Faire des schémas avec un langage et pas avec la souris

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 9 janvier 2009

https://www.bortzmeyer.org/schemas-avec-un-langage.html

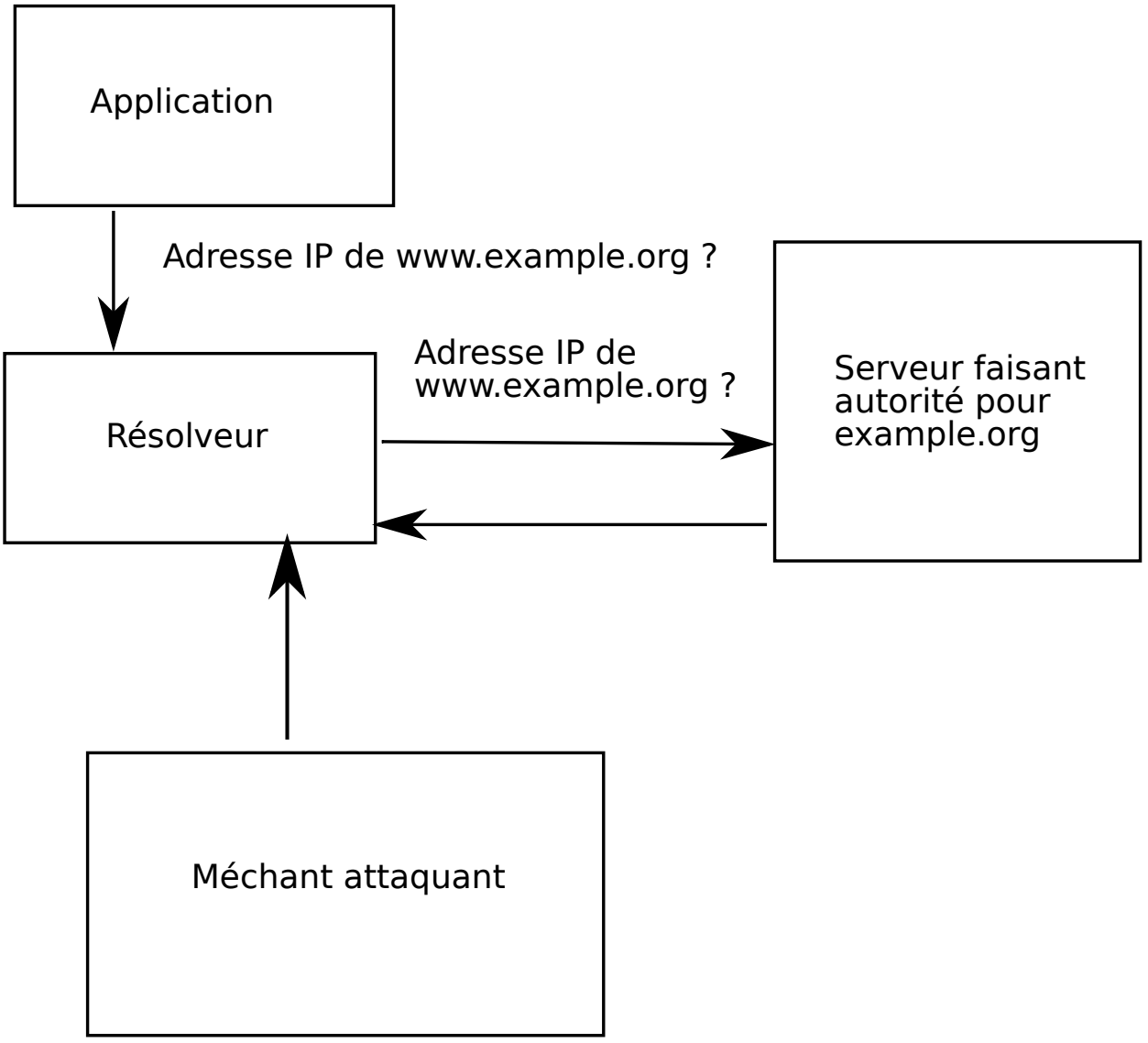
On dit qu'une image vaut mille mots. C'est vrai qu'il est souvent bien plus facile d'expliquer avec un dessin qu'avec de la prose. Dans le monde physique, je le fais souvent. Alors, lorsqu'on communique via des ordinateurs ? Pourquoi n'y a t-il pas plus de dessins sur ce blog ? Une des raisons est que je dessine très mal à la souris, je préférerai utiliser un langage de description de schémas.

