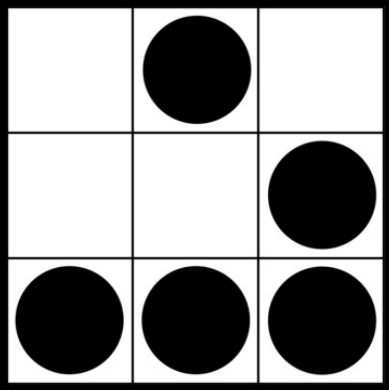
Pour la Liberté

Le Chemin vers Hackerdom

Textes de Eric S. Raymond

Version 9.3



HACKERS

Pour la Liberté...

Le Chemin vers Hackerdom

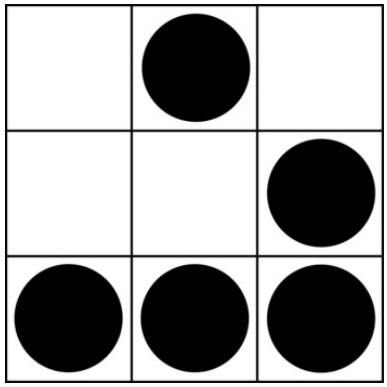
Il existe une communauté, une culture commune, d'experts en programmation et de gourous de la gestion de réseau dont les racines remontent à quelques décennies, au temps des mini-ordinateurs à exploitation partagée et des premières tentatives du réseau ARPAnet. Ce sont les membres de cette culture qui ont introduit le terme hacker. Les hackers ont construit Internet. Ils ont fait du système d'exploitation UNIX ce qu'il est aujourd'hui. Ils font tourner le réseau Usenet. Ils font fonctionner le Web. Si vous faites partie de cette culture, si vous y avez contribué et d'autres membres de cette communauté savent qui vous êtes et vous appellent un hacker, alors vous êtes un hacker.

Eric S. Raymond

Table des matières

Une brève histoire des Hackers

Auteur : Eric S. Raymond
Traducteur : Sébastien Blondeel
1998 ; traduit en février 1999



Essai publié dans le livre Open Sources — Voices from the Open Source Revolution, ISBN 1-56592-582-3, janvier 1999, édité par Chris DiBona, Sam Ockman, et Mark Stone chez O'Reilly & Associates. La version originale de ce document est abritée par le site web personnel d'Eric S. Raymond ;

Prologue : les Vrais Programmeurs

Au commencement, il y avait les *Vrais Programmeurs* [1]

Ce n'est pas le nom qu'ils se donnaient. Ils ne considéraient pas être des « *hackers* », non plus, ou quoi que ce soit en particulier ; le sobriquet « Vrai Programmeur » n'a pas vu le jour avant 1980. Mais à partir de 1945, les techniques de l'informatique ont attiré nombreux des individus parmi les esprits plus brillants et les plus créateurs du monde. Suivant la trace de l'*ENIAC* de MM. Eckert et Mauchly, on a vu se mettre en place, de manière plus ou moins continue, une culture technique émergente de programmeurs enthousiastes, qui construisaient et s'amusaient avec du logiciel pour le plaisir.

Le Vrai Programmeur type était un ingénieur ou un physicien. Il portait des chaussettes blanches et des chemises et cravates en polyester, chaussait des lunettes épaisses et codait en langage machine, en langage d'assemblage, en *FORTRAN* et en une demi-douzaine de langages aujourd'hui oubliés. C'étaient les précurseurs de la culture des *hackers*, les héros trop méconnus de sa préhistoire.

De la fin de la deuxième guerre mondiale au début des années 70, dans ces jours grandioses de la programmation par lots et des « grosses machines centrales »[2], les Vrais Programmeurs représentaient la culture technique dominante dans l'informatique. Certaines portions du folklore vénéré des *hackers* remontent à cette époque, comme la célèbre histoire de Mel (qu'on trouve dans le fichier *Jargon*), diverses listes de lois de *Murphy*, et l'affiche « *Blinkenlights* », qui se moque des Allemands, et qu'on trouve encore dans de nombreuses salles d'ordinateurs.

Certains de ceux qui ont grandi au sein de la culture des « Vrais Programmeurs » sont restés actifs jusque dans les années 1990. Seymour Cray, concepteur de la lignée *Cray* de super-ordinateurs, a la réputation d'avoir pianoté un système d'exploitation complet de son cru, dans un ordinateur de son cru. En octal. Sans faire une seule erreur. Et ça a fonctionné. Le Vrai Programmeur dans toute sa splendeur.

Plus discret, Stan Kelly-Bootle, auteur du *The Devil's DP Dictionary* (dictionnaire du diable, *McGraw-Hill*, 1981) et chroniqueur hors pair du folklore des *hackers*, a programmé le *Manchester Mark I*, premier ordinateur qui stockait les programmes de façon numérique, en 1948. De nos jours, il tient des rubriques techniques humoristiques dans des magazines traitant d'informatique, souvent sous la forme d'un dialogue, vigoureux et entendu, avec la culture des *hackers* d'aujourd'hui.

D'autres, comme David E. Lundstrom, ont couché sur papier les anecdotes de ces vertes années (*A Few Good Men From UNIVAC*,[3] 1987).

On doit à la culture des « Vrais Programmeurs » la montée de l'informatique interactive, des universités, et des réseaux. Ils ont donné naissance à une tradition d'ingénierie continue qui devait déboucher, à terme, sur la culture du *hacker* de logiciel libre d'aujourd'hui.

[1] Certaines traductions choisissent de rendre cette note humoristique par le mot « les Véritables ».

[2] En anglais, « *big iron* » « *mainframes* ».

[3] « Des hommes d'honneur à *UNIVAC* ».

Les premiers *hackers*

On peut placer le point de départ de la culture des *hackers*, telle qu'on la connaît, en 1961, l'année où le *MIT* [1] a fait l'acquisition du premier *PDP-1*. Le comité *Signaux et puissance* du club de modèles réduits ferroviaires du *MIT* [2] éleva la machine au rang de leur jouet technique favori et inventa des outils de programmation, un jargon, et toute une culture associée, dont on trouve encore de nombreuses traces aujourd'hui. Ces premières années sont contées dans la première partie du livre *Hackers*, écrit par Steve Levy (*Anchor/Doubleday*, 1984).

Il semble qu'on doit à la culture informatique du *MIT* la première adoption du terme « *hacker* ». Les *hackers* du *TMRC* ont formé le noyau du laboratoire d'intelligence artificielle (*IA*) du *MIT*, locomotive mondiale en matière de recherche en IA au début des années 1980. Et leur influence s'est répandue bien plus loin après 1969, la première année de l'*ARPAnet*.

L'*ARPAnet* était le premier réseau d'ordinateurs transcontinental et à haut débit. Il avait été construit par le ministère de la défense pour expérimenter les communications numériques, mais a eu pour effet de relier des centaines d'universités, de fournisseurs de l'armée, et de laboratoires de recherche. Il a permis aux chercheurs du monde entier d'échanger des informations avec une vitesse et une souplesse inégalées jusqu'alors, donnant un coup de fouet au travail collaboratif et augmentant énormément l'intensité et l'allure des avancées techniques.

Mais l'*ARPAnet* a eu également un autre effet. Ses autoroutes électroniques ont réuni des *hackers* de tous les États-Unis d'Amérique en une masse critique ; au lieu de demeurer dans des groupes isolés, qui développaient autant de cultures propres et éphémères, ils se sont découvert (ou réinventé) une tribu de réseau.

Les premiers artefacts intentionnels de la culture des *hackers* — les premières listes de jargon, les premières satires, les premières discussions timides de l'éthique — furent tous propagés sur l'*ARPAnet* dans ses jeunes années (la première version du fichier *Jargon*, pour citer un exemple majeur, date de 1973). La culture des hackers s'est développée dans les universités connectées au réseau, et en particulier (mais pas exclusivement) dans leurs sections d'informatique.

D'un point de vue culturel, le laboratoire d'IA du *MIT* était le premier de ses pairs à la fin des années 1960. Mais le laboratoire d'intelligence artificielle de l'université de Stanford (*SAIL*) et, plus tard, l'université *Carnegie-Mellon* (*CMU*), ont commencé à jouer un rôle comparable. Tous trois étaient des centres florissants pour l'informatique et la recherche en IA. Tous trois attiraient à eux des gens brillants, qui ont apporté énormément à la culture des *hackers*, tant d'un point de vue technique que folklorique.

Pour comprendre la suite, cependant, il nous faut examiner de plus près les ordinateurs eux-mêmes, car la montée et la chute du Laboratoire furent toutes deux dues à des vagues de changements dans les techniques de l'informatique.

Depuis l'époque du *PDP-1*, la destinée de la culture des *hackers* avait été liée à la série de mini-ordinateurs *PDP* de la société *Digital Equipment Corporation*. La société *DEC* a ouvert la voie de l'informatique interactive commerciale et des systèmes d'exploitation à temps partagé. Leurs machines étant souples, puissantes, et relativement bon marché pour l'époque, de nombreuses universités s'en procurèrent.

La culture des *hackers* s'est développée dans un médium de partage de temps peu coûteux, et l'*ARPAnet*, pour la majeure partie de son existence, était principalement constitué de machines *DEC*. La plus importante de ces machines était le *PDP-10*, qui sortit en 1967. La *10* est restée la machine préférée des *hackers* pendant près de quinze ans ; on se rappelle encore avec tendresse et nostalgie *TOPS-10* (le système d'exploitation de la société *DEC* pour cette machine) et *MACRO-10* (son langage d'assemblage), et ils ont une place de choix dans le jargon et dans le folklore des *hackers*.

Le *MIT*, qui utilisait pourtant la *PDP-10*, comme tout le monde, a choisi une voie légèrement différente ; ils ont complètement rejeté le logiciel que la société *DEC* proposait pour le *PDP-10* pour lui préférer leur propre système d'exploitation, le légendaire *ITS*.

ITS signifiait *Incompatible Timesharing System* (système à temps partagé incompatible), ce qui donne une bonne idée de leurs dispositions. Ils voulaient travailler à leur manière. Heureusement pour nous tous, les gens du *MIT* étaient aussi intelligents qu'ils étaient arrogants. *ITS*, capricieux, excentrique, et parfois (si pas toujours) bogué, renfermait toute une série d'innovations techniques brillantes, et on peut soutenir, encore aujourd'hui, que c'est le système à temps partagé qui détient le record de la plus longue exploitation en continu.

ITS lui-même avait été écrit en langage d'assemblage, mais de nombreux projets relatifs à *ITS* ont été écrits dans le langage d'IA *LISP*. *LISP* était de loin plus puissant et plus souple tout autre langage de son temps ; en fait, il tient toujours la dragée haute à la plupart des langages d'aujourd'hui, car il reste mieux conçu, vingt-cinq ans plus tard. *LISP* a permis aux *hackers* de l'*ITS* de réfléchir en des termes nouveaux et

créateurs. C'était l'un des facteurs principaux de leur réussite, et il demeure l'un des langages favoris des *hackers*.

On utilise encore aujourd'hui de nombreuses créations techniques de la culture d'*ITS* ; l'éditeur *Emacs* est probablement l'exemple le mieux connu. Et le folklore de l'*ITS* reste encore très « vivant » au sein des *hackers*, comme on peut le constater dans le fichier Jargon.

SAIL et *CMU* étaient eux aussi très actifs. De nombreux cadres des *hackers* qui ont grandi autour du *PDP-10* de *SAIL* sont devenus plus tard d'éminentes personnalités dans le monde de l'ordinateur personnel et dans les interfaces utilisateur employées aujourd'hui, à base de fenêtres, d'icônes et de souris. Et les *hackers* de *CMU* travaillaient sur ce qui allait mener aux premières applications pratiques à grande échelle de systèmes experts et de robotique industrielle.

Le *Xerox PARC*, le célèbre centre de recherche de Palo Alto, a lui aussi joué un rôle important dans la culture des *hackers*. Pendant plus de dix ans, du début des années 1970 au milieu des années 1980, le *PARC* a produit un nombre ahurissant d'innovations révolutionnaires, tant au niveau du matériel qu'au niveau du logiciel. C'est là que les interfaces modernes, à base de souris, de fenêtres, et d'icônes, ont été mises au point. C'est là qu'on a inventé l'imprimante laser, et le réseau local (*LAN*) ; et les machines de la série D du *PARC* laissaient présager, avec dix ans d'avance, les ordinateurs personnels puissants du milieu des années 80. Malheureusement, ces génies n'étaient pas prophètes en leur propre société ; à tel point qu'on a pris l'habitude de plaisanter en décrivant le *PARC* comme un lieu caractérisé par le fait qu'on y développait de brillantes idées... pour les autres. Ils ont influencé les *hackers* de manière décisive.

Les cultures de l'*ARPAnet* et du *PDP-10* se sont renforcées et diversifiées tout au long des années 1970. Les listes de diffusion par courrier électronique, réservées jusque là à des groupes partageant un intérêt particulier étalés sur des continents entiers ont commencé à être utilisées dans des buts plus sociaux et de récréation. *DARPA* a délibérément fermé les yeux sur toutes ces activités annexes, techniquement « non autorisées » ; ils ont compris que la surcharge induite, minime, était un prix à payer bien faible pour attirer toute une génération de brillants jeunes gens vers l'informatique.

La plus connue des listes de diffusion à caractère « social » d'*ARPAnet* était peut-être la liste *SF-LOVERS* (amoureux de SF), qui abritait les férus de science-fiction ; elle est toujours bien vivante aujourd'hui, sur l'« Internet », réseau un peu plus grand, en lequel l'*ARPAnet* a évolué. Mais il y en avait de nombreuses autres, ouvrant la voie à un style de communication qui serait plus tard commercialisé par des services de temps partagé à but lucratif, tels que les sociétés *CompuServe*, *GEnie*, et *Prodigy*.

[1] Institut de Technologie du Massachusetts, l'une des universités les plus prestigieuses des États-Unis d'Amérique.

[2] « *MIT Tech Model Railroad Club* », ou « *TMRC* ».

La montée d'*Unix*

Pendant ce temps, cependant, au plus profond de l'État du Nouveau Jersey, quelque chose était sur les rails depuis 1969, qui allait à terme faire de l'ombre à la tradition du *PDP-10*. L'*ARPAnet* a vu le jour la même année que celle où un *hacker* des laboratoires *Bell*, Ken Thompson, a inventé *Unix*.

M. Thompson avait travaillé sur le développement d'un système d'exploitation à temps partagé appelé *Multics*, qui partageait avec *ITS* des ancêtres communs. *Multics* fut un banc de tests pour des idées importantes, comme la manière dont on pouvait dissimuler la complexité d'un système d'exploitation au coeur de ce dernier, sans rien en laisser transparaître à l'utilisateur ni même à la plupart des programmeurs. L'idée était de faciliter grandement l'utilisation (et la programmation !) de *Multics* de l'extérieur, afin de pouvoir vraiment abattre du travail.

Les laboratoires *Bell* se sont retirés du projet quand *Multics* a montré des signes de boursouflement superflu (ce système a plus tard été mis sur le marché par la société *Honeywell* mais n'a jamais connu de succès). Ken Thompson regrettait l'environnement de *Multics*, et a commencé à implanter en s'amusant un mélange des idées de *Multics* et de certaines des siennes propres sur un *DEC PDP-7* qu'il avait sauvé du

rebut.

Un autre *hacker*, appelé Dennis Ritchie, avait inventé un nouveau langage, le « C », pour que M. Thompson puisse l'utiliser dans son embryon d'*Unix*. À l'instar d'*Unix*, C était conçu pour être agréable, sans contraintes, et souple. Aux laboratoires *Bell*, le mot a circulé, et ces outils ont attiré l'attention, jusqu'à être renforcés, en 1971, par une prime remportée par MM. Thompson et Ritchie, pour produire ce qu'on appellerait maintenant un système d'automatisation de bureau pour usage interne. Mais Thompson et Ritchie visaient de plus grands honneurs.

Traditionnellement, les systèmes d'exploitation avaient été écrits en langage d'assemblage, ardu, pour fonctionner le plus rapidement possible sur leurs machines hôtes. MM. Thompson et Ritchie furent parmi les premiers à comprendre que le matériel et les techniques de compilation avaient fait suffisamment de progrès pour permettre d'écrire tout un système d'exploitation en langage C, et en 1974 tout l'environnement avait été porté avec succès sur plusieurs machines de types différents.

