, `messageclosed`, `messageopen`
- `\tikzextractx`, `\tikzextracty`
- `\moeptikzset`

1 What it can do

Did you ever try to build slides for a networking lecture from scratch? If so, do you care about off-topic details such as using solely vector graphics, perfect alignment of nodes, easy modification of hundreds of figures when you decide to change fonts or colors, and doing all that without even thinking about a pointing device? Then you probably like what moɔpTi*k*Z can do for you:

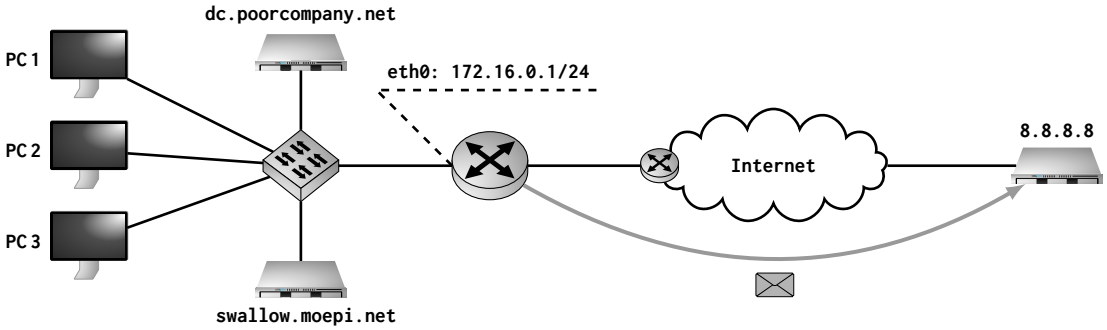

Figure 1: What moɔpTi*k*Z can (or cannot?) do for you

2 What it cannot do

You probably noticed the misalignment of `PC2` (and its friend) in Figure 1. Well, you might not have noticed it in the first place (I didn't either), but now, as you know about that, it must drive you crazy. The good news: it's your fault (in this particular case it is my fault, of course). moɔpTi*k*Z cannot calculate coordinates for you. If you are interested in writing Ti*k*Z instead of pointing around you will notice the difference:

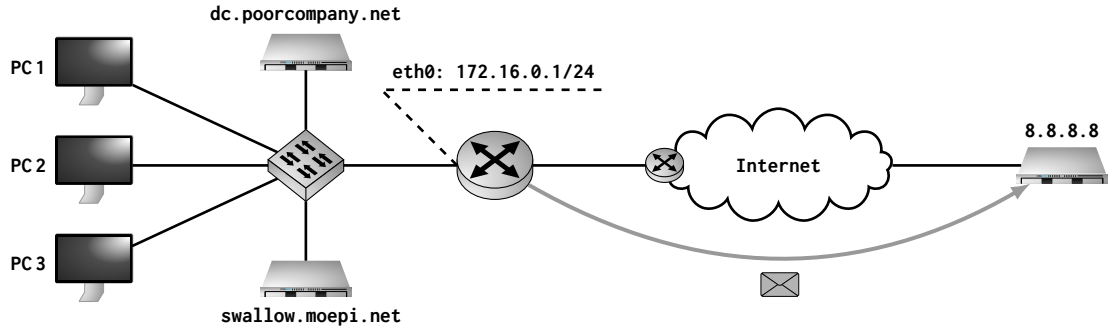

Figure 2: What moɔpTi*k*Z can do for you if you can do your math

3 Package options

The only package option currently defined is `shading=<true|false>`. If true (default), nodes are shaded to create the visual effect of a spotlight. The shading has only a marginal impact on the resulting document size but may be undesirable depending on the personal preference.

4 The shapes

Figure 3 presents the different shapes as printed by `\node[<shape>] at ...` (the label font is typewriter here, but that's up to you). The default label shows the name of the corresponding shape. The gray box around the shape is a unit square making the shapes bounding box visible. The for red crosses mark the label positions specified by a node option such as `label=<pos>:<labeltext>`. The gray bullets demonstrate automatically calculated anchor positions.

If the node option `minimum size` is omitted, nodes will be drawn in a bounding box of 1 cm × 1 cm. Since not all shapes fill their bounding box, default label positions may vary between nodes, e. g. label positions of server and client shapes differ. This may be undesirable proved to be better than having node labels too far away from the visible shape.

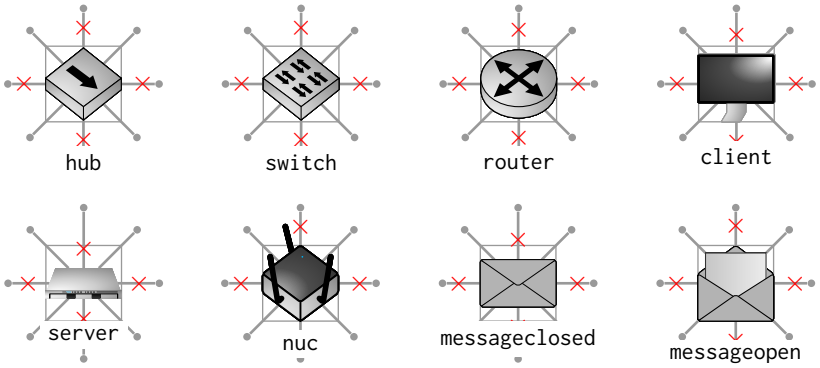

Figure 3: Shapes currently defined

4.1 Changing node colors

There are very few occasions where you might want to highlight a specific node by changing its color. Or you are upset by the default mark color (black), i.e., the color of arrows on top of some shapes. You may change it at will: all shapes support custom colors using the `fill=<color>` option. In addition to that, the message shapes also allow to change the line color using the `draw=<color>` option.

```
\node[router,fill=blue,draw=orange] at (0,0) {};
```

```
\node[messageclosed,fill=red!40,draw=red] at (0,0) {};
```

```
\node[messageopen,fill=orange!40,paper color=blue,draw=orange] at (0,0) {};
```

The effect of the `draw` option depends on the actual shape: a `router` will change the color of its marking (the arrows) while an `openmessage` or `closedmessage` change their line color. This behavior may seem inconsistent but is intended – give it a try, or come up with fix.

4.2 Upcoming features

There will be some additional shapes such as `firewall` and `workstation` soon. Feature requests are welcome. And of course there will be many code updates, since the current pfg code is all but nice.

5 The commands

moɔpTi*k*Z provides two commands to make your live with easier:

1. `\tikzextractx{(x,y)}{\<macroname>}`
   Extracts the x-component of a coordinate and stores it `\<macroname>` for later use.
2. `\tikzextracty{(x,y)}{\<macroname>}` Extracts the y-component of a coordinate and stores it `\<macroname>` for later use.
3. `\moeptikzset{<key>=<value>}` Sets the internal key `<key>` to value `<value>`. Currently, the only defined key is `shading=<true|false>`.

6 Known bugs and missing features

- `[bug]` Sometimes lines from a node to another node do not end at an automatically calculated anchor but at the node's center, causing the line to cross parts of a shape. Please report such cases (including your code) to `moepi@moepi.net`.
- `[missing]` `openmessage` has a shiny sheet of shaded blue paper. At the moment there is no way to change its color (or turn off the shading) except for hacking my hacky pgf code.