Prenons un exemple, pour un exposé que je dois faire sur les vulnérabilités du DNS, par exemple la faille Kaminksy <https://www.bortzmeyer.org/comment-fonctionne-la-faille-kaminsky.html>. Je voudrais représenter l'application, le résolveur DNS, le serveur faisant autorité et le méchant pirate qui essaie d'empoisonner le résolveur ainsi que les paquets qu'ils s'échangent.

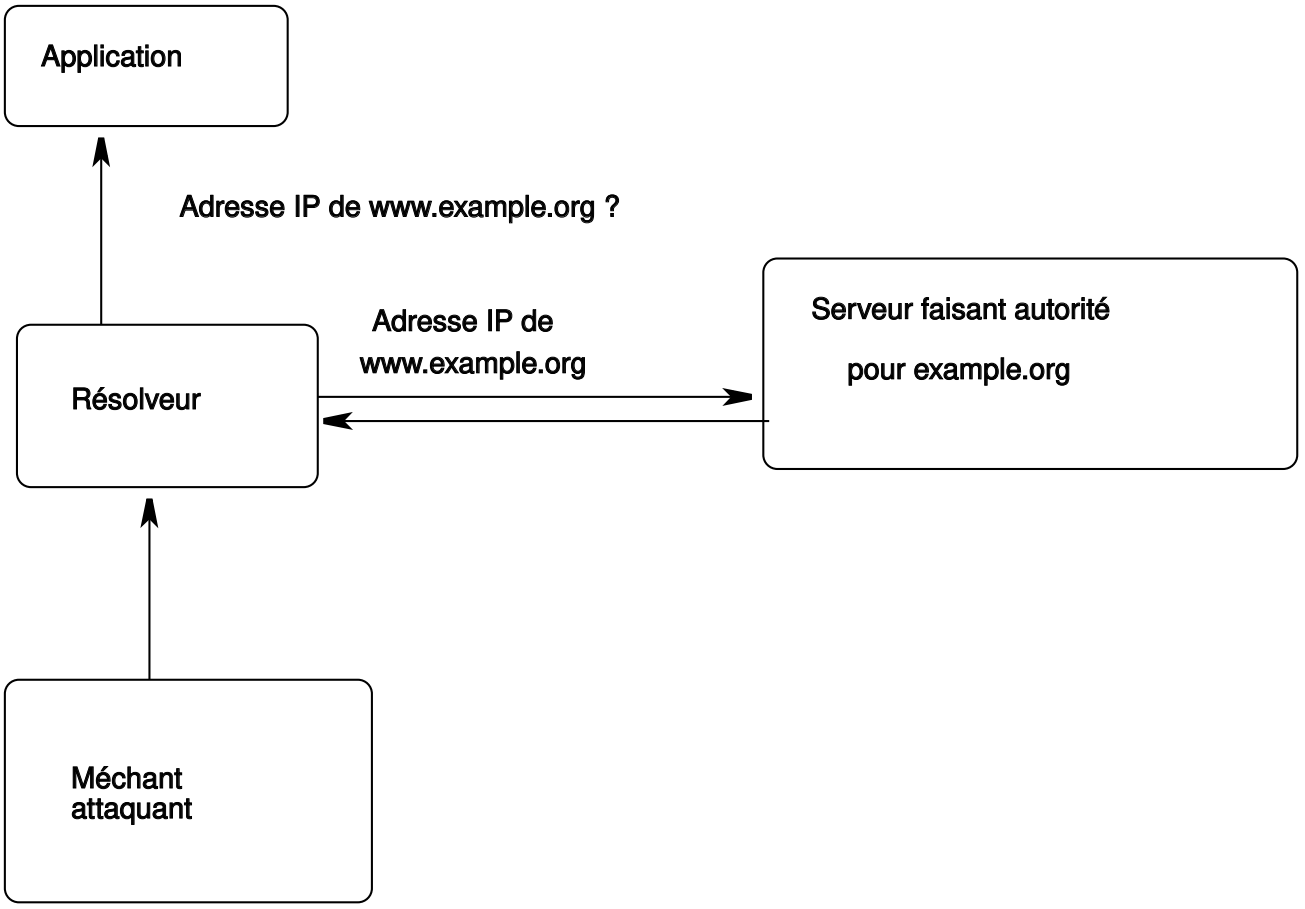
La solution qui semble venir immédiatement à l'esprit de beaucoup de gens est de faire un dessin « à main levée » avec un logiciel de dessin vectoriel. Prenons Inkscape par exemple, cela donne . Je trouve le processus de dessin lent et pénible, il faut que je place tous les objets à la main, c'est impossible à automatiser (par exemple pour des rechercher/remplacer sur plusieurs images, ou bien pour faire produire une partie du dessin par un programme) et pas très réussi au final. Mais le format d'enregistrement est ouvert (SVG) et modifiable par un programme. Par contre, avec son interface ultra-chargée, Inkscape convient peut-être au dessinateur industriel professionnel mais certainement pas au pauvre technicien qui veut juste faire un schéma avec quatre boîtes et six flèches.