Cela n'avait jamais été réalisé auparavant, et les implications étaient énormes. Si *Unix* pouvait présenter le même visage, les mêmes possibilités, sur des machines de nombreux types différents, il pourrait servir d'environnement logiciel commun à toutes ces machines. Les utilisateurs ne souffriraient plus des coûts des nouvelles conceptions de logiciels chaque fois qu'une machine deviendrait obsolète. Les *hackers* pourraient transporter des boîtes à outils logicielles d'une machine à l'autre, plutôt que de devoir réinventer la roue et l'eau chaude à chaque fois.

En plus de leur caractère portable, *Unix* et C avaient d'autres atouts dans leur manche, et pas des moindres. Tous deux avaient été construits en suivant la philosophie du « *Keep It Simple, Stupid* » (acronyme signifiant « baiser » et dont la version développée conseille de *faire les choses simplement, sans prétentions*). Un programmeur pouvait facilement retenir la totalité de la structure logique du C (à la différence de la plupart des autres langages, antérieurs ou postérieurs) sans devoir se référer sans cesse à des manuels ; et *Unix* était structuré comme une boîte à outils souple de programmes simples mis au point dans le but de se combiner utilement les uns avec les autres.

Ces combinaisons se sont révélées pouvoir s'adapter à une large gamme de tâches informatiques, à la plupart desquelles leurs concepteurs n'avaient même pas songé. *Unix* s'est rapidement développé sous l'impulsion de la société *AT&T*, malgré l'absence de programme d'assistance formel. En 1980, il s'était répandu sur de nombreux sites informatiques d'universités et de pôles de recherche, et des milliers de *hackers* en faisaient leur environnement de travail privilégié.

Les chevaux de labour de la culture *Unix* des premières années étaient les *PDP-11* et ses descendants, les *VAX*. Mais *Unix* étant portable, il pouvait fonctionner quasiment à l'identique sur un plus grand nombre de machines, qu'on pouvait trouver sur l'*ARPAnet*. Et personne n'utilisait de langage d'assemblage ; les programmes écrits en C étaient facilement portés d'une machine à l'autre.

Unix disposait même, en quelque sorte, de son propre protocole réseau — le protocole de copie d'*Unix* à *Unix* (*UUCP*) : lent et (alors) peu fiable, il avait l'avantage d'être peu coûteux. Deux machines *Unix* quelconques pouvaient s'échanger du courrier électronique point à point grâce à des lignes de téléphone ordinaires ; cette fonctionnalité était construite dans le système, ce n'était pas un extra facultatif. Les sites *Unix* ont commencé à former un réseau dans le réseau, accompagné par une culture dans la culture des *hackers*. 1980 vit la première mouture de l'*Usenet*, réseau qui dépasserait bientôt l'*ARPAnet*.

Certaines sites *Unix* se trouvaient eux-mêmes sur l'*ARPAnet*. Les cultures *PDP-10* et *Unix* ont commencé à se rencontrer et à se mêler, mais ce mélange n'était pas toujours heureux. Les *hackers PDP-10* avaient tendance à considérer les gens d'*Unix* comme une bande de parvenus, qui utilisaient des outils d'allure ridicule et primitive si on les comparait aux adorables complexités baroques de *LISP* et d'*ITS*. « Couteaux en silex et peaux de bêtes ! » murmuraient-ils.

Et il existait encore un troisième courant. Le premier ordinateur personnel avait été mis sur le marché en 1975. La société *Apple* fut fondée en 1977, et les avancées ont suivi à un rythme effréné et incroyable dans les années qui ont suivi. Le potentiel des micro-ordinateurs était limpide, et ils ont attiré une autre génération de jeunes *hackers* brillants. Ils utilisaient le langage *BASIC*, qui était si primitif que les partisans de *PDP-10* comme les aficionados d'*Unix* le jugeaient indigne de leur mépris.

La fin du bon vieux temps

Telle était la situation en 1980 : trois cultures, qui se recouvraient en partie mais qui étaient organisées autour de techniques bien distinctes. La culture *ARPAnet/PDP-10*, vouée au *LISP*, au *MACRO*, au *TOPS-10*, et à *ITS*. Les gens d'*Unix* et du C, forts de leurs *PDP-11*, de leurs *VAX*, et de leurs connexions téléphoniques rudimentaires. Et une horde anarchique d'enthousiastes des premiers micro-ordinateurs, déterminés à voler aux autres le pouvoir de faire de l'informatique.

Parmi ces cultures, la culture de l'*ITS* pouvait s'enorgueillir de sa position dominante. Mais l'orage menaçait, et les nuages s'accumulaient au-dessus du Laboratoire. Les techniques utilisées dans le *PDP-10* vieillissaient, et le Laboratoire lui-même était divisé par des factions au cours des premières tentatives de commercialisation des techniques de l'IA. Certains, parmi les meilleurs du Laboratoire (et du *SAIL* et de *CMU*) ont succombé aux sirènes d'un emploi très lucratif au sein d'une société nouvelle.

Le coup de grâce est venu en 1983, quand la société *DEC* a cessé d'assurer l'avenir de la gamme *PDP-10* pour se concentrer sur les modèles *PDP-11* et *VAX*. *ITS* devrait en rester là. Puisqu'il n'était pas portable, il aurait fallu déployer plus d'efforts que quiconque ne pouvait se le permettre pour porter *ITS* sur les nouveaux matériels. C'est la variante d'*Unix* de Berkeley, qui fonctionnait sur un *VAX*, qui est devenu le système des *hackers* par excellence, et quiconque gardait un oeil fixé sur l'avenir pouvait deviner que les micro-ordinateurs augmentaient si rapidement en puissance que bientôt ils balaieraient tout sur leur passage.

C'est autour de cette époque que M. Levy a rédigé le livre *Hackers*. L'une de ses sources privilégiées fut Richard M. Stallman (l'inventeur d'*Emacs*), un chef de file au Laboratoire, et le plus féroce opposant à la commercialisation des techniques mises au point par le Laboratoire.

M. Stallman (qu'on connaît mieux par ses initiales, qui constituent aussi son nom de *login*, *RMS*) a continué ; il a créé la fondation du logiciel libre (*FSF*) et s'est consacré à la production de logiciel libre de première qualité. M. Levy en fait le panégyrique en le présentant comme « le dernier véritable *hacker* », description qui s'est fort heureusement révélée inexacte.

Le grand projet de M. Stallman personnifiait joliment la transition vécue par la culture des *hackers* au début des années 80 — en 1982, il a entrepris la construction d'un clone complet d'*Unix*, écrit en C et librement disponible. Ainsi, on retrouvait l'esprit et la tradition de l'*ITS* dans une grande partie de la nouvelle culture des *hackers*, centrée autour d'*Unix* et des *VAX*.

C'est aussi à cette époque que les microprocesseurs et les réseaux locaux ont commencé à avoir un impact considérable sur la culture des *hackers*. Les techniques *Ethernet* et le microprocesseur *Motorola 68000* étaient un tandem potentiellement très puissant, et plusieurs sociétés se sont montées pour construire la première génération de ce qu'on appelle de nos jours des stations de travail.

En 1982, un groupe de hackers *Unix* de Berkeley a fondé la société *Sun Microsystems* car ils croyaient que faire fonctionner un système *Unix* sur du matériel relativement bon marché à base de *68000* serait une combinaison gagnante dans une vaste gamme d'applications. Ils avaient raison, et leur vision a placé la première pierre de toute une industrie. Bien qu'encore hors de prix pour la plupart des individuels, les stations de travail étaient bon marché pour les sociétés et pour les universités ; les réseaux de stations de travail (une par utilisateur) ont rapidement remplacé les *VAX* et autres systèmes à temps partagé, plus anciens.

L'ère de l'*Unix* propriétaire

À partir de 1984, au dépouillement de la société *AT&T* et alors qu'*Unix* devenait pour la première fois un produit commercial, c'est une « nation réseau » relativement cohérente, centrée autour de l'*Internet* et de l'*Usenet* (et dont la plupart des membres utilisaient un mini-ordinateur — ou des machines de type stations de travail sous *Unix*) et un vaste arrière-pays d'enthousiastes des micro-ordinateurs qui représentaient l'essentiel de la culture des *hackers*.

Les stations de travail, construites par *Sun* et d'autres, ouvraient de nouveaux horizons aux *hackers*. Elles avaient été conçues dans l'optique de proposer des graphiques de grande qualité et de partager les données grâce au réseau. Dans les années 1980, les *hackers* étaient soucieux des défis posés par la recherche de la meilleure exploitation de ces fonctionnalités, en matière de logiciels et de construction d'outils. L'*Unix* de Berkeley fournissait les protocoles d'*ARPAnet*, qui proposaient une solution au problème du réseau et encourageaient la croissance de l'*Internet*.

On a tenté à plusieurs reprises de dompter les graphiques des stations de travail. C'est le système *X Window* qui s'est imposé. Le fait que les développeurs de *X*, suivant en cela l'éthique des *hackers*, souhaitaient mettre gratuitement à disposition de tous le code source de leur solution fut un critère déterminant dans sa réussite ; et c'est l'*Internet* qui a facilité cette distribution. La victoire de *X* sur les systèmes graphiques propriétaires (notamment celui que proposait la société *Sun*) était un présage important de changements qui, quelques années plus tard, affecteraient profondément le système *Unix* lui-même.

La rivalité *ITS*/*Unix* survivait encore au sein de quelques dissensions qui surgissaient à l'occasion (principalement à l'initiative d'anciens partisans du système *ITS*). Mais la dernière machine employant *ITS* fut arrêtée pour de bon en 1990 ; les zélateurs n'avaient plus rien à défendre et se sont pour la plupart intégrés à la culture *Unix*, en grommelant plus ou moins.

Mais même au sein des *hackers* en réseau, la grande rivalité des années 1980 opposait les défenseurs du système *Unix* de Berkeley aux versions proposées par la société *AT&T*. On trouve encore des copies d'une affiche de l'époque, qui représente à la manière d'une bande dessinée un vaisseau spatial de combat aux ailes en *X* (comme ceux qu'on trouve dans la trilogie *La guerre des étoiles*, très populaire au sein des *hackers*) filant à toute allure pour s'éloigner d'une *Étoile de la Mort* en train d'exploser, et couverte du logo de la société *AT&T*. Les *hackers* de Berkeley aimaient se considérer comme des rebelles s'opposant aux empires des sociétés dépourvues d'âmes. L'*Unix* de la société *AT&T* n'a jamais rattrapé *BSD*/*Sun* en termes de parts de marché, mais il a gagné la guerre des standards. En 1990, les versions d'*AT&T* et de *BSD* étaient devenues plus difficiles à distinguer, chacune ayant emprunté beaucoup d'innovations l'autre.

Au début des années 1990, les capacités des stations de travail de la décennie précédente commençaient à être menacées par les ordinateurs personnels, plus récents, de faible prix, et aux performances élevées, construits autour d'un processeur de type *Intel 80386* ou de l'un de ses descendants. Pour la première fois, des *hackers* pouvaient se permettre, à titre individuel, de s'acheter des machines domestiques comparables en puissance et en capacité de stockage aux mini-ordinateurs qu'on trouvait dix ans auparavant — des *Unix*-ettes capables de proposer tout un environnement de développement et de communiquer sur l'*Internet*.

Le monde *MS-DOS* a benoîtement négligé tout cela d'un air béat. Ces premiers enthousiastes des micro-ordinateurs avaient beau s'être rapidement retrouvés, sur les environnements *MS-DOS* et *MacOS*, quelques ordres de grandeur plus nombreux que ceux qu'on trouvait dans la « nation réseau », ils n'ont jamais formé de culture consciente d'elle-même. Le rythme des changements était si élevé que cinquante cultures techniques différentes ont vu le jour pour s'éteindre aussi rapidement que des éphémères, sans jamais atteindre la stabilité nécessaire au développement d'une tradition commune comportant jargon, folklore, et histoires mythiques. En l'absence d'un véritable réseau, comparable à *UUCP* ou à l'*Internet*, elles n'ont jamais pu devenir elles-mêmes une nation réseau. L'accès grand public aux services commerciaux en ligne tels que *CompuServe* et *GEnie* commençait à prendre forme, mais le fait que les systèmes non *Unix* n'étaient pas livrés avec des outils de développement signifiait qu'il était très difficile d'y compiler du code source. C'est pourquoi il ne s'est développé, dans ces cultures, aucune tradition de *hackers* travaillant de manière collaborative.

Le courant principal des *hackers*, (dés)organisés sur l'*Internet* et qu'on pouvait maintenant clairement assimiler à la culture technique d'*Unix*, se fichait des services commerciaux. Ils voulaient de meilleurs outils et plus d'*Internet*, et des ordinateurs personnels de type *PC* d'architecture 32 bits, qui promettaient de mettre tout cela à portée de la main.

Mais qu'en était-il du logiciel ? Les *Unix* commerciaux demeuraient onéreux, ils coûtaient plusieurs milliers de dollars américains (plusieurs milliers d'euros). Au début des années 1990, plusieurs sociétés ont

tenté de vendre les ports d'*Unix* d'*AT&T* et de *BSD* sur des machines personnelles de type *PC*. Elles ont rencontré un succès fort limité, les prix baissaient peu, et (ce qui était le pire) on ne disposait pas du code source du système d'exploitation, qu'on ne pouvait donc pas modifier et redistribuer. Le modèle traditionnel des entreprises de logiciels ne donnait pas aux *hackers* ce qu'ils souhaitaient.

La fondation du logiciel libre non plus. Le développement du *Hurd*, le noyau de l'*Unix* libre promis depuis longtemps par RMS aux *hackers*, s'est embourbé de nombreuses années et n'a commencé à produire un noyau vaguement utilisable qu'en 1996 (alors que dès 1990 la *FSF* proposait la plupart des autres portions, dont les plus difficiles, d'un système d'exploitation de type *Unix*).

Pis, au début des années 1990, il devenait limpide que dix années d'efforts de commercialisation des *Unix* propriétaires se soldaient par un échec. La promesse d'*Unix*, la portabilité d'une plate-forme à l'autre, avait cédé le pas aux chamailleries induites par une demi-douzaine de versions propriétaires d'*Unix*. Les acteurs du monde *Unix* propriétaire se sont révélés si lourds, si aveugles, et si inaptes à la mercatique, que la société *Microsoft* a pu leur prendre une large portion de leur marché avec son système d'exploitation *MS-Windows*, qui était pourtant étonnamment inférieur sur le plan technique.

Au début de l'année 1993, un observateur hostile avait de quoi penser que l'histoire d'*Unix* était sur le point de se conclure, et qu'avec elle mourrait la bonne fortune de la tribu des *hackers*. Et on ne manquait pas d'observateurs hostiles dans la presse informatique professionnelle, beaucoup d'entre eux ayant régulièrement prédit la mort imminente d'*Unix* selon un rituel semestriel, depuis la fin des années 1970.

À l'époque, il était sage de penser que l'ère du techno-héroïsme individuel avait pris fin, et que l'industrie du logiciel et l'*Internet* naissant seraient peu à peu dominés par des colosses comme la société *Microsoft*. La première génération des *hackers Unix* semblait vieillissante et fatiguée (le groupe de recherche en informatique de Berkeley s'est essoufflé et a perdu son financement en 1994). Le moral était au plus bas.

Heureusement, des gens avaient concocté, à l'insu de la presse professionnelle, et à l'insu même de la plupart des *hackers*, de quoi produire des développements extrêmement encourageants à la fin de l'année 1993 et en 1994. À terme, ils auraient pour effet de faire changer de cap à toute la culture des *hackers*, pour l'emmener vers des réussites dont ils n'auraient osé rêver.

Les premiers *Unix* libres

Un étudiant de l'université d'Helsinki, nommé Linus Torvalds, a comblé le vide laissé par l'échec du *Hurd*. En 1991, il a commencé à développer un noyau *Unix* libre pour les machines de type *Intel 80386*, en utilisant la boîte à outils de la fondation du logiciel libre. Ses premières réussites, rapides, ont attiré de nombreux *hackers* de l'*Internet* qui l'ont aidé à développer *Linux*, un système *Unix* complet, au code source entièrement libre et redistribuable.

Linux ne manquait pas de concurrents. En 1991, à la même époque que les premières expériences de Linus Torvalds, William et Lynne Jolitz portaient, de manière expérimentale, les sources de l'*Unix* de *BSD* sur le *386*. La plupart des observateurs qui comparaient la technique proposée par *BSD* aux rudes premiers efforts de Linus s'attendaient à ce que les ports *BSD* jouent le rôle du système *Unix* libre le plus important sur l'ordinateur personnel de type *PC*.

