FOSDEM 2025

Stéphane Bortzmeyer
<stephane+blog@bortzmeyer.org>

Première rédaction de cet article le 5 février 2025

https://www.bortzmeyer.org/fosdem-2025.html

Le FOSDEM est sans doute le plus grand rassemblement de développeurs (et quelques développeuses) de logiciel libre en Europe. L'édition 2025, qui marquait le vingt-cinquième anniversaire de la création du FOSDEM, vient de se tenir à Bruxelles, les 1 et 2 février.

Impossible de tout raconter. Le FOSDEM, ce sont des milliers de participants (et 950 kg de frites servies) et presque un millier d'orateurices. (Il y avait 2 064 propositions, 955 ont été acceptées.) Personne ne peut tout suivre, sans compter la difficulté à rentrer dans les salles les plus populaires. Malgré la générosité de l'ULB, qui prête ses locaux, il n'y a jamais assez de salles, ni assez de place dans les salles. Je vais donc me contenter de quelques points.

Saluons d'abord l'exploit de l'équipe technique : l'accès à l'Internet via le WiFi a parfaitement marché, de partout, tout le week-end. On est en 2025, donc il était évidemment en IPv6 seul. 8 846 machines ont été vues.

Vous pouvez regarder tout le programme <https://fosdem.org/2025/schedule/> et les vidéos <https://video.fosdem.org/2025/>.

J'ai suivi la salle <https://fosdem.org/2025/schedule/track/modern-email/> sur le courrier électronique. Vieille technologie que cet ancien réseau social (bien plus social que ce que vendent les entreprises privées aujourd'hui) ? Mais le courrier reste la seule solution de communication ouverte, reposant sur des protocoles normalisés et du logiciel libre, et largement accessible. Il y avait un excellent exposé de Robin Jarry sur le logiciel client ¿aerc <https://git.sr.ht/~rjarry/aerc> (un concurrent de mutt et autres logiciels tournant dans un terminal). Il gère même JMAP (RFC 8621 [1]). Et, via w3m, peut afficher les messages marketing en HTML. Mais il n'a pas encore de langage de filtrage. L'auteur n'envisage pas d'utiliser le langage normalisé Sieve (RFC 5228).

1. Pour voir le RFC de numéro NNN, https://www.ietf.org/rfc/rfcNNN.txt, par exemple https://www.ietf.org/rfc/rfc8621.txt

Autre client de courrier, très différent, Parula <https://parula.beonex.com/>. Lui est graphique, et se veut un concurrent crédible des grosses usines à gaz tout-en-un utilisées dans le monde "*corporate*". Si le grand public utilise Gmail, les entreprises en interne sont plutôt Microsoft. Ce souci d'être "*enterprise-ready*" pousse les développeurs de Parula à déprioritiser le chiffrement « très peu de gens l'utilisent ». (Le FOSDEM a toujours été partagé entre ceux qui veulent adapter le logiciel libre au monde "*corporate*", pour qu'il y gagne succès et respectabilité, et ceux qui préfèrent rester différents.)

Et côté serveurs et pas clients ? Il y a eu deux présentations de serveurs tout-en-un. Installer un serveur de messagerie aujourd'hui nécessite l'installation et la configuration de nombreux logiciels différents, par exemple Postfix, Dovecot, SpamAssassin et les logiciels qui font faire du SPF, du DKIM et du DMARC, sans parler du "*greylisting*" (RFC 6647). Sur mon serveur, qui utilise Postfix, les trois logiciels SPF, DKIM et DMARC utilisent trois façons différentes de s'interfacer à Postfix. . .Bref, il est tentant de remplacer cette ménagerie par un logiciel qui fait tout et le café. C'est ce que proposent Stalwart <https://stalw.art/> (écrit en Rust) et mox <https://www.xmox.nl/> (écrit en Go, l'autre langage qui concurrence Rust pour les serveurs Internet), présentés ici. Stalwart inclut même l'analyse des rapports DMARC. . .

Je n'ai pas aimé le côté trop marketing de la présentation de Stalwart (cf. ma remarque plus haut sur le partage du FOSDEM entre "*corporate*"-compatibles et libristes pur·es et dur·es). L'orateur abusait de "*intuitive*", "*enterprise-ready*" et autres "*red flags*" du discours marketing. En outre, il semble que bien des fonctions de base soient uniquement dans la version payante.

Personnellement, je ne compte pas utiliser ces logiciels car j'ai déjà une configuration qui fonctionne mais, si je devais monter un serveur de messagerie en partant de zéro, cela vaudrait sans doute un essai.

Le présentateur de mox, Mechiel Lukkien, insistait sur tous les problèmes qui sont de toute façon en dehors du contrôle du serveur de messagerie et rendent donc le « tout-en-un complet » illusoire. Ainsi, notait-il, il faut quinze enregistrements DNS pour un domaine (je n'ai pas vérifié le chiffre ; peut-être comptait-il DNSSEC jugé, à juste titre, indispensable en 2025), et le serveur de messagerie ne peut pas les ajouter lui-même. Pas de norme, des API privatrices partout. En attendant une éventuelle normalisation, mox utilise un relais, dnsclay, avec qui il parle en mise à jour dynamiques (RFC 2136), charge à lui de relayer via différents dorsaux vers les API de l'hébergeur DNS.