Bon, si on n'aime pas Inkscape, le vénérable xfig marche encore très bien. Bien plus rapide et léger, cela donne . Il y a quelques trucs agaçants (il semble qu'on ne puisse pas avoir des blocs multi-lignes de texte) mais ce n'est pas trop grave pour des schémas simples : je ne refais pas les plans de l'Airbus A380.

Et, encore plus simple, il y a le traditionnel art ASCII. Mon éditeur favori, emacs, a un mode pour faciliter cet art, `picture-mode`.




```
+--------------+
| Application  |
|              |
+--------------+
       |
       | Adresse IP de www.example.org ?
       v                        +------------------------+
+--------------+   Addr. IP ?|                        |
| Résolveur    |------------>|    Serveur             |
|              |             |    faisant autorité    |
+--------------+<------------+------------------------+
       ^           2001:DB8:42::deca:fbad
       |
       | 2001:db8:666:1
+--------------+
| Méchant      |
|              |
+--------------+
```



Grâce à ce mode, c'est bien plus rapide que Inkscape ou xfig même si le résultat a un côté... un peu dinosaurien. Quelques petits trucs sur `picture-mode` : attention aux tabulations, il vaut mieux appeler `untabify`. Voir aussi le très utile `blank-mode`. Comme un des équivalents clavier de `picture-mode` (celui pour changer le sens de déplacement du curseur, pour aller de haut en bas et plus de gauche à droite) ne me plait pas, je l'ai redéfini :

```
; The default is "\C-c." which I don't like
(setq picture-mode-hook '(lambda ()
                       (local-set-key "\C-cv" 'picture-movement-down)))
```

Bon, mais toutes ces méthodes ont le même défaut : au lieu de décrire le schéma dans un langage, on doit bouger ses doigts pour tout placer au millimètre, ce que je trouve désagréable. Ce que je préférerai, ce serait un langage où j'indique mes éléments et où un programme les place plus ou moins joliment.