La spécificité la plus importante de *Linux* n'était technique, mais bien sociologique. Jusqu'au développement de *Linux*, tout le monde croyait que tout logiciel aussi compliqué qu'un système d'exploitation devait être développé de manière soigneusement coordonnée par un petit groupe de gens, étroitement liés. Ce modèle était et demeure représentatif des logiciels commerciaux et des grandes cathédrales libres construites par la fondation du logiciel libre dans les années 1980 ; c'était aussi le cas des projets *FreeBSD*/*NetBSD*/*OpenBSD*, qui ont émergé du port originel de *386BSD* par M. et Mme. Jolitz.

Linux a évolué de manière complètement différente. Dès le début, ou presque, des hordes de *hackers* volontaires se sont échinés à le modifier librement, et la coordination ne se faisait que par l'*Internet*. Ce n'étaient pas des normes rigides ou l'autocratie qui garantissaient la qualité, mais la publication hebdomadaire du logiciel et la collecte des commentaires de centaines d'utilisateurs quelques jours plus tard,

créant ainsi une sorte de sélection darwinienne accélérée sur les mutations introduites par les développeurs. À la surprise générale, ce système a très bien fonctionné.

À la fin de l'année 1993, *Linux* était au niveau, tant du point de vue de la stabilité que du point de vue de la fiabilité, de la plupart des *Unix* commerciaux, et proposait bien plus de logiciels. Il commençait déjà à susciter les ports d'applications logicielles propriétaires. Un effet indirect de ce développement fut de mettre fin aux petits vendeurs d'*Unix* commerciaux — en l'absence de développeurs et de *hackers* à qui vendre leur produit, ils se sont écroulés. L'un des rares survivants, *BSDI* (*Berkeley Systems Design, Incorporated*, n'a dû son salut et son essor qu'au fait d'offrir le code source complet de son système *Unix* à base *BSD* et au fait d'entretenir des relations étroites avec la communauté des *hackers*.

Ces développements sont passés inaperçus à l'époque, même au sein de la communauté des *hackers*, et complètement inaperçus à l'extérieur. La culture des *hackers*, défiant les prédictions répétées de sa mort annoncée, commençait tout juste à renvoyer la balle au monde du logiciel commercial. Mais cette tendance prendrait encore cinq années à s'affirmer franchement.

La grande explosion du *web*

La croissance initiale de *Linux* s'est produite en même temps qu'un autre phénomène : la découverte de l'*Internet* par le grand public. Le début des années 1990 a aussi vu le début d'une florissante industrie de fourniture d'accès à l'*Internet*, qui vendait au particulier le fait de se connecter pour quelques dollars américains par mois (quelques euros). Suite à l'invention de *World Wide Web*, la croissance de l'*Internet*, déjà rapide, a accéléré à une allure folle.

En 1994, l'année ou le groupe de développement de l'*Unix* de Berkeley s'est officiellement dissous, c'est sur diverses versions libres d'*Unix* (*GNU/Linux* et les descendants de *386BSD*) que la plupart des *hackers* focalisaient leurs activités. Le système *GNU/Linux*, distribué commercialement sur des CD-ROM, se vendait comme des petits pains. À la fin de l'année 1995, les sociétés d'informatique les plus importantes vantaient les mérites de leurs logiciels et matériels en matière d'*Internet* !

À la fin des années 1990 les *hackers* se sont concentrés sur le développement de *Linux* et sur les activités liées à l'*Internet*. Le *World Wide Web* a au moins eu pour effet de transformer l'*Internet* en médium de masse, et de nombreux *hackers* des années 1980 et du début des années 1990 fondèrent des sociétés de prestations de services liés à l'*Internet* en vendant ou en offrant aux masses un accès à l'*Internet*.

Le passage de l'*Internet* au premier plan a même apporté aux *hackers* les bribes d'une respectabilité bon teint et ils ont commencé à jouer un rôle politique. En 1994 et 1995, l'activisme *hacker* a saboté la proposition *Clipper*, qui aurait placé la cryptographie forte sous le contrôle du gouvernement (des États-Unis d'Amérique). En 1996, les *hackers* ont mobilisé une large coalition pour défaire le mal nommé « *Communications Decency Act* » (proposition de loi pour contrôler la décence des communications), ou *CDA*, et empêcher ainsi la censure sur l'*Internet*.

La victoire sur le *CDA* nous fait passer d'un registre historique à un registre d'actualités. On entre aussi dans une période où votre historien joue un rôle plus actif que celui d'observateur. Cette narration continue dans « La revanche des *hackers* ».

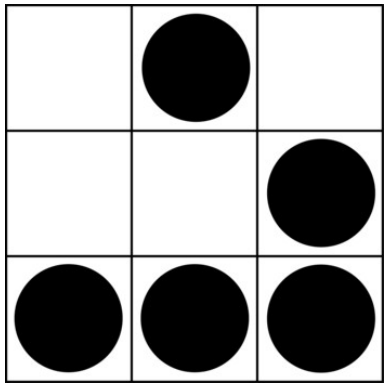
> « Les gouvernements sont tous, plus ou moins, des coalitions opposées au peuple . . . et les dirigeants n'ayant pas plus de morale que ceux qu'ils dirigent . . . on ne peut maintenir le pouvoir d'un gouvernement dans les limites qu'il s'est imposées qu'en lui faisant la démonstration d'une puissance égale à la sienne, le sentiment de tout un peuple. »
>
> Benjamin Franklin Bache, dans un éditorial du *Philadelphia Aurora*, 1794

La revanche des Hackers

Auteur : Eric S. Raymond

Traducteur : Sébastien Blondeel

1998 ; traduit en mars 1999



Essai publié dans le livre Open Sources — Voices from the Open Source Revolution, ISBN 1-56592-582-3, janvier 1999, édité par Chris DiBona, Sam Ockman, et Mark Stone chez O'Reilly & Associates.

La revanche des *hackers*

J'ai écrit la première version de Une brève histoire des hackers en 1996, pour le web. J'avais été fasciné par la culture des *hackers* depuis de longues années, bien avant que j'aie édité la première édition de *The New Hacker's Dictionary*, en 1990. À la fin de l'année 1993, nombreux étaient ceux (et j'en faisais partie) qui en étaient venus à me considérer comme l'historien de la tribu des *hackers*, et leur ethnographe dépêché sur place. Ce rôle me plaisait bien.

Je n'avais pas encore la moindre idée que mon anthropologie amateur se révélerait être un catalyseur significatif qui provoquerait des changements. Je fus plus surpris que quiconque d'observer cela. Mais les conséquences de cette surprise résonnent encore de nos jours sur la culture des *hackers* et les mondes des affaires et des nouvelles techniques.

Dans cet essai, je récapitulerai, de mon point de vue, les événements qui ont immédiatement précédé la « coup de canon de retentissement mondial », de janvier 1998, de la révolution de l'*Open Source*. Je présenterai mes réflexions sur la distance remarquable qui a été parcourue depuis. Enfin, j'offrirai quelques projections tentantes sur l'avenir.

Au delà de la loi de Brooks

J'ai rencontré Linux pour la première fois à la fin de l'année 1993, à travers une distribution sur CD-ROM du pionnier *Yggdrasil*. À cette époque, j'avais déjà été impliqué dans la culture des *hackers* depuis plus de quinze ans. Mes premières expériences remontaient à l'*ARPAnet* primitif de la fin des années 1970 ; j'ai même été brièvement touriste sur les machines *ITS*. J'avais déjà écrit du logiciel libre et je l'avais publié sur l'*Usenet* avant que la *Free Software Foundation* ne voie le jour en 1984, et je fus l'un des premiers contributeurs à la *FSF*. Je venais de publier la deuxième édition de *The New Hacker's Dictionary*. Je pensais très bien connaître la culture des *hackers* --- et ses limitations.

La rencontre avec Linux fut un choc. J'avais beau avoir baroudé de nombreuses années dans la culture des *hackers*, je traînais toujours l'idée reçue que des *hackers* amateurs, même très doués, ne pourraient jamais rassembler suffisamment de ressources ou de talent pour produire un système d'exploitation multi-tâches utilisable. Les développeurs du *Hurd*, après tout, avaient ostensiblement échoué sur ce point pendant une décennie.

Mais là où ils ont échoué, Linus Torvalds et sa communauté ont réussi. Et ils ne se sont pas contentés de ne remplir que les exigences minimales en matière de stabilité et d'interfaces fonctionnant à la *Unix*. Oh non. Ils ont explosé ces critères par leur exubérance et leur flair, en fournissant des centaines de millions d'octets de programmes, de documents, et d'autres ressources. Des suites complètes d'outils pour l'*Internet*, des logiciels de publication de qualité professionnelle, la possibilité d'utiliser le mode graphique, des éditeurs, des jeux --- tout cela, et bien plus encore.

Observer cette débauche de codes merveilleux étalés sous mes yeux fut une expérience bien plus puissante que de me contenter de savoir, d'un point de vue uniquement intellectuel, que toutes les portions existaient probablement déjà quelque part. C'est comme si je m'étais baladé au milieu de piles de pièces de rechange dépareillées pendant des années pour me retrouver face aux même pièces, assemblées sous la forme d'une *Ferrari* rouge et rutilante, portière ouverte, les clefs se balançant sur le contact, et le moteur ronronnant des promesses de puissance...

La tradition des *hackers*, que j'avais observée pendant vingt ans, semblait soudain prendre vie d'une nouvelle et vibrante manière. Dans un certain sens, je faisais déjà partie de cette communauté, car plusieurs de mes projets de logiciel libre avaient été ajoutés à la mêlée. Mais je voulais y pénétrer plus profondément, car chacune des merveilles que j'observais approfondissait aussi mon étonnement. C'était trop beau !

Les coutumes du génie logiciel sont inféodées à la loi de Brooks, qui prédit que lorsque le nombre *N* de programmeurs augmente, le travail accompli augmente en proportion mais que la complexité et la

vulnérabilité aux bogues augmente en *N* au carré. *N* au carré est le nombre de chemins de communication (et d'interfaces de code potentielles) séparant les bases de code des développeurs.

La loi de Brooks prédit qu'un projet comptant des milliers de contributeurs devrait être un capharnaüm floconneux et instable. D'une certaine manière, la communauté de Linux a vaincu l'effet N au carré en produisant un système d'exploitation d'une qualité exceptionnelle. J'étais décidé à comprendre la manière dont ils avaient procédé.

Il m'a fallu trois ans de participation et d'observation de près pour développer une théorie, et une année encore pour la mettre en pratique. Je me suis alors assis à mon bureau et j'ai rédigé « La cathédrale et le bazar » (*CatB*), pour expliquer ce que j'avais vu.

Des mimiques et des mythes

Ce que je voyais autour de moi, c'était une communauté qui avait mis au point la méthode de développement logiciel la plus efficace de tous les temps, *sans même s'en rendre compte* ! C'est-à-dire qu'une pratique efficace s'était mise en place sous la forme d'un ensemble de coutumes, transmises par l'imitation et l'exemple, sans la théorie ou le langage qui permette d'expliquer pourquoi la pratique fonctionnait.

En y repensant, l'absence de cette théorie et de ce langage nous a gêné de deux manières. D'abord, on ne pouvait pas mettre en place une réflexion systématique sur la manière d'améliorer nos propres méthodes. Ensuite, on ne pouvait pas expliquer ou vendre la méthode à d'autres.

À l'époque, seul le premier effet retenait mon attention. Ma seule intention, quand j'ai écrit ce papier, était de donner à la culture des *hackers* le langage approprié, qu'elle utiliserait de manière interne, pour s'expliquer à elle-même. C'est ainsi que j'ai couché sur le papier ce que j'avais vu, en lui donnant l'allure d'une narration et en utilisant des métaphores vivantes et appropriées pour décrire la logique qu'on pouvait deviner derrière ces coutumes.

CatB ne contient pas vraiment de théorie fondamentale. Je n'ai inventé aucune des méthodes qu'il décrit. Ce qui est nouveau, dans ce papier, ce ne sont pas les faits, mais les métaphores et la narration --- une histoire simple et puissante qui encourageait le lecteur à voir les faits sous un jour nouveau. J'essayais d'appliquer le génie mimétique aux mythes fondateurs de la culture des *hackers*.

J'ai d'abord soumis le papier complet au *Linux Kongress*, en mai 1997, en Bavière. L'intense attention et les applaudissements nourris qu'il a suscités de la part d'un public ne contenant que quelques personnes dont l'anglais était la langue maternelle semblaient confirmer que j'étais sur quelque chose d'important. Mais il se trouve que le hasard, qui m'a placé aux côtés de Tim O'Reilly lors du banquet du jeudi soir, a ébranlé un ensemble de conséquences plus important.

J'admirais le style institutionnel des éditions O'Reilly depuis longtemps, je brûlais donc de rencontrer Tim O'Reilly depuis plusieurs années. Notre conversation embrassa de nombreux thèmes (et en particulier notre intérêt commun pour la science-fiction classique) ce qui provoqua mon invitation à la conférence de Perl, donnée par Tim plus tard la même année, afin d'y présenter *CatB*.

Une fois encore, le papier fut bien reçu --- il obtint, en réalité, des acclamations et que la salle se lève pour lui rendre hommage. Le courrier électronique que je recevais m'avait appris que depuis la Bavière, le bouche à oreille avait rempli son office sur l'*Internet*, à propos de *CatB*, plus rapidement qu'un feu de brousse. Nombreux étaient ceux qui dans l'assistance l'avaient déjà lu, et mon discours fut moins une révélation pour eux qu'une occasion de célébrer le nouveau langage, et la prise de conscience qui l'accompagnait. La salle ne s'est pas tant levée pour me rendre hommage que pour célébrer la culture des *hackers* elle-même --- et en cela, elle avait bigrement raison.

Je ne le savais pas encore, mais mon expérience en ingénierie mimétique était sur le point de bouter un feu bien plus important. Certains de ceux qui découvrirent mon discours ce jour-là travaillaient pour la société *Netscape Communications, Inc*. Et cette société avait des problèmes.

Netscape, pionnier des nouvelles techniques de l'*Internet*, extravagant de *Wall Street*, était sur la liste noire de la société *Microsoft*. Cette dernière craignait à juste titre que les normes pour le web, incarnées par

le navigateur de la société *Netscape*, n'érodent le monopole lucratif dont le géant de Redmond disposait alors sur la plate-forme des compatibles PC. Tout le poids de ses milliards, ainsi que ses tactiques inavouables, qui lui vaudraient quelque temps plus tard d'être poursuivi dans le cadre de la loi *antitrust*, étaient alors déployés pour anéantir le navigateur de la société *Netscape*.

Pour la société *Netscape*, le problème était moins le revenu associé à son navigateur (qui ne représentait qu'une faible proportion de ses recettes) que de maintenir une zone de sécurité pour les affaires associées à leur serveur, bien plus lucratives. Si le navigateur *Internet Explorer*, de la société *Microsoft*, se trouvait en position dominante sur le marché, cette dernière pourrait corrompre les protocoles du web en les éloignant des normes ouvertes pour les transformer en canaux propriétaires que seuls ses propres serveurs pourraient proposer.

À l'intérieur de la société *Netscape*, le débat battait son plein sur la manière de contrer la menace. Une option proposée dès le début fut de libérer le code source du navigateur --- mais cette position était difficile à tenir en l'absence de bonnes raisons de croire que cela empêcherait la domination du logiciel *Internet Explorer*.

Je ne le savais pas encore alors, mais *CatB* fut un avocat déterminant de cette position. Au cours de l'hiver 1997, alors que je travaillais sur mon prochain article, tout était prêt pour que la société *Netscape* abandonne les règles du jeu du commerce habituel et offre à ma tribu une occasion sans précédent.

La route de Mountain View

Le 22 janvier 1998, la société *Netscape* annonçait qu'elle publierait le code source de son client pour le web sur l'*Internet*. Je n'appris cette nouvelle que le lendemain, pour apprendre peu après que Jim Barksdale, le PDG, avait présenté mon travail aux journalistes qui couvraient l'événement nationalement comme une « inspiration fondamentale » pour sa décision.

C'est cet événement que les commentateurs de la presse professionnelle en informatique appelleraient plus tard « coup de canon de retentissement mondial » --- et M. Barksdale avait fait de moi son Thomas Paine[1], que je le veuille ou non. Pour la première fois dans l'histoire de la culture des *hackers*, l'une des entreprises préférées du groupe *Fortune 500*[2]avait parié son avenir sur la croyance que les *hackers* avaient raison. Et, plus spécifiquement, que mon analyse de la culture des *hackers* était correcte.