La réunion a aussi été l'occasion de parler du projet "*Structured email*" <https://structured.email/> à l'IETF, projet visant à normaliser des messages structurés et donc facilement compréhensibles par un programme (par exemple des confirmations de commande). Le groupe de travail a plusieurs documents sur le feu <https://datatracker.ietf.org/wg/sml/documents/>.

J'ai bien sûr suivi la réunion <https://fosdem.org/2025/schedule/track/dns/> DNS. Valentin Gosu, de Mozilla, a présenté la difficulté de faire des résolutions DNS depuis une application typique. D'accord, il y a `getaddrinfo` qui, en gros, marche, et partout, mais qui a plein de limites, par exemple il ne donne pas le TTL (rappel, car l'orateur ne l'a pas mentionné : la résolution de noms en adresses IP n'utilise pas forcément le DNS). Il n'indique pas si la réponse a été validée par DNSSEC. Et il ne permet de récupérer que les adresses IP, alors qu'on voudrait, par exemple, les enregistrements HTTPS (RFC 9460). Il existe des API spécifiques à une plate-forme donnée (`DNSQuery_A` sur Windows) pour avoir ces informations, mais qui sont plus ou moins documentées, et marchent plus ou moins bien (par exemple selon les versions de Windows). Bref, on l'oublie souvent, mais une des motivations de DoH (RFC 8484) était de permettre à une application comme Firefox de faire de la résolution DNS proprement.

Alexey Milovidov a fait un amusant exposé sur sa mesure systématique des enregistrements PTR pour tout l'Internet IPv4. Vous pouvez en voir une jolie visualisation <https://reversedns.space/

>. Chaque bloc d'une même couleur concerne des adresses IP proches et résolvant vers des noms de domaine proches. N'hésitez pas à zoomer et à cliquer, c'est amusant. (Défi : réfléchir à comment cela pourrait se faire en IPv6.)

Peter van Dijk (remplaçant Rémi Gacogne, empêché) a présenté le travail (qui n'a pas été un échec mais n'a pas complètement marché) pour réduire l'empreinte de dnsdist <https://www.dnsdist.org/> et le rendre ainsi utilisable sur de petits routeurs utilisant OpenWrt. (Au passage, dnsdist est le logiciel derrière mon résolveur DNS public <https://doh.bortzmeyer.fr/policy>.) Beaucoup de choses ont été essayées (options du compilateur, retrait d'une partie du code…) sans trop de résultat. Changer de bibliothèque (remplacer OpenSSL par wolfSSL) n'a pas beaucoup aidé non plus, d'autant plus que le code de ces bibliothèques peut quand même être chargé par une autre application (il faudrait qu'elles changent toutes en même temps).

Ah, et comme le DNS est très politique, Farzaneh Badiei (Digital Medusa <https://digitalmedusa.org/>) a présenté le travail en cours d'analyse de l'utilisation des résolveurs DNS <https://www.bortzmeyer.org/resolveur-dns.html> publics. Ils sont utilisés pour des raisons diverses. Par exemple, dans plusieurs pays (dont la France), ils servent entre autre à contourner la censure faite via des résolveurs menteurs. Digital Medusa a déjà produit un rapport préliminaire <https://digitalmedusa.org/wp-content/uploads/2023/12/Upload-DNS-Resolvers-First-Draft-October.pdf> dont je vous recommande la lecture.

La réunion Retrocomputing <https://fosdem.org/2025/schedule/track/retrocomputing/> est comme d'habitude une des plus pittoresques du FOSDEM. C'est ainsi que Raphaël Zumer a expliqué comment casser les mots de passe écrits en japonais pour pouvoir jouer à un ancien jeu vidéo japonais. (La devise de Retrocomputing pourrait être « si c'est utile, ça n'a pas sa place ici ».)

Hans Hübner a présenté son travail sur le "*Bildschirmtext*", le « Minitel allemand ». En raison du peu de matériels fabriqués, il n'avait pas accès à un vrai terminal (et encore moins à ses serveurs) et donc a tout reconstitué en logiciel. Il a réussi à trouver certaines documentations (épaisses couches de papier mal imprimé à partir d'un texte tapé à la machine), notamment auprès de musées <https://btx-museum.de/>. Le logiciel original semble perdu (Software Heritage n'existait pas encore, cela permet de comprendre son importance).

Les auteurs (pas les simples lecteurs) de pages "*Bildschirmtext*" devaient utiliser un clavier spécial. La part la plus importante de son exposé était consacrée à la reproduction de ce clavier en Javascript, pour que cela puisse tourner dans un navigateur Web (avec une description en XML et un programme XSLT pour produire du SVG…). Beaucoup de "*hacks*" ont enchanté l'audience comme le travail pour fabriquer une police de caractères qui ressemble à celle du "*Bildschirmtext*", uniquement à partir des copies d'écran existantes. Le code du « serveur » a été refait en Common Lisp.