Il existe de tels langages. Par exemple, avec graphviz, je peux écrire, dans le langage DOT :

```
igraph resolution_dns {

// Machines
node [shape=box, fontsize=14];
   Application;
   Résolveur;
   Serveur_faisant_autorité;
```

```
   Méchant;

 edge [fontsize=10];
   Application -> Résolveur [label = "Adresse IP de www.example.org ?"];
   Résolveur -> Serveur_faisant_autorité [label = "Adresse IP de www.example.org ?"];
   Serveur_faisant_autorité -> Résolveur [label = "2001:DB8:42::deca:fbad"];
   Méchant -> Résolveur [label="2001:db8:666:1"];
}
```

puis de le convertir dans un format graphique (ici, PNG) :

```
% dot -Gcharset=latin1 -Tpng resolution-dns-graphviz.dot > \
            resolution-dns-graphviz.png
```

et hop, j'ai un graphique dessiné automatiquement : . (Une bonne introduction à DOT est l'article "*Drawing graphs with dot*" <http://www.graphviz.org/pdf/dotguide.pdf>.)

graphviz dispose de nombreux moyens d'influencer le résultat, si le logiciel n'a pas produit le schéma parfait (par exemple, il est composé de plusieurs programmes, comme `dot` ci-dessus, utilisant des algorithmes de placement différent ; si `dot` ne donne pas un bon résultat, on peut tenter sa chance avec `neato`). Mais il a ses limites, par exemple le placement des étiquettes près des flèches est franchement peu réussi. C'est particulièrement vrai pour les étiquettes qu'on voudrait voir attachées aux extrémités de la flèche (ce qui n'est pas le cas dans l'exemple ci-dessus, où j'ai utilisé `label` et pas `headlabel` ou `taillabel`). Le problème est connu <http://www.graphviz.org/mywiki/FaqEdgeLabelPlace> mais n'a pas vraiment de solution.

Ne critiquons pas trop : le placement automatique d'un graphe sur un plan est un problème très ancien de l'algorithmique et est loin d'être complètement résolu. On ne peut donc pas encore tout sous-traiter à un programme.

Néanmoins, Graphviz va dans la bonne direction. On décrit les éléments, on ne dessine pas et on laisse le programme dessiner. C'est ma solution favorite, même si je dois parfois repasser à un logiciel WYSIWYG lorsque je n'arrive pas à convaincre Graphviz de produire un beau résultat.

Voici le source Graphviz complet (en ligne sur `https://www.bortzmeyer.org/files/resolution-dns-gr` du schéma montré plus haut. Autres exemples, un schéma entité/relation d'un registre de noms de domaine (voir le source (en ligne sur `https://www.bortzmeyer.org/files/entity-relation-registry.dot`)) : ou bien le protocole TURN du RFC 8656 [1] (voir le source (en ligne sur `https://www.bortzmeyer.org/files/turn2.dot`) : .

À noter que, avant le langage DOT de Graphviz, une solution présentant certaines ressemblances existait, pic (une documentation très complète est incluse dans groff ou bien peut être trouvée en ligne <http://floppsie.comp.glam.ac.uk/Glamorgan/gaius/web/pic.html> ; voir aussi <http://www.troff.org/papers.html>). Avec pic, je peux écrire :

1. Pour voir le RFC de numéro NNN, `https://www.ietf.org/rfc/rfcNNN.txt`, par exemple `https://www.ietf.org/rfc/rfc8656.txt`

```
.PS
box "Début";
arrow;
box "Milieu";
arrow;
box "Fin";
.PE
```