C'est là un choc bien difficile à encaisser. Je n'avais pas vraiment été surpris que *CatB* modifie l'image que la culture des *hackers* avait d'elle-même ; c'est ce que j'avais cherché à faire, après tout. Mais j'ai été soufflé (et c'est peu dire) par les nouvelles du succès qu'il rencontrait à l'extérieur. C'est pourquoi j'ai réfléchi intensément pendant quelques heures, après avoir appris la nouvelle. J'ai réfléchi à l'état de Linux et de la communauté des *hackers*. J'ai réfléchi à *Netscape*. Et je me suis demandé si, personnellement, j'étais assez costaud pour faire le prochain pas.

Il n'était pas difficile de conclure que le fait d'aider la société Netscape à réussir son pari venait d'acquérir le statut de priorité fondamentale pour la culture des *hackers*, et par conséquent, pour moi personnellement. Si ce pari échouait, les *hackers* subiraient probablement l'opprobre de cet échec de plein fouet. Nous serions discrédités pendant encore dix ans. Et ce seraient dix ans de trop.

À cette époque je faisais partie de la culture des *hackers* et je vivais ses différentes phases depuis vingt ans. Pendant vingt ans j'avais observé des idées brillantes, des débuts prometteurs, et des techniques supérieures se faire invariablement écraser par une mercatique bien menée. Pendant vingt ans j'avais observer les *hackers* rêver, suer et construire, pour trop souvent constater que les pairs du vieux méchant *IBM* ou du nouveau méchant *Microsoft* repartaient nantis des récompenses concrètes. Pendant vingt ans j'avais vécu dans un ghetto --- un ghetto raisonnablement confortable, rempli de camarades intéressants, mais emmuré malgré tout derrière une vaste barrière de préjugés, intangible, annonçant « ici, vous ne trouverez que des excentriques ».

L'annonce de *Netscape* avait lézardé cette barrière, pour au moins un court instant ; le monde des affaires avait été secoué dans sa complaisance sur l'idée qu'il se faisait des capacités des « *hackers* ». Mais les

habitudes mentales paresseuses ont une inertie énorme. Si la société *Netscape* échouait, ou peut-être même si elle réussissait, l'expérience pourrait être perçue comme un fait unique et exceptionnel, qu'il serait inutile de tenter de reproduire. Et nous serions de nouveau parqués dans le même ghetto, aux murs un peu plus hauts qu'avant.

Pour éviter cela, il fallait que *Netscape* réussisse. Alors j'ai récapitulé ce que j'avais appris du mode de développement de type « bazar », et j'ai appelé la société *Netscape*, en leur proposant de les aider à développer leur licence et à mettre au point les détails de leur stratégie. Début février, j'ai pris l'avion pour Mountain View à leur demande, j'ai assisté à sept heures de réunions avec divers groupes au sein de leur quartier général, et je les ai aidés à développer les grandes lignes de ce qui deviendrait la licence publique de *Mozilla* et l'organisation *Mozilla*.

J'ai profité de ma présence en ces lieux pour rencontrer plusieurs personnes clefs dans la *Silicon Valley* et dans la communauté Linux des États-Unis d'Amérique (mais ces détails sont contés plus en détail sur la page d'histoire du site web de l' Open Source). Venir en aide à la société *Netscape* était clairement une priorité à court terme, et tous ceux à qui j'ai parlé avaient déjà compris la nécessité d'une stratégie à plus long terme, pour faire suite à la sortie de *Netscape*. Il était temps d'en mettre une au point.

[1] Révolutionnaire américain.

[2] Ensemble des 500 entreprises les plus riches des États-Unis d'Amérique

Les origines du mouvement *Open Source*

Les grandes lignes étaient faciles à deviner. Il nous fallait prendre les arguments pragmatiques et nouveaux que j'avais énoncés dans *CatB*, les développer plus avant, et en faire une promotion poussée en public. Puisque les gens de *Netscape* avaient eux-mêmes intérêt à convaincre leurs investisseurs que cette stratégie n'était pas folle, on pouvait compter sur eux pour nous aider dans le cadre de cette promotion. Nous avons également recruté Tim O'Reilly (et à travers lui, la société *O'Reilly & Associates*) très rapidement.

Cependant, la véritable percée conceptuelle fut pour nous d'admettre qu'il nous fallait monter une *campagne de mercatique* --- et que cela mettrait en oeuvre des techniques de mercatique (spin, construction d'une image, re-branding) pour que tout cela fonctionne.

D'où le terme « *Open Source* », que les premiers participants à ce qui deviendrait plus tard la campagne de l'*Open Source* (et, finalement, l'organisation de l'« Initiative de l'*Open Source* ») ont inventé lors d'une réunion tenue à Mountain View, dans les locaux de VA Research, le 3 février.

Il nous paraissait clair, en regardant en arrière, que le terme « *free software* » avait causé énormément de tort à notre mouvement au cours des années. Une partie en incombait à l'ambiguïté bien connue « *free-speech/free-beer* »[1]. Mais une partie plus importante provenait de quelque chose de pire --- l'association d'idées très répandue entre les termes « *free software* » et l'hostilité au droit de la propriété intellectuelle, le communisme, et d'autres idées que tout décideur en matières de techniques de l'informatique ne porte pas dans son coeur.

Il était, et c'est toujours le cas, hors sujet d'expliquer que la *Free Software Foundation* n'est pas hostile à toute propriété intellectuelle et que sa position n'est pas exactement celle d'une organisation communiste. Nous le savions. Mais nous avons réalisé, sous la pression de la sortie de *Netscape*, que la véritable position de la *FSF* ne comptait pas vraiment. Seul le fait que son évangélisme s'était retourné contre ses prédicateurs importait : désormais la presse professionnelle et l'industrie du logiciel associaient les mots « *free software* » aux stéréotypes négatifs exposés ci-dessus.

La réussite de notre initiative, suite au cas *Netscape*, ne serait possible que si on parvenait à remplacer les stéréotypes négatifs associés à la *FSF* par des stéréotypes positifs choisis par nous --- des contes pragmatiques, doux aux oreilles des gestionnaires et des investisseurs, parlant de fiabilité accrue, de coûts réduits, et de meilleures fonctionnalités.

En termes de mercatique conventionnelle, notre travail consistait à donner au produit une nouvelle image, et à lui construire une réputation qui donnerait envie à l'industrie du logiciel de l'embrasser.

Linus Torvalds a accepté l'idée le lendemain de la réunion. On a commencé à travaillé sur le sujet quelques jours plus tard. Moins d'une semaine plus tard, Bruce Perens avait enregistré le domaine opensource.org et avait mis en ligne la première version du site web de l'*Open Source*. Il a aussi suggéré qu'on adopte comme « définition de l'Open Source » les grandes lignes du logiciel libre mises au point par *Debian*, et il a commencé à enregistrer le terme « *Open Source* » en tant que marque de certification de telle sorte qu'on puisse légalement exiger des gens qu'ils utilisent le terme « *Open Source* » dans le cadre de produits conforme à cette définition.

Même les tactiques particulières, nécessaires pour mettre en place cette stratégie, m'ont paru claires dès ces premiers moments (et on les avait explicitement discutés au cours de la réunion initiale). Les points-clefs :

Oublions la tactique de la conquête par le bas ; il faut convaincre la tête

> L'une des choses les plus claires était que la stratégie historique d'*Unix*, d'un évangélisme de conquête par le cas (reposant sur les ingénieurs qui convaincraient leurs patrons à l'aide d'arguments rationnels) avait été un échec. C'était une stratégie naïve, aisément démentie par la société *Microsoft*. De plus, la percée de la société *Netscape* ne provenait pas d'un ingénieur, mais d'un décideur stratégique (Jim Barksdale) qui avait compris tout cela et avait imposé sa vision des choses à ses subalternes.
>
> La conclusion s'imposait d'elle-même. Au lieu de conquérir la base, il nous faudrait cibler par notre discours, les directions --- en visant directement les directions générales, techniques et informatiques.

Linux est notre meilleur atout.

> Il nous faut faire de Linux notre porte-étendard. Oui, on trouve d'autres exemples dans le monde de l'*Open Source*, et la campagne leur rendra un hommage respectueux --- mais Linux a le nom le plus connu, la plus grande base installée, et la plus grande communauté de développeurs. Si Linux ne peut pas consolider la percée, rien d'autre, d'un point de vue pragmatique, n'a la moindre chance.

Capturons les *Fortune 500*

> D'autres segments du marché dépensent plus d'argent (les PME/PMI et entreprises familiales en sont l'illustration la plus évidente) mais ces marchés sont plus diffus et difficiles à cibler. Les entreprises de *Fortune 500* ne se contentent pas de disposer de quantités d'argent phénoménales, elles les concentrent là où il est facile d'en approcher. C'est pourquoi l'industrie du logiciel est en grande partie aux ordres des *Fortune 500*. C'est par conséquent le *Fortune 500* qu'il nous faut convaincre.

Mettons dans le coup les journaux prestigieux qui sont au service des *Fortune 500*

> Le choix de cibler les *Fortune 500* implique de capturer les médias qui forgent le climat des opinions des décideurs et des investisseurs les plus importants ; pour être précis, il s'agit des publications suivantes : *the New York Times*, *the Wall Street Journal*, *the Economist*, *Forbes*, et *Barron's Magazine*.
>
> À ce sujet, il est nécessaire mais insuffisant de mettre dans le coup la presse technique informatique ; cela n'est utile qu'en tant que pré-condition pour insuffler la Bourse de *Wall Street* elle-même dans les médias les plus en vue.

Inculquons aux *hackers* les tactiques de la mercatique de guérilla

> Il était tout aussi clair que l'éducation de la communauté des *hackers* serait aussi importante que notre approche du monde réel. À quoi bon envoyer une poignée d'ambassadeurs tenir un discours efficace si, sur le terrain, la plupart des *hackers* en tenaient un autre, qui ne convaincrait personne sinon eux-mêmes ?

Utilisons la marque de certification de l'*Open Source* pour garantir la pureté des choses

> L'une des menaces qui nous attendait était la possibilité que le terme «*Open Source*» soit «récupéré et amélioré» par la *Microsoft* ou une autre société importante, le corrompant au passage, en annihilant notre message. C'est pour cette raison que Bruce Perens et moi-même avons rapidement décidé d'enregistrer ce terme comme une marque de certification et de le lier à la définition de l'*Open Source* (qui est une copie des grandes lignes du logiciel libre mises au point par *Debian*). Cela nous permettrait de dissuader les esprits chagrins sous la menace d'une poursuite en justice.

[1] Les anglais utilisent le même terme pour parler d'« expression libre » et de « bière gratuite », aussi le logiciel libre est-il souvent perçu comme du vulgaire « logiciel gratuit », ce qui fait fuir les industriels.

Révolutionnaire malgré moi

La mise au point de cette stratégie fut assez facile. Le plus dur (en tout cas, pour moi) fut d'accepter le rôle que j'aurais à y tenir.

J'avais compris une chose dès le début : c'est que la presse fait la sourde oreille aux abstractions. Ils n'écrivent rien sur des idées que ne défendent pas les personnalités les plus en vue du moment. Il faut que tout ne soit qu'histoires, drames, conflits, larmes et sang. Sans quoi, la plupart des journalistes retournent se coucher --- et s'ils continuent malgré tout, ce sont leurs rédacteurs en chef qui opposeront leur veto.

C'est pourquoi je savais qu'on aurait besoin d'une personne aux caractéristiques très précises pour faire front à la réponse de la communauté face à l'opportunité de *Netscape*. Il nous fallait un brandon, un porte-drapeau, un propagandiste, un ambassadeur, un évangéliste --- quelqu'un qui sache danser et chanter sur tous les toits, séduire les journalistes, fricoter avec les PDG, et asséner de grands coups sur la machine des médias jusqu'à ce que ses rouages tournent dans l'autre sens et proclament : « C'est la révolution ! »

À la différence de celui de la plupart des *hackers*, mon cerveau est celui d'un extraverti et j'ai déjà une solide expérience des contacts avec la presse. En examinant mon entourage, je n'ai trouvé personne de plus qualifié que moi pour tenir le rôle de l'évangéliste. Mais je ne voulais pas de ce travail, car je savais qu'il me mènerait la vie dure pendant des mois, peut-être des années. Je n'aurais plus d'intimité. La presse grand public me décrirait probablement comme un informaticien autiste et (ce qui est pire) je serais méprisé par une proportion significative des miens, qui me considéreraient comme un vendu ou quelqu'un qui tire la couverture à lui. Et, pire que tout le reste, je n'aurais plus le temps de programmer !

Je devais me poser la question : en as-tu suffisamment marre d'observer ta tribu perdre les batailles pour faire ce qu'il en coûte de remporter la victoire ? J'ai décidé d'y répondre par l'affirmative --- et cela étant acquis, je me suis consacré à la tâche ingrate mais nécessaire de devenir une personnalité publique et un interlocuteur des médias.

J'avais appris quelques ficelles des médias alors que j'éditais *The New Hacker's Dictionary*. Cette fois-ci, j'ai pris le travail au sérieux, et j'ai développé toute une théorie de manipulation des médias que j'ai ensuite appliquée. Ce n'est pas l'endroit idéal pour l'exposer en détail, mais elle gravite autour de l'utilisation de ce que j'appelle une « dissonance séduisante » pour attiser une curiosité dévorante à l'encontre de l'évangéliste, et exploiter jusqu'au bout cette cette dernière pour faire passer les idées.

La combinaison de l'étiquette « *Open Source* » et de ma promotion délibérée a eu les bonnes et mauvaises conséquences que j'escomptais. Dix mois après l'annonce de *Netscape*, on constate une croissance continue et exponentielle du nombre d'articles dans les médias traitant de Linux et du monde de l'*Open Source* en général. Pendant toute cette période, environ un tiers de ces papiers me citaient directement ; la plupart des deux autres tiers faisaient appel à moi en tant que source indirecte d'informations. Au même moment, une minorité belliqueuse de *hackers* m'ont traité d'intraitable égoïste. J'ai réussi à garder mon sens de l'humour et à plaisanter sur ces deux sujets (même si cela s'est parfois révélé difficile).

Depuis le début, mon plan consistait à confier le rôle de l'évangéliste à un successeur, un individuel ou une organisation. Le temps viendrait où le charisme personnel devrait céder la place à une respectabilité

d'une institution plus largement répandue (et, en ce qui me concerne, au plus vite au mieux !). Au moment où je rédige ces lignes je tente de transférer mon carnet d'adresses personnel et la réputation que je me suis savamment construire auprès de la presse à l'*Open Source Initiative*, une société à but non lucratif fondée dans le seul but de gérer la marque de certification *Open Source*. J'en suis actuellement le président, mais j'espère ne pas le demeurer indéfiniment.

Les phases de la campagne

La campagne de l'*Open Source* a débuté lors de la réunion de Mountain View, et a rapidement mis en place un réseau informel d'alliés connectés par l'*Internet* (y compris des personnalités-clefs de *Netscape* et d'*O'Reilly & Associates*). Quand j'écris « nous », ci-dessous, je me réfère à ce réseau.

Du 3 février au jour où *Netscape* a effectivement publié son code source, le 31 mars, notre souci principal fut de convaincre la communauté des *hackers* que la marque « *Open Source* » et les arguments qui lui étaient associés étaient la meilleure solution pour tenter de convaincre le grand public. Ils se sont révélés plus réceptifs que nous n'imaginions. En réalité, courant était le désir refoulé d'un message moins dogmatique que celui de la *Free Software Foundation*.

Quand la vingtaine de meneurs de la communauté présents au sommet du logiciel libre le 7 mars ont voté et adopté le terme « *Open Source* », ils ont ratifié formellement une tendance qui était déjà claire sur le terrain, parmi les développeurs. Six semaines plus tard, une majorité confortable de la communauté parlait notre langage.

En avril, suite au sommet et à la publication du code source de *Netscape*, notre souci principal fut de recruter autant de parents adoptifs que possible au mouvement de l'« *Open Source* ». Le but était de rompre l'isolement de *Netscape* --- et de nous acheter une assurance au cas où Netscape fasse mauvaise figure et manque ses objectifs.