Le résultat de son travail est visible en ligne <https://btx.vaxbusters.org/> (page d'accueil du CCC en 1987). Il faut cliquer sur le clavier (pas sur l'écran).

Un autre exposé, celui de Dmitriy Kostiuk, portait sur l'histoire des souris. Nous avons vu plein de jolies images de vieilles souris, qu'on peut retrouver sur le site de l'auteur <https://mouses.info/>.

Un des meilleurs exposés était celui de Rodrigo Arias Mallo (dans "*Minimalistic computing*" <https://fosdem.org/2025/schedule/track/declarative/>) sur le navigateur Web Dillo. Dillo existe depuis 1999. C'est un navigateur graphique minimaliste (HTML et CSS mais pas JavaScript, il est donc parfait pour les sites Web avec du contenu, de l'information, mais pas pour les applications). Si vous vous préoccupez de l'empreinte environnementale du numérique, commencez par regarder si votre

site Web est visible avec Dillo (celui de l'ADEME n'est pas visible, en raison d'un problème TLS ; la sécurité et l'écologie peuvent être difficiles à concilier). Le projet a été abandonné de 2017 à 2024 (le domaine `dillo.org` étant même perdu) mais Rodrigo Arias Mallo l'a relancé, avec un nouveau domaine <`https://dillo-browser.github.io/`>. Dillo a un système de greffons et il existe des greffons pour voir des pages de manuel, des fichiers locaux ou bien des capsules Gemini.

Bien sûr, le logiciel libre est très politique, son utilisation et son développement posent de nombreuses questions politiques. J'ai apprécié la "*keynote*" sur l'utilisation d'infrastructures privatrices pour développer du logiciel libre. Karen Sandler et Denver Gingerich ont commencé en demandant de lever la main à ceux qui utilisaient Slack. Discord ? Confluence ? Teams ? Jira ? GitHub ? À la fin, la grande majorité des mains dans l'amphi étaient levées. (Personnellement, j'ai quitté GitHub <`https://www.bortzmeyer.org/github-to-gitlab.html`> il y a des années.) Comme dit en conclusion, « "*GitHub is free. Free as in free cocaine.*" ». La discussion après l'exposé a beaucoup porté sur la connexion des forges libres entre elles ; une des critiques les plus souvent entendues sur l'utilisation de forges libres, comme un gitlab auto-hébergé, est qu'il faut se créer un compte dans chaque forge, ne serait-ce que pour créer un ticket. Des solutions techniques sont envisageables (ActivityPub ? Le courrier électronique ?) mais aucune ne s'est encore imposée.

Le FOSDEM, ce sont aussi d'innombrables stands <`https://fosdem.org/2025/stands/`> (comme celui d'Haiku, toujours très suivi et très fréquenté). J'ai eu une longue discussion avec les gens de Qubes OS. Je croyais le projet mort mais, si sa fondatrice, Joanna Rutkowska, est bien partie, le projet est toujours vivant. L'idée de base est d'avoir un système d'exploitation qui soit « raisonnablement sécurisé » (un système « parfaitement sécurisé » serait un système sur un ordinateur éteint). Il faut qu'il soit sécurisé mais qu'on puisse quand même faire des trucs et des machins. (Au contraire des approches "*compliance*", où on ne se soucie que de cocher des cases pour être en conformité et où on se moque de si les gens pourront travailler ou pas.) Pour atteindre cet objectif, Qubes OS partitionne le système en plusieurs machines virtuelles (utilisant Xen, qui est toujours un projet actif). Par exemple, l'une est dédiée au travail sur un projet très confidentiel, et n'a pas accès à l'Internet, une autre sert à regarder Pornhub (non, je plaisante, mais vous avez l'idée). De même, une machine virtuelle est dédiée à USB, pour limiter le risque d'une attaque type Stuxnet. Qubes OS fait tourner des applications non modifiées, avec toutes leurs bogues, donc, mais les isole les unes des autres pour limiter les conséquences d'une faille de sécurité. Le gestionnaire de fenêtres décore les machines virtuelles de couleurs différentes pour limiter le risque de confusion. Par défaut, les machines virtuelles ne communiquent pas (par exemple, le copier-coller ne marche qu'entre fenêtres d'une même machine virtuelle) mais rappelez-vous que le but est de faire de la sécurité raisonnable et utilisable donc la communication reste possible, mais explicite, et donc un peu plus compliquée pour que l'utilisateurice soit bien conscient de franchir des frontières.

Juste à côté, il y avait un autre projet de sécurité, qui partait sur une démarche très différente. Genode est un noyau écrit en partant de zéro (alors que Qubes OS utilise Linux+Xen), qui fait tourner des machines virtuelles bien isolées. Pour pouvoir gérer le maximum de matériel, Genode peut utiliser les pilotes de Linux en mode utilisateur (pas dans le noyau).

L'allée des "*food trucks*", ici celui de Mozilla :