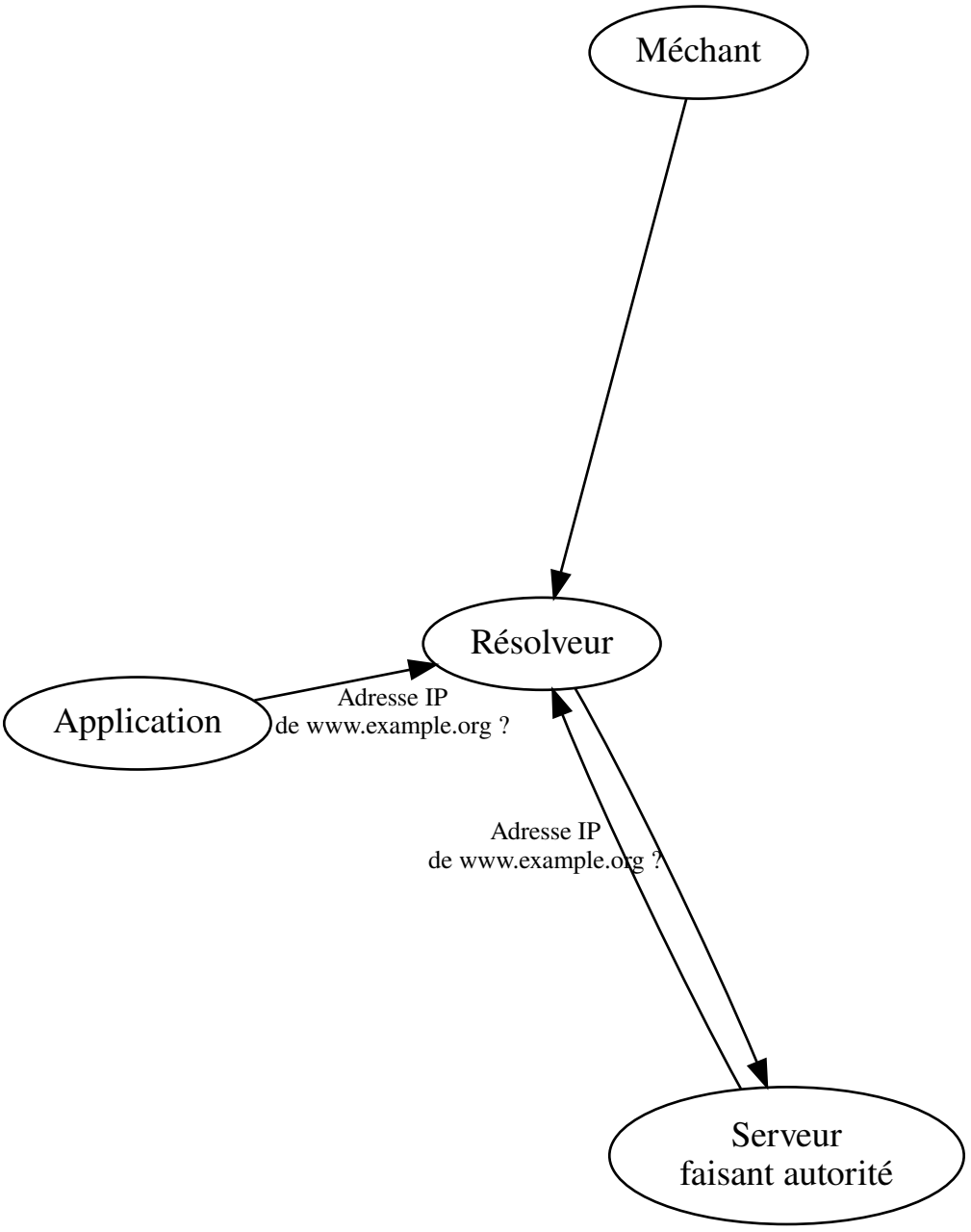
et la commande suivante :

```
% groff -p -Tps test.pic > test.ps
```