Ce fut la période la plus éprouvante. Les apparences étaient pourtant encourageantes : techniquement, Linux proposait les fonctionnalités les plus modernes les unes après les autres, le phénomène de l'*Open Source*, plus général, bénéficiait d'une couverture croissante dans la presse informatique, et nous commencions à jouir d'une couverture positive dans la presse grand public. Cependant, j'avais douloureusement conscience du fait que notre réussite était encore fragile. Suite à une débauche initiale de contributions, la participation de la communauté au développement de *Mozilla* a largement souffert de la nécessité de disposer de la bibliothèque *Motif*. Aucun des grands éditeurs indépendants de logiciels ne s'était encore engagé à porter son produit sur la plate-forme GNU/Linux. *Netscape* paraissait encore isolé, et son navigateur continuait à concéder des parts de marché à *Internet Explorer*. Un revers grave ne manquerait pas de faire les choux gras de la presse et de marquer l'opinion publique.

Notre première percée, suite à l'affaire *Netscape*, vint le 7 mars quand la société *Corel* a annoncé qu'elle proposerait un ordinateur pour le réseau , *Netwinder*, fondé sur Linux. Mais cela ne suffisait pas ; pour nourrir la flamme, il nous fallait des engagements, non pas de la part de seconds couteaux désireux de gratter des parts de marché où ils pourraient les trouver, mais de la part de ceux qui mènent la danse dans leur propre branche. Ce sont donc les annonces des sociétés *Oracle* et *Informix*, à la mi-juillet, qui ont mis fin à cette période fragile.

Les pontes des bases de données avaient rejoint le parti de Linux trois mois plus tôt que je ne pensais, mais nous ne nous en sommes pas plaint. Nous nous étions demandés combien de temps pourrait durer l'aura positive de notre mouvement en l'absence d'engagements de la part d'éditeurs indépendants de logiciels (*ÉIL*), et nous devenions nerveux en attendant de telles déclarations. Après les annonces d'*Oracle* et d'*Informix* d'autres *ÉIL* ont annoncé les uns après les autres qu'ils proposeraient une version pour Linux de leurs produits, à tel point que c'en est devenu une routine et qu'on pourrait même survivre à un échec de l'expérience de *Mozilla*.

La phase de consolidation prit place de la mi-juillet à début novembre. C'est à cette époque que nous avons commencé à remarquer une couverture relativement régulière de la part des médias prestigieux que j'avais ciblés à l'origine, dont les têtes de gondole étaient des articles dans *the Economist* et un article

annoncé sur la couverture de *Forbes*. Divers éditeurs de logiciels et fabriquants de matériels ont envoyé des gens prendre le pouls de la communauté de l'*Open Source* et ont commencé à réfléchir à des stratégies pour profiter de ce nouveau modèle. Et de façon interne, le plus grand éditeur de logiciels fermés commençait à se poser sérieusement des questions.

À *quel* point, nous l'apprîmes avec précision quand les « documents Halloween », désormais de sinistre réputation, ont fui de chez *Microsoft*.

Les documents *Halloween* étaient de la dynamite. C'était un témoignage éclatant à la gloire des forces à l'oeuvre dans le développement selon le modèle *Open Source*, de la part de la société qui avait le plus à perdre de la réussite de Linux. Et ils ont confirmé nombreux des soupçons les plus obscurs quand aux tactiques que *Microsoft* emploieraient dans le but d'endiguer ce mouvement.

Les documents *Halloween* ont bénéficié d'une couverture massive dans la presse les premières semaines de novembre. Ils ont provoqué une nouvelle vague d'intérêt pour le phénomène de l'*Open Source*, confirmant de façon fortuite et heureuse toutes les idées que nous tentions de faire passer depuis des mois. Et ils ont directement provoqué une invitation de votre serviteur à une conférence au coeur d'un groupe trié sur le volet des investisseurs les plus importants de Merrill Lynch, sur l'état de l'industrie du logiciel et sur les perspectives de l'*Open Source*.

Wall Street, enfin, nous tendait les bras.

Les faits concrets

Alors que la campagne de l'*Open Source* battait son plein dans les médias, en engageant une guerre virtuelle, les faits techniques et les phénomènes du marché, bien concrets, changeaient eux aussi. J'en passerai brièvement ici quelques-uns en revue car ils forment un tout intéressant avec les tendances de la presse et de la perception du phénomène par le grand public.

Dans les dix mois qui ont suivi la sortie de *Netscape*, Linux a continué d'accumuler les compétences techniques. Le développement d'une proposition solide pour le *SMP* et le nettoyage effectif du code 64 bits ont installé d'importantes fondations pour l'avenir.

La salle de *linuxettes* utilisée pour calculer les scènes d'images de synthèse du film *Titanic* a fait peur aux constructeurs de machines graphiques onéreuses. Puis le projet *Beowulf*, de construire des super-ordinateurs à partir de machines peu coûteuses, a démontré que la sociologie de Linux, fondée sur le modèle des petits ruisseaux, pouvait faire des grandes rivières, même dans le domaine hyper moderne du calcul scientifique.

Rien ne signification n'a projeté les concurrents *Open Source* de Linux sous les feux de la rampe. Et les *Unix* propriétaires ont continué à perdre des parts de marché ; en fait, dès le début du second semestre, seuls les systèmes *NT* et Linux continuaient de rogner des parts de marché au sein des *Fortune 500*, et à sa fin, Linux progressait plus rapidement.

Le logiciel Apache a confirmé son avance dans le marché des serveurs pour le web. En novembre, le navigateur de *Netscape* a renversé sa courbe de parts de marché et a commencé à reprendre du terrain à *Internet Explorer*.

Perspectives d'avenir

J'ai relaté ici les événements récents en partie pour les consigner. De façon plus importante, cela met en place un décor qui peut nous servir à comprendre les tendances à court terme et faire quelques projections pour l'avenir (j'écris ces lignes à la mi-décembre 1998).

Voici tout d'abord quelques prédictions pour l'année prochaine :

- La population des développeurs selon le modèle *Open Source* continuera d'exploser, et cette

croissance sera alimentée par le prix sans cesse plus abordable des compatibles *IBM* PC et des connexions sur l'*Internet*.

- Linux continuera à mener la danser, la taille relative de sa communauté de développeurs compensant les compétences techniques en moyenne plus élevées des gens de l'*Open Source* qui se consacrent aux projets BSD, et de la minuscule équipe travaillant sur le *Hurd*.
- Les *ÉIL* seront de plus en plus nombreux à proposer des solutions pour la plate-forme Linux ; les engagements des vendeurs de bases de données marquèrent un tournant décisif. L'engagement de la société *Corel* de proposer une version complète de leur suite bureautique pour Linux montre la voie.
- La campagne de l'*Open Source* volera de victoire en victoire et fera évoluer les consciences des directions générales, techniques, informatiques, et des investisseurs. Les directions subiront une pression sans cesse croissante d'utiliser des produits issus du monde de l'*Open Source*, non pas de leurs subalternes, mais de leurs supérieurs.
- Les solutions discrètes de serveurs Samba sur plate-forme Linux remplaceront un nombre croissant de machines sous *NT*, même dans des boutiques dont la ligne de conduite est de n'utiliser que des produits *Microsoft*.
- La part de marché des *Unix* propriétaires continuera à s'affaiblir. L'un au moins des concurrents les plus faibles (probablement *DG-UX* ou *HP-UX*) en sera même réduit à déposer le bilan. Mais quand cela se produira, les analystes y verront plus l'oeuvre de Linux que celle de *Microsoft*.
- *Microsoft* ne proposera pas un système d'exploitation prêt pour l'entreprise, car *MS-Windows* 2000 ne sera pas vendu sous une forme exploitable. (Avec 60 millions de lignes de code, ce nombre croissant encore, son développement n'est plus maîtrisable.)

En extrapolant ces tendances on peut risquer quelques prédictions, plus audacieuses, à moyen terme (de 18 à 32 mois d'ici) :

- Le fait de proposer des solutions techniques aux utilisateurs commerciaux de systèmes d'exploitation *Open Source* sera un nouveau secteur d'activité, très rentable, provoqué par, et renforçant, leur utilisation dans le monde des affaires.
- Les systèmes d'exploitation *Open Source* (Linux menant la marche) engloutiront le marché des *FAI*, centres de données. *NT* ne pourra pas résister de manière efficace à ce changement ; le coût réduit, la disponibilité du code source, et une fiabilité sans défaillance, 24h/24 et 7j/7 formeront un tout irrésistible.
- Le secteur de l'*Unix* propriétaire s'écroulera presque entièrement. Il paraît raisonnable de supposer que le système *Solaris* survivra sur le matériel haut de gamme de Sun, mais la plupart des autres systèmes jouant dans la cour du logiciel propriétaire appartiendront bientôt au passé.
- *MS-Windows* 2000 sera soit annulé soit mort-né. Quoi qu'il en soit, ce sera une catastrophe sans nom, le pire désastre stratégique qui soit jamais arrivé à la société *Microsoft*. Cela affectera malgré tout assez peu leur mainmise sur le marché des postes de travail, ces deux prochaines années.

Au premier coup d'oeil, ces tendances ressemblent à une recette pour ne faire survivre que Linux. Mais la vie compliquée (et *Microsoft* tire tant d'argent et de présence sur le marché suite à sa position dominante sur les postes de travail qu'on ne peut pas le laisser pour compte, même après le déraillage de *MS-Windows* 2000.

D'ici deux ans, ma boule de cristal devient plus laiteuse. L'avenir qui nous attend dépend de questions comme : Le ministère de la justice des États-Unis d'Amérique scindera-t-il *Microsoft* ? Le système *BeOS* ou *OS/2* ou *Mac OS/X*, ou un autre créneau de système d'exploitation propriétaire, à moins qu'il ne s'agisse d'une conception radicalement novatrice, trouvera-t-il le chemin de l'*Open Source* et se posera-t-il en concurrent efficace de la conception de Linux, qui a 30 ans d'âge ? Les problèmes liés à l'an 2000 jetteront-ils l'économie mondiale dans une dépression telle que tout le monde pourra jeter ses prévisions ?

Ce sont là des impondérables. Mais une question demeure intéressante : la communauté Linux pourra-t-elle enfin fournir une bonne interface graphique à tout le système ?

Je pense que le scénario le plus probable pour une échéance à deux ans, est que Linux contrôlera les serveurs, les centres de données, les *FAI*, et l'*Internet*, alors que *Microsoft* gardera la mainmise du poste de travail. Ensuite, tout dépendra du fait que *GNOME*, *KDE*, ou une autre interface graphique (ainsi que les applications, construites ou reconstruites pour l'exploiter) devienne suffisamment bonne pour concurrencer

Microsoft sur son propre terrain.

Si cela était avant tout un problème technique, l'issue ne ferait aucun doute. Mais ce n'est pas le cas ; c'est un problème de conception ergonomique et de psychologie de l'interface, et les *hackers* ont historiquement été mauvais à ce petit jeu-là. Ils sont très bons dans la conception d'interfaces réservées à d'autres *hackers*, mais assez mauvais dès qu'il s'agit de modéliser suffisamment bien les processus cognitifs des 95 % restants de la population pour écrire des interfaces que toto et sa tata seraient prêts à acquérir.

Cette année fut l'année des applications ; il est désormais clair qu'on décidera suffisamment d'*ÉIL* pour obtenir celles que nous n'écrirons pas nous-mêmes. Je pense que ces deux prochaines années, le problème sera plutôt de croître suffisamment pour rattraper (et dépasser !) les normes de qualité en matière d'interfaces dictées par la société Macinstosh, et de combiner ces dernières avec les vertus d'un *Unix* traditionnel.

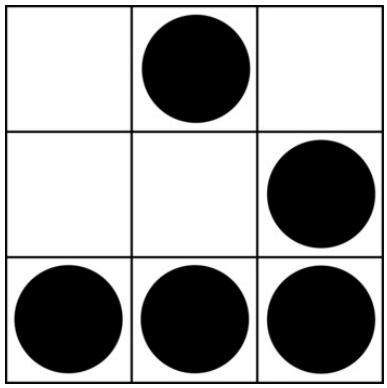
On plaisante à demi à propos de « devenir les maîtres du monde », mais la seule manière d'y parvenir est de rendre service à tout le monde. Il s'agit de plaire à toto et sa tata ; et cela impose de reconsidérer ce que nous faisons d'une manière radicalement différente, et de réduire sans pitié la complexité visible de l'environnement par défaut pour la restreindre à un strict minimum.

Les ordinateurs sont des outils au service des êtres humains. Il faut donc qu'à terme les défis inhérents à la conception de matériel et de logiciel reviennent à concevoir des objets pour les êtres humains - tous les êtres humains.

Ce chemin sera long, et difficile. Mais nous nous devons - et nous devons à autrui - de le suivre jusqu'au bout et de faire les choses correctement. Que l'*Open Source* soit avec toi !

Comment poser les questions de manière intelligente

Auteur : Eric S. Raymond & Rick Moen
Traducteur : Alexandre Courbot
2001 ; traduit en decembre 2001



Traduction en français du document de Eric S. Raymond, dont l'original peut être trouvé ici.
Conforme à la révision 3.4 du 24 mars 2007.

Introduction

Dans le monde des hackers (N.d.T. : Une version traduite de ce document est disponible ici), le type de réponses que vous recevez à vos questions d'ordre technique dépend autant de la manière dont vous formulez la question que de la difficulté à développer la réponse. Ce guide va vous apprendre à poser des questions de telle sorte que vous ayez le plus de chances possible d'obtenir une réponse satisfaisante.

Maintenant que l'usage du logiciel libre (ou *open source*) s'est généralisé, vous pouvez souvent obtenir d'aussi bonnes réponses de la part des utilisateurs expérimentés que des hackers. C'est une bonne chose ; les utilisateurs ont tendance à être légèrement plus tolérants envers les erreurs commises par les débutants. Néanmoins, traiter ces utilisateurs expérimentés de la même manière que les hackers dont nous parlons ici est généralement la meilleure façon d'obtenir de bonnes réponses de leur part.

La première chose à comprendre est que les hackers aiment les problèmes compliqués et les bonnes questions qui font travailler les méninges. Si vous nous donnez une question intéressante en pâture nous vous en serons reconnaissants ; les bonnes questions sont un stimulant et une aubaine. Les bonnes questions nous aident à développer notre propre compréhension, et révèlent souvent des problèmes que nous n'avions pas remarqués et auxquels nous n'aurions pas pensé autrement. Entre hackers, « Bonne question ! » est un compliment fort et sincère.

Malgré cela, les hackers ont la réputation de traiter les questions simples avec hostilité ou arrogance. Parfois, il semble que nous sommes inconditionnellement hostiles aux débutants ou aux ignorants. Mais ce n'est pas tout à fait vrai.

Nous sommes en fait hostiles aux gens qui ont l'air de ne pas avoir réfléchi au problème et n'ont pas fait leur propre travail de recherche avant de poser des questions. De telles personnes sont des gaspilleurs de temps — ils prennent sans donner en retour, ils gaspillent le temps que nous aurions pu passer sur une autre question plus intéressante pour une autre personne qui mérite, elle, une réponse. Nous appelons ces gens-là des "*losers*" (N.d.T. : perdants) (et pour des raisons historiques nous l'écrivons parfois "lusers").

Nous savons bien que beaucoup de personnes veulent juste utiliser les logiciels que nous écrivons, et n'ont aucune envie de s'investir dans les détails techniques. Pour la plupart des gens, un ordinateur n'est rien de plus qu'un outil, un moyen et non pas un but ; ils ont d'autres choses beaucoup plus intéressantes à faire et leur propre vie à mener. Nous sommes tout à fait d'accord avec cela, et nous ne nous attendons pas à ce que tout le monde s'intéresse aux détails techniques dont nous sommes passionnés. Néanmoins, notre style de réponse est adapté aux gens qui sont intéressés et veulent participer activement à la résolution des problèmes. Cela ne risque pas de changer. Et de toute façon il n'y a aucune raison pour que cela arrive ; si cela changeait, nous deviendrions moins efficaces pour les choses que nous savons faire le mieux.

Il faut comprendre que nous sommes (pour la plupart) des volontaires. Nous prenons du temps sur nos vies déjà bien remplies pour répondre à des questions dont nous sommes parfois submergés. Par conséquent, nous n'avons aucun remords à les filtrer. En particulier, nous rejetons les questions de personnes qui ont l'air de *losers* pour passer notre temps de support de manière plus efficace, pour les "*winners*" (N.d.T. : gagnants).

Si vous trouvez cette attitude odieuse, condescendante ou arrogante, repensez-y. Nous ne vous demandons pas de vous mettre à genoux devant nous — en fait, la plupart d'entre nous ne demande pas mieux que de traiter d'égal à égal avec vous et de vous accueillir dans notre culture, si vous faites l'effort nécessaire pour que ce soit possible. Mais il n'est tout simplement pas efficace pour nous d'apporter notre aide à des gens qui ne veulent pas s'aider eux-mêmes. Il n'y a pas de mal à être ignorant ; mais il y en a à faire l'idiot.

Par conséquent, s'il n'est pas nécessaire d'être déjà techniquement compétent pour attirer notre attention, il est en revanche nécessaire d'afficher une attitude susceptible d'amener à cette compétence — être attentif, réfléchi, observateur, consentant à être un partenaire actif au développement de la solution. Si vous ne pouvez pas vivre avec cette sorte de discrimination, nous vous recommandons de payer quelqu'un pour un contrat de support commercial au lieu de demander aux hackers de donner de leur personne pour vous.

Si vous décidez de nous demander de l'aide, vous ne voudrez sans doute pas faire partie des losers. Vous ne voulez sans doute pas non plus avoir l'air d'en être. La meilleure manière d'obtenir une réponse rapide et enthousiaste est de poser votre question comme une personne intelligente, confiante, ayant des indices sur son problème et qui a juste besoin d'un coup de pouce sur un point particulier.

(Les améliorations à ce guide sont les bienvenues. Vous pouvez écrire vos suggestions à esr*(at)*thyrsus*(point)*com ou respond-auto*(at)*linuxmafia*(point)*com. Veuillez cependant noter que ce document n'a pas vocation à être un guide général à la netiquette et que nous rejetons les suggestions qui ne concernent pas l'obtention de réponses utiles sur un forum technique.)

(N.d.T. : Notez bien que les éventuelles erreurs de ce document sont peut-être dûes au traducteur, et non à l'auteur. Si vous avez une amélioration à transmettre, essayez de vérifier si elle est toujours pertinente sur le document original — si ce n'est pas le cas, ou si l'anglais n'est pas votre fort, contactez plutôt le traducteur à gnurou*(at)*gmail*(point)*com. En particulier, les corrections orthographiques et grammaticales sont les bienvenues. Merci aux nombreuses personnes qui ont apporté des corrections, et en particulier à Guillaume Estival pour sa seconde traduction.)

Avant de demander

Avant de poser une question par email, dans les groupes de discussion, ou dans le forum de discussion d'un site web, assurez-vous d'avoir fait les choses suivantes :

- Essayez de trouver une réponse en cherchant sur le Web,
- Essayez de trouver une réponse en lisant le manuel,
- Essayez de trouver une réponse en lisant la FAQ,
- Essayez de trouver une réponse par inspection et expérimentation,
- Essayez de trouver une réponse en demandant à un de vos amis qui s'y connaît,
- Si vous êtes programmeur, essayez de trouver une réponse en lisant le code source.

Quand vous posez votre question, mettez en avant le fait que vous avez déjà fait ces choses ; cela aidera à établir que vous n'êtes pas un pique-assiette qui fait perdre du temps aux autres. Mieux, mettez en avant ce que vous avez *appris* en faisant ces choses. Nous aimons répondre aux questions de ceux qui ont prouvé qu'ils peuvent apprendre à partir de réponses.

Utilisez des tactiques telles que faire une recherche Google sur le texte ou le message d'erreur que vous obtenez (recherchez sur Google groupes en plus des pages Web). Cela risque fort de vous amener directement vers la bonne documentation ou vers un fil de liste de diffusion qui répond à votre question. Et même si ça n'est pas le cas, la phrase « j'ai recherché cette phrase sur Google mais ça ne m'a rien montré d'intéressant » est toujours bonne à inclure dans un email ou un message demandant de l'aide.

Préparez votre question. Pensez-y bien. Les questions précipitées reçoivent des réponses précipitées, voire rien du tout. Plus vous montrez que vous avez fait des efforts pour résoudre votre problème avant de demander de l'aide, plus vous avez de chances d'être aidé.

Faites attention à ne pas poser la mauvaise question. Si vous en posez une basée sur des assertions erronées, le hacker moyen va sûrement vous envoyer une réponse qui vous prendra au mot tout en pensant « Quelle question stupide... », et espérera vous donner une leçon en vous donnant non pas ce dont vous aviez besoin, mais ce que vous aviez demandé.

Ne pensez pas que vous êtes *redevable* d'une réponse. Ce n'est pas le cas ; après tout, vous ne payez rien pour le service rendu. Vous recevrez une réponse, si vous en recevez une, en posant une question qui est riche, intéressante, et qui fait travailler les méninges — une question qui contribue implicitement à l'expérience de la communauté au lieu d'exiger passivement l'aide des autres.

Rendre clair le fait que vous êtes capable et avez la volonté d'aider au développement de la solution est un très bon début. « Quelqu'un peut-il me donner un tuyau ? », « Que manque-t-il à mon exemple ? » et « Y a-t-il un site que j'aurais dû aller voir ? » ont beaucoup plus de chances d'avoir une réponse que « Dites-moi exactement ce que je dois faire, merci. » parce que vous mettez en évidence le fait que vous voulez bien

finir le travail si quelqu'un vous indique simplement la bonne direction.

Quand vous posez votre question

Choisissez bien votre forum

Choisissez l'endroit où vous poserez votre question avec soin. Vous avez toutes les chances d'être ignoré, ou de passer pour un *loser*, si vous :

- posez votre question dans un forum où elle est hors-sujet,
- posez une question élémentaire dans un forum où des questions avancées sont attendues, et vice-versa,
- posez la même question à plein d'endroits différents,
- envoyez un email privé à une personne qui n'est ni une de vos connaissances ni responsable de la résolution de votre problème.

Les hackers rejettent les questions qui ne sont pas bien ciblées de manière à protéger leurs canaux de communication d'un envahissement de messages hors-sujet. Vous ne voulez pas que cela vous arrive non plus.

La première étape est par conséquent de trouver le bon forum. À nouveau, Google et les autres moyens de recherche sur internet sont vos amis. Utilisez-les pour trouver la page Web du projet le plus proche du matériel ou du logiciel qui vous cause des difficultés. Généralement, vous trouverez des liens vers une liste de FAQs (Foire Aux Questions), et vers les listes de diffusion du projet ainsi que leurs archives. Ces listes de diffusion sont les endroits ultimes où rechercher de l'aide, si tant est que vos propres efforts (qui incluent la *lecture* de ces FAQs que vous aurez trouvées) ne vous ont pas sorti d'affaire. La page du projet peut également comporter une procédure pour rapporter les bugs, ou comprendre un lien vers elle. Si c'est le cas, suivez-la.

Balancer un email à une personne ou un forum qui ne vous est pas familier est très risqué. Par exemple, ne vous imaginez pas que l'auteur d'une page informatique ait envie d'être votre consultant gratuit. Ne faites pas de pronostiques optimistes sur l'accueil que recevra votre question — si vous n'être pas sûr, envoyez-la ailleurs, ou retenez-vous de l'envoyer.

Lorsque vous choisissez un forum, un groupe de discussion ou une liste de diffusion, ne faites pas trop confiance à son nom ; vérifiez sur une FAQ que votre question corresponde au sujet de discussion. Lisez quelques uns des messages récents avant d'envoyer votre message histoire d'avoir une idée de comment les choses se déroulent. En fait, c'est une très bonne idée que de faire une recherche avec les mots-clé correspondant à votre problème dans les archives du groupe discussion avant d'envoyer votre message. Vous pourriez trouver ainsi une réponse, ou sinon une meilleure manière de formuler votre question.

Ne tirez pas à vue sur tous les canaux d'aide en même temps, c'est similaire à hurler et irrite les gens.

Connaissez votre sujet de discussion! Une des erreurs les plus classiques est de poser des questions sur l'interface de programmation Unix ou Windows dans un forum dédié à un langage ou à une bibliothèque. Si vous ne voyez pas où se trouve la boulette là-dedans, vous feriez mieux de ne pas poser de questions du tout jusqu'à ce que vous ayez compris.

En général, une question posée sur un bon forum public a plus de chances d'obtenir des réponses utiles que la même question sur un forum privé. Il y a plusieurs raisons à cela. L'une d'entre elles est simplement le nombre de gens susceptibles de vous aider. Une autre est le nombre de personnes composant l'audience ; les hackers ont plutôt tendance à répondre à des questions qui peuvent en apprendre à beaucoup de gens plutôt qu'à des questions utiles pour peu de personnes.

Il est compréhensible que les hackers talentueux et les auteurs de logiciels populaires reçoivent déjà largement leur part de messages mal adressés. En vous ajoutant au flux, vous pourriez dans le cas extrême être la goutte d'eau qui fait déborder le vase — il est déjà arrivé que des contributeurs à des logiciels populaires abandonnent leur support parce que les dommages collatéraux causés par les emails indésirables

sur leurs adresses personnelles sont devenus ingérables.

Les forums Web et IRC destinés aux débutants donnent souvent la réponse la plus rapide

Votre groupe local d'utilisateurs, ou votre distribution Linux, font peut-être la promotion d'un forum Web ou d'un canal IRC permettant aux débutants d'obtenir de l'aide. Ce sont de bons endroits à interroger, en particulier si vous pensez que votre problème est relativement simple à résoudre. Un canal IRC bien promu est une invitation à y poser des questions pour souvent obtenir des réponses en temps réel.

En fait, si le programme qui vous pose problème vient d'une distribution Linux (comme c'est souvent le cas aujourd'hui), il est probablement mieux de poser votre question sur le forum ou la liste de votre distribution avant plutôt que ceux du projet du programme. Les hackers du projet risquent de vous répondre : utilise *notre* binaire.

Avant de poster sur un forum Web, vérifiez s'il dispose d'une fonction de recherche. Si oui, essayez de faire quelques recherches par mot-clé pour quelque chose qui se rapproche de votre problème ; cela pourrait vous aider. Si vous avez déjà fait une recherche Web auparavant (comme vous devriez le faire), cherchez quand même dans le forum ; le moteur de recherche que vous aviez utilisé n'a peut-être pas encore indexé tout le contenu de ce forum.

La tendance actuelle pour les projets est de fournir le support utilisateur au travers d'un forum Web ou d'un canal IRC, l'email étant réservé au trafic lié au développement. Essayez donc de trouver ces canaux en premier lorsque vous cherchez de l'aide par rapport à un projet précis.

En deuxième ressort, utilisez les listes de diffusion du projet concerné

Lorsqu'un projet possède une liste de diffusion dédiée au développement, écrivez sur celle-ci, et non pas à un développeur en particulier, même si vous pensez savoir qu'il répondra mieux à votre question. Recherchez l'adresse d'une liste de diffusion dans la documentation du projet et sur son site web, et utilisez-la. Il y a plusieurs raisons de procéder ainsi :

- Une question qui est assez bonne pour être demandée directement à un développeur aura également de la valeur pour le groupe entier. De la même manière, si vous pensez qu'une question est trop bête pour une liste de diffusion, ce n'est pas une excuse pour importuner un développeur.
- Poser des questions sur une liste répartit la charge entre tous les développeurs. Un développeur particulier (surtout s'il est le leader du projet) sera sans doute trop occupé pour vous répondre.
- La plupart des listes de diffusion sont archivées et les archives référencées par les moteurs de recherche. Quelqu'un pourra par la suite trouver votre question et sa réponse sur le web au lieu de la reposer à nouveau sur la liste.
- Si certaines questions reviennent souvent, les développeurs peuvent se servir de cette information pour améliorer la documentation ou le logiciel lui même afin de les rendre moins confus. Mais si ces questions sont posées en privé, personne ne peut vraiment savoir quelles questions reviennent le plus souvent.

Si le projet possède à la fois une liste de diffusion/forum "utilisateurs" et une autre "développeurs" (ou "hackers"), et que vous ne hackez pas le code vous-même, posez votre question sur la liste "utilisateurs". N'assumez pas que vous serez bienvenu sur la liste développeurs, car ceux-ci risquent d'interpréter votre question comme du bruit qui perturbe le trafic normal de la liste.

Toutefois, si vous êtes *sûr* que votre question n'est pas triviale, et que vous n'obtenez pas de réponse depuis la liste "utilisateurs", essayez de la poser sur la liste "développeurs". Vous seriez bien avisé par ailleurs d'observer le trafic de la liste pendant quelques jours avant d'y poser votre question, afin d'assimiler les moeurs locales (ce conseil est d'ailleurs valable pour toute liste privée ou semi-privée).

Si vous ne pouvez pas trouver de liste de diffusion pour le projet, et ne voyez que l'adresse du mainteneur, adressez vous alors à lui. Mais même dans ce cas, ne supposez pas que la liste de diffusion n'existe pas. Mettez en évidence dans votre email que vous avez cherché une liste de diffusion mais ne l'avez

pas trouvée. Précisez aussi que vous n'avez aucune objection à ce que votre message soit envoyé à d'autres personnes. (Beaucoup de gens pensent que les emails privés doivent rester privés, même s'il n'y a rien de secret dedans. En lui permettant de rediriger votre mail, vous donnez à votre correspondant le choix sur la manière de traiter votre message.)

Utilisez des sujets explicites et adaptés

Sur les listes de diffusion, les newsgroups ou les forums Web, le sujet du message est une occasion en or pour attirer l'attention d'experts qualifiés en 50 caractères ou moins. Ne le gaspillez pas en babillages comme "Aidez-moi SVP" (et ne pensez même pas à "AIDEZ MOI SVP !!!!!!" ; les messages avec ce genre de sujets sont ignorés par réflexe). N'essayez pas de nous faire apitoyer sur votre sort, utilisez plutôt l'espace disponible pour décrire le problème de manière très concise.

Une bonne convention pour les sujets de message, utilisée par beaucoup de supports techniques, est "objet - déviation". "objet" décrit quel composant pose problème, "déviation" explique en quoi le comportement dévie du comportement attendu.

Stupide :

AIDEZ MOI ! La vidéo ne marche pas sur mon portable !

Intelligent :

Curseur de souris difforme sur X.org 4.1, chipset vidéo Fooware MV1005

Encore mieux :

Curseur de souris difforme sur X.org 4.1 avec chipset vidéo Fooware MV1005

Écrire une description du type "objet - déviation" vous aidera à mieux organiser votre pensée à propos du problème. Qu'est-ce qui est affecté? Seulement le curseur de souris ou d'autres éléments graphiques? Est-ce spécifique à X.org? À la version 4.1? Est-ce spécifique aux chipsets vidéos Fooware? Au modèle MV1005? Un hacker qui voit le résultat peut ainsi immédiatement comprendre ce qui vous cause ce problème *et* quel est votre problème, en un coup d'oeil.

Plus généralement, imaginez que vous regardiez l'index d'une archive de questions, dans lequel seuls les lignes de sujet sont visibles. Faites que votre ligne de sujet reflète suffisamment bien votre problème, afin que la prochaine personne qui cherche dans l'archive pour une question similaire à la votre puisse trouver votre question au lieu de la poser à nouveau.

N'appuyez pas sur "répondre" dans un message existant pour commencer un nouveau sujet. Cela va limiter votre audience. Certains lecteurs de mails, comme mutt, permettent de trier les messages par sujet et de grouper les messages d'un même ensemble de réponses en le repliant. Les personnes qui font cela ne verront jamais votre message.

Changer le sujet n'est pas suffisant. Mutt, et probablement d'autres lecteurs de mails, utilisent d'autres informations que le sujet pour assigner un message à un groupe. Commencez plutôt un nouveau sujet.

Sur les forums Web les règles de bonne pratique sont légèrement différentes, parce que les messages sont usuellement attachés à un fil de discussion et sont invisibles en dehors de celui-ci. Changer le sujet lorsqu'on pose une question n'est pas essentiel. Tous les forums ne permettent pas de mettre un sujet sur les réponses, et de toute façon pratiquement personne ne les lit. Toutefois, poser une question en réponse à un autre fil est une pratique douteuse en elle-même, car elle ne sera vue que par ceux qui lisent ce fil. Alors, à moins que vous ne soyez *sûr* de vouloir poser votre question uniquement aux personnes actives dans un fil particulier, commencez-en un nouveau.

Facilitez le travail de ceux qui vous répondent

Terminer votre requête par « Merci d'envoyer la réponse à ... » élimine une bonne partie de vos chances d'obtenir une réponse. Si vous ne prenez pas la peine d'investir dans les quelques secondes nécessaires à préciser un champ "*Répondre à...*" correct dans votre logiciel de courrier, nous n'allons pas prendre la peine d'investir dans les quelques secondes nécessaires à la considération de votre problème. Si votre logiciel de courrier ne vous permet pas de faire cela, installez-en un meilleur . Si votre système d'exploitation ne supporte pas de logiciels de courrier permettant ceci, installez-en un meilleur.