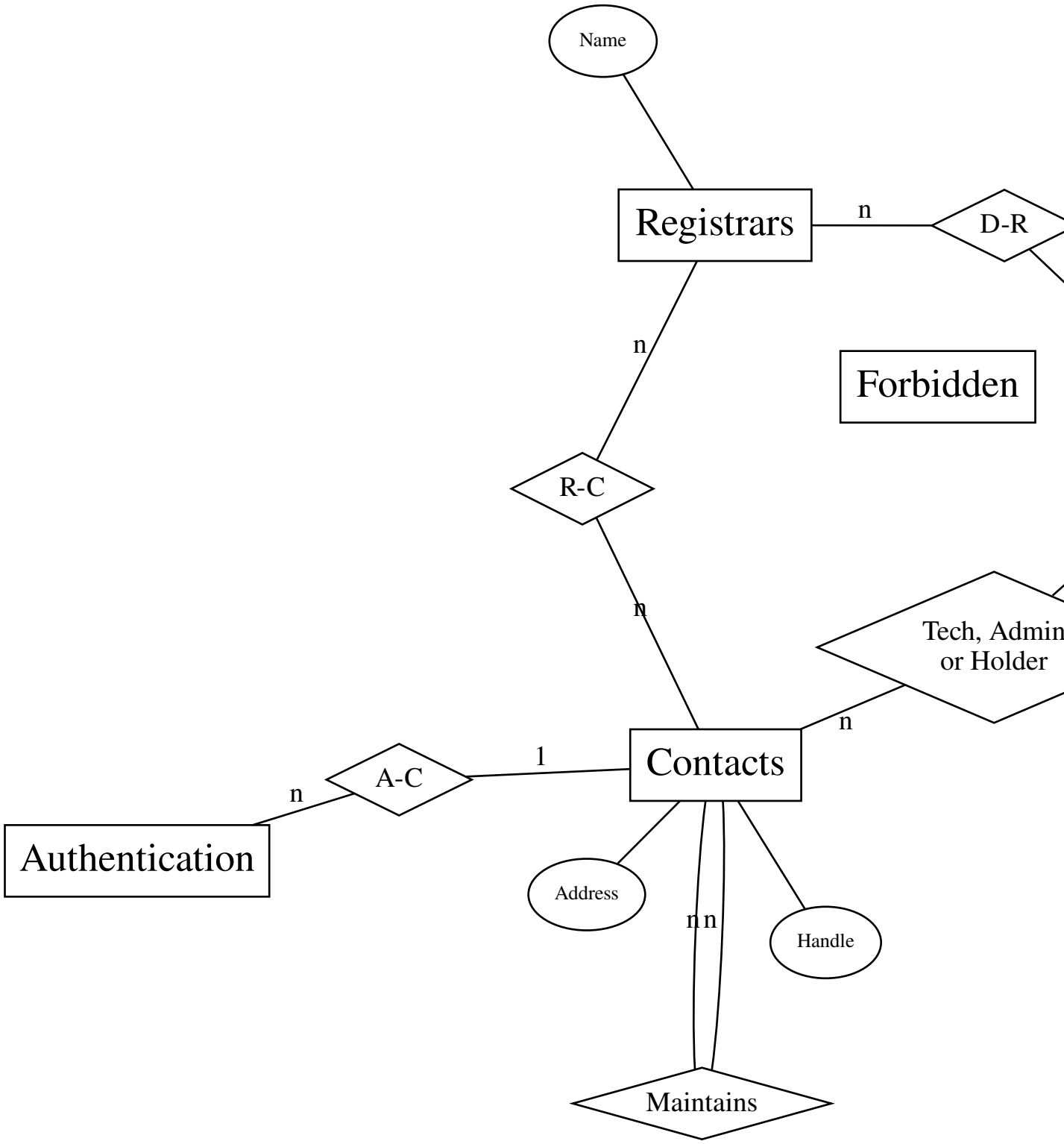
produira un beau fichier Postscript avec le dessin.

pic, plus ancien, est de bien plus bas niveau que Graphviz. Notamment, il nécessite souvent de donner des instructions de placement très détaillées, ce qui annule une bonne partie de l'effet du langage. Voici l'exemple complet en pic (en ligne sur `https://www.bortzmeyer.org/files/resolution-dns-pic.pic`) et son résultat : .

À noter que, sur son blog, David Monniaux avait posé un problème similaire, LaTeX, XFig, InkScape `<http://david.monniaux.free.fr/dotclear/index.php/2008/07/23/232-latex-xfig-inkscape>`. Damien Wyart me fait remarquer que les meilleurs sont Metapost `<http://www.tug.org/metapost.html>` et Asymptote `<http://asymptote.sourceforge.net>` (qui ne fait pas le placement lui-même, il faut tout lui indiquer). Même chose pour PGF et TikZ `<http://www.fauskes.net/nb/pgftikzexamples/>` qui sont spécifiques à LaTeX (tous les autres outils produisent des formats utilisables partout).