Sur les forums Web, demander une réponse par email est terriblement impoli, à moins que vous ne pensiez que l'information soit sensible (et que quelqu'un pourrait, pour une quelconque raison, vous expliquer quelque chose à vous et pas au reste du forum). Si vous voulez une copie par email des réponses au sujet, demandez au forum de vous l'envoyer ; cette fonctionnalité est supportée pratiquement partout via les options "Surveiller ce sujet", "Recevoir un email quand une réponse arrive", etc.

Écrivez dans un langage clair, faites attention aux fautes de grammaire et d'orthographe

Nous savons par expérience que les gens qui ne font pas attention à la forme de leur écrit ne font en général pas non plus attention à ce qu'ils disent et pensent (du moins, nous l'avons vu assez souvent pour le croire). Répondre aux questions de ceux qui ne font pas attention à ce qu'ils disent n'est pas vraiment valorisant ; nous préférons passer notre temps à faire autre chose.

C'est pourquoi exprimer clairement votre question est important. Si vous ne prenez pas la peine de faire cela, nous ne prendrons pas la peine d'y faire attention. Faites un effort pour travailler votre langage. Cela ne veut pas dire qu'il doit être rigide ou formel — en fait, la culture hacker privilégie plutôt le langage informel, familier et humoristique utilisé avec précision. Mais il *doit* être précis ; il doit y avoir une indication que vous pensez également au problème et y prêtez attention.

Orthographiez correctement, utilisez correctement ponctuation et majuscules. Ne confondez pas "ce" avec "se" ou "c'est" avec "s'est". Ne TAPEZ PAS TOUT EN MAJUSCULES, c'est lu comme si vous criiez et considéré comme impoli (le tout en minuscules n'est qu'un peu moins ennuyeux, car c'est difficile à lire. Alan Cox peut se permettre de le faire, mais pas vous).

Plus généralement, si vous écrivez comme un porc, vous serez probablement ignoré. Ecrire comme un l33t hax0r est le baiser de la mort ultime, et la garantie que vous ne recevrez rien sauf le silence en retour (ou alors, au mieux, on se moquera de vous).

Si vous posez des questions dans un forum qui ne parle pas votre langue natale, faire des erreurs de grammaire ou d'orthographe est excusable, mais ne pas faire d'efforts, non (et, oui, nous pouvons faire la différence). Alors, à moins d'être sûr de la langue natale de vos correspondants, écrivez en anglais. Les hackers ne répondent pas aux questions posées dans une langue qu'ils ne comprennent pas, et l'anglais est la langue la plus courante sur le net. En écrivant en anglais vous maximisez les chances qu'a votre question d'être lue.

Envoyez vos questions dans des formats qui sont facilement lisibles

Si vous rendez vos questions artificiellement dures à lire, elles vont sans doute être ignorées au profit d'autres qui ne le sont pas. Donc :

- Envoyez vos mails en texte simple, et non pas en HTML.
- Les pièces jointes au format MIME sont généralement OK, mais seulement si elles sont du réel contenu (comme un fichier source ou un patch) et pas de la bouillasse générée par votre client mail (comme une deuxième copie de votre message).
- N'envoyez pas de mail dans lequel les paragraphes sont écrits sur une seule ligne dans toute leur longueur (cela rend difficile de répondre à une partie seulement du message). Considérez que vos correspondants lisent leurs messages sur des terminaux texte à 80 caractères et paramétrez le passage à la ligne à 80 caractères ou moins.
- N'utilisez pas de caractères encodés MIME sur des forums anglophones. Cet encodage peut s'avérer

nécessaire si vous postez dans une langue non couverte par le codage ASCII, mais la plupart des clients mail ne le supportent pas - dans ces cas-là votre mail se retrouvera avec plein de caractères de type =20 et sera horrible à lire.
- Ne vous attendez jamais à ce qu'un hacker puisse lire des formats de documents fermés et propriétaires comme Microsoft Word. La plupart des hackers réagissent à cela comme vous réagiriez si on déposait du purin à votre porte.
- Si vous envoyez vos mails à partir d'une machine Windows, désactivez la stupide option "Guillemets intelligents". Cela pour éviter des caractères indésirables dans votre mail.
- Dans les forums web, n'abusez pas des smileys et autres effets HTML. Un smiley ou deux passent bien, mais du texte colorié dans tous le sens vous fera passer pour un blaireau. Abuser des smileys et des couleurs vous fera passer pour une adolescente en fleur, ce qui n'est généralement pas une bonne idée sauf si le sexe vous intéresse plus qu'une réponse.

Si vous utilisez un client mail graphique, comme Netscape Messenger ou Microsoft Outlook, soyez conscient du fait que leurs paramètres par défaut peuvent violer ces règles. La plupart de ces clients disposent d'une commande "Voir le source". Utilisez-la sur un des messages de votre dossier d'envoi, pour vérifier que vos mails sont bien du texte simple, sans autres choses inutiles.

Soyez précis et explicite sur votre problème

- Décrivez les symptômes du problème ou bug consciencieusement et clairement.
- Décrivez l'environnement sur lequel le problème apparaît (machine, OS, application, etc.).
- Précisez votre distribution et sa version (ex : Fedora Core 4, Slackware 9.1, etc.).
- Décrivez les recherches que vous avez effectuées pour comprendre le problème.
- Décrivez les étapes de votre diagnostic et vos tentatives de résoudre le problème.
- Décrivez tout changement récent dans votre ordinateur ou configuration logicielle qui pourrait être en rapport.

Essayez d'anticiper les questions qu'un hacker pourrait vous demander, et d'y répondre en avance dans votre demande d'aide. Simon Tatham a écrit un excellent article intitulé *Comment signaler efficacement un bug*. Je vous recommande grandement de le lire.

Le volume n'est pas la précision

Vous devez être précis et informatif. Coller un gros tas de code ou de données dans votre demande d'aide ne va pas vous y aider. Si vous avez un cas précis de grande taille qui fait planter un programme, essayez de l'élaguer et de le réduire au maximum.

Ceci est utile pour au moins trois raisons. Un : voir que vous faites un effort pour simplifier la question augmentera vos chances d'obtenir une réponse. Deux : simplifier la question augmentera vos chances d'obtenir une réponse utile. Trois : en simplifiant le problème, vous pourriez très bien finir par le résoudre vous même.

Ne prétendez pas avoir trouvé un bug

Quand vous avez des problèmes avec un logiciel, ne prétendez pas avoir trouvé un bug à moins d'être très, *très* sûr de vous. Indice : sauf à être capable de fournir un patch réparant le problème, ou un test de régression sur une ancienne version qui démontre un comportement incorrect, vous n'êtes pas suffisamment sûr de vous. Ceci s'applique également aux pages web et à la documentation ; si vous avez trouvé un "bug" de documentation, vous devez pouvoir fournir un texte de remplacement et la liste des pages auxquelles il s'applique.

Rappelez-vous que beaucoup d'utilisateurs ne rencontrent pas votre problème. Sinon vous en auriez entendu parler en lisant la documentation ou en cherchant sur le Web (car vous l'avez fait avant de vous plaindre, pas vrai?). Cela signifie que c'est très probablement vous qui avez fait quelque chose d'incorrect, et non pas le logiciel.

Les personnes qui ont écrit le logiciel travaillent très dur pour le faire fonctionner aussi bien que possible. Si vous affirmez avoir trouvé un bug, vous mettez en cause leur compétence, ce qui pourrait offenser certains d'entre eux même si vous avez raison. Il n'est particulièrement pas diplomate de mettre "bug" dans le sujet du message.

Quand vous posez votre question, le mieux est d'écrire comme si vous assumiez que **vous** avez fait quelque chose d'incorrect, même si vous êtes secrètement sûr d'avoir trouvé un bug. S'il s'agit vraiment d'un bug, vous le saurez avec la réponse. Jouez-la de telle sorte que les mainteneurs aient envie de s'excuser si le bug est réel, plutôt que ce soit vous qui deviez vous excuser s'il s'avère que vous vous êtes trompé.

Vous écraser ne vous dispense pas de chercher par vous même

Certaines personnes comprennent qu'elles ne doivent pas être arrogantes ou impolies pour obtenir une réponse, et réagissent à l'extrême opposé en se mettant plus bas que terre. "Je sais que je suis un pauvre looser, mais...". Ce comportement est perturbant et n'aide personne. En particulier lorsqu'il est couplé à un problème décrit de façon très vague.

Ne perdez pas votre temps, ni le nôtre, avec de la politique de primates. En lieu et place, présentez les faits que vous avez accumulés et votre question aussi clairement que vous le pouvez. C'est une bien meilleure façon de vous représenter.

Parfois, les forums Web ont des endroits spécifiques pour les débutants. Si vous sentez que vous avez une question de débutant, allez la poser là-bas. Mais ce n'est pas plus une raison de vous y humilier.

Décrivez les symptômes du problème, pas ce que vous devinez

Il n'est pas vraiment utile d'expliquer aux hackers ce qui pourrait causer le problème selon vous (si votre diagnostic était si bon, pourquoi demanderiez-vous de l'aide aux autres ?) Par conséquent, soyez sûr que vous leur indiquez les symptômes bruts de ce qui ne va pas, et non pas vos interprétations et théories. Laissez-les faire le travail d'interprétation et de diagnostic. Si vous pensez que votre interprétation a de l'importance, décrivez-la comme telle et expliquez pourquoi cette solution ne fonctionne pas.

Stupide :

> J'ai des erreurs SIG11 répétées lors de la compilation du noyau, et je pense que ça vient d'une microcoupure sur ma carte mère. Comment puis-je vérifier cela ?

Intelligent :

> Mon K6/233 que j'ai monté moi même sur ma carte mère FIC-PA2007 (chipset VIA Apollo VP2) avec 256Mo de SDRAM PC 133 Corsair me donne des erreurs SIG11 environ 20 minutes après le démarrage pendant que je compile le noyau, mais jamais pendant les 20 premières minutes. Si je reboote, l'horloge ne revient pas à zéro, mais si je l'éteins toute la nuit, si. Changer la RAM ne résout pas le problème. Ci-après, la partie intéressante du log de compilation.

Comme ce point semble être difficile à assimiler pour beaucoup de personnes, rappelez-vous qu'il ne suffit pas d'expliquer : il faut montrer.

Décrivez les symptômes de votre problème dans l'ordre chronologique

Les indices les plus utiles pour trouver ce qui ne va pas se situent souvent dans ce qui s'est produit juste avant. Donc, votre explication devrait décrire précisément ce que vous avez fait, et ce que la machine a fait, pour arriver à la panne. Dans le cas de programmes en ligne de commande, avoir un log de session (c.-à-d. utiliser l'utilitaire script) et citer la vingtaine de lignes intéressantes est très utile.

Si le programme qui a planté possède des options de diagnostic (comme -v pour verbose (N.d.T. : bavard, verbeux)), essayez de sélectionner les options qui vont ajouter des informations de déboguage utiles à

la transcription.

Si votre explication devient trop longue (plus de quatre paragraphes), il serait utile de résumer le problème au-dessus, puis continuer avec le récit chronologique. De cette manière, les hackers sauront ce qu'il faut regarder en lisant votre explication.

Décrivez le but, pas les étapes

Si vous essayez de deviner comment faire quelque chose (par opposition à reporter un bug), commencez par décrire votre but. Ensuite seulement, décrivez l'étape à laquelle vous êtes bloqué.

Souvent, les gens qui demandent de l'aide ont un but particulier en tête et sont bloqués dans ce qu'ils pensent être une étape vers ce but. Ils demandent de l'aide sur cette étape, mais ne réalisent pas que c'est leur approche dans sa totalité qui est fausse. Il est difficile de le deviner dans ces conditions.

Stupide :

> Comment puis-je donner une valeur RGB hexadécimale au programme FooDraw?

Intelligent :

> J'essaie de remplacer la table des couleurs d'une image par des valeurs de mon choix. Pour le moment, le seul moyen que j'ai trouvé est d'éditer chaque valeur individuellement, mais je ne parviens pas à donner une valeur RGB hexadécimale au sélecteur de couleur de FooDraw.

La seconde version de la question est pertinente. Elle permet une réponse qui suggère un outil mieux adapté à la tâche.

Ne demandez pas de réponse privées

Les hackers pensent que les problèmes doivent être résolus en public, de manière transparente, de telle sorte qu'un premier élément de réponse puisse et doive être corrigé si quelqu'un connaissant mieux le problème s'aperçoit qu'il est incomplet ou incorrect. Aussi, leur récompense pour avoir réagi au problème est en partie qu'ils sont reconnus comme compétents et connaisseurs par leurs pairs.

Quand vous demandez une réponse privée, vous cassez ce processus et cette récompense. Ne le faites donc pas. C'est à la personne qui répond de décider s'il faut vous répondre en privé et s'il le fait, c'est en général parce qu'il pense que la question est trop évidente ou mal formée pour être intéressante pour les autres.

Il y a une petite exception à cette règle. Si vous pensez que la question risque d'entraîner beaucoup de réponses qui seront pour la plupart similaires, la formule magique est alors de dire "envoyez-moi vos réponses et je ferai un résumé pour le groupe". Il est courtois d'essayer d'éviter à la liste de diffusion ou au newsgroup d'être inondés par un ensemble de messages similaires - mais vous devez tenir votre promesse de faire un résumé.

Soyez explicite à propos de votre question

Les questions trop générales sont perçues comme une perte de temps. Les personnes les plus à même de vous répondre correctement sont également les plus occupées (entre autres parce qu'elles prennent la plus grosse part du travail). Ces personnes sont allergiques aux pertes de temps, et donc aux questions trop générales.

Vous aurez plus de chances d'obtenir une réponse si vous êtes explicite dans ce que vous voulez que vos correspondants fassent (donner un tuyau, envoyer du code, vérifier un patch, etc.). Cela va leur permettre de concentrer leurs efforts et de mieux vous aider.

Pour comprendre dans quel monde les experts vivent, pensez que l'expertise est une ressource abondante mais que le temps pour répondre manque cruellement. Moins vous demandez de temps, plus vous

avez de chance d'obtenir une réponse de quelqu'un de vraiment bon et vraiment occupé.

Il est donc utile de couper votre question pour minimiser le temps requis pour y répondre -- mais ce n'est pas la même chose que simplifier la question. Par exemple, "Pouvez-vous me donner une adresse vers une bonne explication de X ?" est une bien meilleure question que "Pouvez-vous m'expliquer X ?". Si vous avez du code qui ne marche pas, il est en général plus avisé de demander ce qui ne va pas avec plutôt que de demander de le réparer.

Si vous posez une question à propos d'un programme

Ne demandez pas aux autres de débugger votre code défectueux sans leur donner un indice sur le problème qu'ils doivent chercher. Si vous postez quelques milliers de lignes de code suivies d'un "Ça ne marche pas", vous serez tout simplement ignoré. Mais en postant une petite douzaine de lignes avec un message disant "Je m'attendais à voir <x> après la ligne 7, mais c'est <y> qui est apparu", vous aurez probablement une réponse.

Si vous voulez simplement une révision de votre code, dites-le clairement, et n'oubliez pas de mentionner quelles parties nécessitent selon vous une révision, et pourquoi.

Ne demandez pas de réponse à vos devoirs

Les hackers sont bons pour répondre aux problèmes liés aux devoirs, étant donnés que la plupart les ont faits eux-mêmes. C'est à vous de répondre à ces questions, pour que vous appreniez par expérience. Il est OK de demander des indices, mais pas la solution complète.

Si vous suspectez avoir affaire à ce genre de question, mais ne pouvez pas la résoudre vous-même, essayez de demander à un forum d'utilisateurs ou (en dernier recours) à une liste/forum d'utilisateurs d'un projet. Même si les hackers vont reconnaître la question, certains des utilisateurs avancés pourront au moins vous donner un indice.

Evitez les demandes inutiles

Résistez à la tentation de terminer votre demande d'aide par une question sans intérêt pour la discussion comme "Quelqu'un peut-il m'aider ?" ou "Y a-t-il une réponse à ce problème ?". Premièrement : Si vous avez décrit votre problème de manière correcte, ce genre de phrase est inutile. Deuxièmement : parce qu'elles sont inutiles, les hackers vont les trouver ennuyeuses et risquent de vous envoyer une réponse qui répond à votre question sans vraiment vous aider comme "Oui, quelqu'un peut t'aider" et "Non, il n'y a pas de réponse pour toi".

En général, poser des questions appelant à répondre "oui" ou "non" est une chose à éviter, à moins que ce ne soit la réponse que vous désiriez.