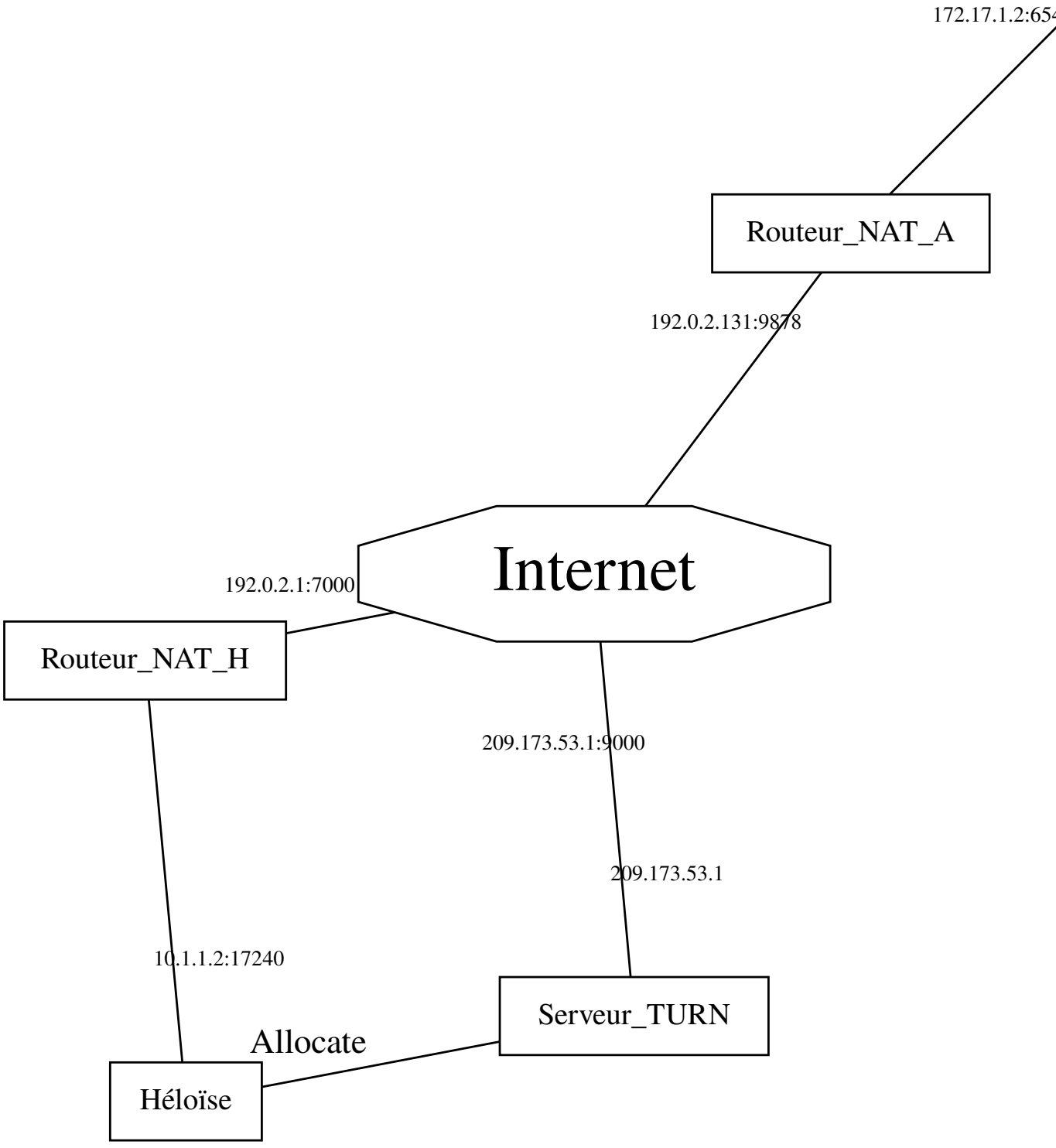


Résolution DNS,
en présence d'un attaquant



Entity-Relationship mode

https://www.bortzmeyer.org/schemas-avec-un-langage.html



Fonctionnement de TURN

https://www.bortzmeyer.org/schemas-avec-un-langage.html