Ne dites pas que votre problème est "urgent", même si c'est le cas

C'est votre problème, pas le nôtre. Prétendre l'urgence sera vraisemblablement contre-productif : les hackers interprètent ces messages comme des tentatives malpolies et égoïstes d'attirer immédiatement l'attention et les effacent.

Il y a une demi-exception. Il peut être utile de préciser que vous utilisez le programme dans un domaine de pointe, dont les hackers seront intéressés ; dans un tel cas, si vous êtes pressé par le temps, et le spécifiez poliment, les gens pourraient être suffisamment intéressés pour vous répondre rapidement.

C'est cependant une chose risquée à faire, car ce qui intéresse les hackers n'est pas forcément ce qui vous intéresse. Poster depuis la Station Spatiale Internationale serait intéressant, par exemple, mais probablement pas poster au nom d'une organisation politique ou de charité pleine de bonnes intentions. En fait, poster un truc du genre : "Urgent! Aidez-moi à sauver les bébés phoques!" ne fera qu'attirer l'ire sur vous, même de la part des hackers qui adorent les bébés phoques.

Si vous trouvez ça bizarre, relisez le reste de ce document jusqu'à ce que vous le compreniez avant de poster quoi que ce soit.

La courtoisie ne fait pas de mal, au contraire

Soyez courtois. Utilisez "S'il vous plaît" et "Merci pour votre attention". Mettez en évidence le fait que vous appréciez le temps que les autres ont passé à vous aider gratuitement.

Pour être honnête, ce n'est pas aussi important (et ne peut en constituer un remplacement) qu'écrire clairement, en évitant les fautes, en étant précis, qu'éviter les formats propriétaires, etc. les hackers en général vont plutôt recevoir des rapports de bugs bruts mais techniquement précis plutôt qu'un flou poli. (Si cela vous étonne, rappelez-vous que nous donnons de la valeur aux questions qui nous apprennent quelque chose.)

Cependant, si la technique n'est pas votre fort, la politesse augmente vos chances d'obtenir une réponse utile.

(Notez que la seule objection sérieuse que nous avons reçue à propos de ce how-to est la recommandation d'utiliser "Merci d'avance". Certains hackers prennent cela comme le fait que vous ne remercierez personne une fois le problème résolu. Notre recommandation est soit de dire "Merci d'avance" en premier *et* de remercier ceux qui vous répondent, ou d'exprimer la courtoisie de manière différente, par exemple en disant "Merci pour votre attention".)

Réagissez à la solution par une petite note

Envoyez une note une fois que le problème est résolu à tous ceux qui vous ont aidé ; faites-leur savoir comment le problème a été résolu et remerciez-les encore pour leur aide. Si le problème a généré de l'intérêt dans la liste de diffusion ou le newsgroup, il est approprié d'envoyer une telle note.

De manière optimale, votre réponse devrait être dans le même fil de discussion que la question initiale, et devrait avoir le mot 'RESOLU' ou tout autre mot aussi explicite dans le sujet. Sur les listes de diffusion à haut trafic, un lecteur voyant un sujet disant "Problème X" se terminant par "Problème X - RESOLU" sait qu'il ne doit pas perdre son temps sur ce sujet (à moins que le problème X ne l'intéresse) et peut donc s'attaquer à un autre problème.

Votre note n'a pas besoin d'être longue et excessivement reconnaissante ; un simple "Ca y est - c'était en fait un câble réseau défectueux ! Merci a tous. - Bill" sera mieux que rien du tout. En fait, un résumé rapide et sympa sera mieux qu'une longue dissertation à moins que la solution n'ait un véritable intérêt technique.

Pour les problèmes ayant une certaine profondeur, il est approprié de poster un résumé de votre progression vers la solution. Décrivez votre problème final. Décrivez la solution qui a fonctionné, et décrivez les éventuels culs-de-sacs *après cela*. Les culs-de-sacs doivent venir après la solution et autres résumés du problème, afin d'éviter de transformer le message en thriller à suspense. Nommez les gens qui vous ont aidés - vous vous ferez des amis.

En plus d'être poli et informatif, ce genre de note permet aux prochaines personnes qui cherchent dans les archives de la liste de diffusion/newsgroup/forum de connaître quelle solution vous a aidé et pourrait potentiellement les aider.

Enfin, ce genre de note permet à ceux qui ont participé de sentir que le problème a bien été résolu. Si vous n'êtes pas technicien ou hacker vous-même, croyez-nous, ce sentiment est très important pour les gourous et experts à qui vous avez demandé de l'aide. Les histoires de problèmes qui finissent par ne jamais être résolus sont des choses frustrantes ; les hackers n'en dorment pas tant qu'ils ne sont pas résolus. Si vous leur évitez cela, cela vous sera très utile la prochaine fois que vous aurez besoin de poser une question.

Essayez de prévoir comment vous pouvez éviter aux autres d'avoir le même problème dans le futur. Demandez-vous si un patch dans la documentation ou la FAQ pourrait aider, et si la réponse est oui envoyez le au mainteneur.

Parmi les hackers, ce genre de bon comportement est plus important que la politesse brute. En vous

comportant ainsi avec les autres, vous vous construirez une bonne réputation, ce qui est un très bon avantage.

Comment interpréter les réponses

RTFM et STFW, ou comment expliquer que vous vous êtes planté

Il existe une tradition ancienne et sacrée : si vous recevez une réponse contenant "RTFM", la personne qui vous a envoyé cela pense que vous devriez "Read The Fucking Manual" (N.d.T. : Lire Le Putain de Manuel). Et elle a certainement raison. Allez-y.

RTFM a un petit frère. Si vous recevez une réponse contenant "STFW", la personne qui vous a envoyé cela pense que vous auriez dû "Search The Fucking Web" (N.d.T. : Chercher Sur le Putain de Web). Et elle a certainement raison. Allez-y. (De manière plus douce, on peut également vous dire "Google Is Your Friend" - Google est ton ami).

Dans les forums Web, on peut également vous inviter à chercher dans les archives. En fait, une personne pourrait être assez gentille pour vous diriger vers le sujet qui répond à votre question. Mais ne comptez pas là-dessus : faites votre propre recherche.

Souvent, la personne qui vous dit de faire une recherche a le manuel ou la page web avec les informations que vous recherchez devant les yeux, et y regarde pendant qu'elle vous répond. Ces réponses veulent dire que cette personne pense que (a) l'information que vous cherchez est excessivement facile à trouver, et (b) que vous allez en apprendre plus si vous cherchez l'information vous-même au lieu de vous la faire apporter sur un plateau.

Il ne faut pas être offensé par cela ; vis-à-vis des autres hackers, il vous montre une certaine forme de respect simplement par le fait qu'il ne vous ignore pas. Vous devriez au contraire le remercier pour sa trop grande gentillesse.

Si vous ne comprenez pas...

Si vous ne comprenez pas la réponse, n'expédiez pas immédiatement une demande de clarification. Utilisez les mêmes outils que ceux que vous avez utilisés pour chercher une réponse à votre problème initial (manuels, FAQ, le Web, les amis) pour comprendre la réponse. Si vous devez demander une clarification, montrez ce que vous avez appris.

Par exemple, supposez que je vous dise : "Je dirais que tu as un zentry qui est coincé ; tu devrais le dégager."

Voici une mauvaise réaction : "*Euh c'est quoi un zentry ?*"

Voici une bonne réaction : "*Ok, j'ai regardé la page man et les zentrys ne sont mentionnés que pour les options -z et -p. Aucune de ces options ne parle de dégager les zentrys. Est-ce que c'est l'un de ceux-là ou alors j'ai raté quelque chose ?*"

Comment réagir face à l'impolitesse

La plupart du temps, ce qui est perçu comme de l'impolitesse dans les propos d'un hacker n'est pas vraiment dit avec l'intention de vous offenser. Il s'agit plutôt du résultat d'un style de communication direct, "ne-perd-pas-ton-temps-en-conneries" qui est naturel pour les gens qui sont plus préoccupés à résoudre les problèmes qu'à être chaleureux avec les autres.

Quand vous sentez que quelqu'un est impoli avec vous, essayez de réagir avec calme. Si quelqu'un dépasse vraiment les bornes, un ancien de la liste, du newsgroup ou du forum va certainement le lui faire savoir. Si cela n'arrive pas et que vous perdez votre calme, alors il est probable que la personne qui vous a mis hors de vous agissait dans les "normes" de la communauté hacker et que la faute retombe sur vous. Cela

risque de compromettre grandement vos chances d'obtenir l'information ou l'aide dont vous avez besoin.

D'un autre côté, vous allez de temps en temps être le témoin d'impolitesse gratuite. L'autre facette de ce qui est dit au-dessus est qu'il est tout à fait acceptable de réprimander violemment les vrais fauteurs de troubles, en les remettant à leur place d'une manière bien ficelée. En tout cas, soyez vraiment, vraiment sûr que c'est le cas avant de passer à l'acte. La limite entre corriger une incivilité et déclencher une flamewar inutile est assez fine pour que les hackers eux-mêmes ne s'y risquent pas trop ; si vous êtes un débutant ou quelqu'un de l'extérieur, vos chances de passer à travers sont relativement faibles. Si vous cherchez l'information plus que l'amusement, il vaut mieux éviter de toucher au clavier et ne pas risquer cela.

(Certaines personnes affirment que beaucoup de hackers agissent comme des autistes et qu'il leur manque la partie du cerveau qui gère les relations humaines. Cela peut être vrai ou faux. Si vous n'êtes pas un hacker vous même, cela peut sans doute vous rassurer de penser que nous sommes des malades mentaux. Allez-y. On s'en fout. Nous aimons être ce que nous sommes, et en général nous réagissons avec un bon scepticisme aux étiquettes qu'on nous colle sur le dos.)

Dans la section suivante, nous allons parler d'un autre problème : le genre d'"impolitesse" que vous verrez lorsque vous vous comportez mal.

Comment ne pas réagir comme un loser

Il est probable que vous alliez vous planter un certain nombre de fois sur les forums de la communauté hacker -- de la même manière que ce qui est décrit dans cet article, ou d'une autre manière. Et on vous dira exactement comment vous vous êtes plantés, parfois même de manière brutale. Et en public.

Quand cela arrive, la pire chose à faire est de vous lamenter de cette expérience, crier sur tous les toits que vous avez été insulté, réclamer des excuses, pleurer, menacer de retenir votre respiration, engager des poursuites judiciaires, vous plaindre aux employeurs, laisser le siège des toilettes relevé, etc. A la place, voici ce que vous pouvez faire :

Laissez cela passer au-dessus de vous. C'est normal. En fait, c'est salutaire et approprié.

Les règles dans les communautés ne se font pas respecter toutes seules : Elles sont maintenues par des gens qui les appliquent à la lettre, de manière visible, en public. Ne vous lamentez pas sur le fait que vous pensez que les critiques auraient dû être envoyées en privé : ça ne fonctionne pas comme ça. Il n'est pas non plus utile d'insister sur le fait que vous avez été personnellement insulté quand quelqu'un dit que l'une de vos affirmations est fausse, ou que son point de vue est différent. Ce sont des attitudes de loser.

Il y a eu des forums de hackers dans lesquels, en raison d'une courtoisie poussée à l'extrême, les participants se faisaient bannir pour avoir simplement critiqué les messages d'autres personnes, et se faisaient dire "Ne dis rien si tu ne veux pas aider l'utilisateur". La fuite des participants-clés qui s'ensuivit a eu pour conséquence de faire descendre ces forums au niveau de babillages et de les rendre inutiles en tant que forums techniques.

Sympathique à l'extrême (dans ce sens) ou utile : Choisissez.

Rappelez-vous : Quand un hacker vous dit que vous vous êtes planté, et (peu importe la manière) vous dit de ne pas recommencer, il agit pour (1) vous et (2) sa communauté. Il serait bien plus simple pour lui de vous ignorer et de vous filtrer de sa vie. Si vous ne pouvez pas en être reconnaissant, au moins ayez un peu de dignité, ne vous lamentez pas, et ne vous attendez pas à être traité comme du sucre parce que vous êtes un nouveau venu avec une âme dramatiquement sensible et des fantasmes de reconnaissance.

Parfois des gens vous attaqueront personnellement, sans raison apparente, même si vous n'avez rien fait de mal (ou fait quelque chose de mal dans leur imagination). Dans ces cas-là, vous plaindre est la meilleure manière d'empirer les choses.

Ces *flamers* sont soit des *lamers* qui ne comprennent rien mais s'imaginent qu'ils sont des experts, ou des psychologues en herbe voulant tester si vous allez vous planter. Les autres lecteurs vont les ignorer, ou s'occuper d'eux à leur propre manière. Le comportement des *flamers* ne crée des problèmes qu'à eux-mêmes,

et vous ne devriez pas vous en mêler.

Ne vous laissez pas entraîner dans une guéguerre non plus. La plupart des attaques sont à ignorer - après que vous ayez vérifié qu'il s'agisse vraiment d'attaques, et que votre question a été posée correctement et a été bien comprise (ce qui peut arriver également).

Les questions à ne pas poser

Voici quelques questions stupides usuelles, et ce à quoi les hackers pensent quand ils n'y répondent pas.

Q : Où puis-je trouver le programme X ?
R : Au même endroit que moi, sot - et sans doute après une bonne recherche sur le Web. C'est pas vrai, il y a encore des gens qui ne savent pas se servir de Google ?

Q : Comment puis-je utiliser X pour faire Y?
R : Si tu veux faire Y, tu devrais poser la question sans présupposer une méthode qui n'est peut-être pas appropriée. Les questions de cette forme indiquent une personne qui n'est pas totalement ignorante de X, mais qui n'a pas bien déterminé le problème Y qu'elle cherche à résoudre et est trop fixée sur les détails d'une situation particulière. Le mieux est d'ignorer ces personnes jusqu'à ce qu'elles définissent mieux leur problème.

Q : Comment puis-je configurer mon invite de commande?
R : Si tu es capable de poser cette question, tu es sans doute capable de faire un petit RTFM pour trouver toi-même la réponse.

Q : Puis-je convertir un document AcmeCorp en fichier TeX en utilisant le convertisseur Bass-o-matic?
R : Tu n'as qu'à essayer. Si tu l'avais fait, tu aurais (a) eu la réponse et (b) évité de gaspiller mon temps.

Q : Mon {programme, configuration, bloc SQL} ne marche pas.
R : Ce n'est pas une question, et je n'ai pas vraiment envie de jouer à Questions/Réponses pour deviner quel est ton vrai problème -- j'ai des choses plus intéressantes à faire.

Quand je vois quelque chose comme cela, ma réaction est en général l'une des suivantes :

- Tu n'as rien d'autre à ajouter à cela ?
- Ah, quel dommage, j'espère que ça va s'arranger.
- Et en quoi cela me concerne-t-il ?

Q : J'ai un problème avec ma machine sous Windows. Vous pouvez m'aider ?
R : Bien sûr. Vire ce déchet de chez Microsoft, et installe Linux.

Note : Vous pouvez poser des questions en rapport avec les machines sous Windows si elles concernent un programme qui a un binaire Windows officiel, ou interagit avec des machines Windows (ex : Samba). Ne soyez cependant pas surpris si la réponse est que le problème vient de Windows, car Windows est tellement mal foutu que c'est souvent le cas.

Q : Mon programme ne marche pas. Je pense qu'il y a un problème avec l'appel système X.
R : Même s'il est techniquement possible que tu sois la première personne à remarquer une déficience évidente dans des appels systèmes utilisés par des centaines de milliers de personnes, il est bien plus probable que tu n'aie rien compris. Des affirmations extraordinaires nécessitent d'être appuyées par des preuves ; si vous faites une telle remarque, vous devez la soutenir avec des informations sur les cas qui mettent la déficience en évidence.

Q : J'ai des problèmes pour installer Linux ou X. Vous pouvez m'aider ?
R : Non, j'aurais besoin d'avoir un accès physique à ta machine pour cela. Va plutôt demander à ton LUG local pour une aide plus proche. (Vous pouvez trouver une liste de groupes d'utilisateurs (LUG) ici).

Note : les questions sur l'installation de Linux peuvent être appropriées si vous êtes sur une forum ou une liste de diffusion d'une distribution particulière, et que ce problème concerne *cette* distribution ; ou sur un forum de groupe d'utilisateurs. Dans ce cas, soyez sûr de décrire les détails du problème. Mais faites une recherche avant, sur "linux" et *toutes* les pièces matérielles suspectes.

Q : Comment est-ce que je peux obtenir les droits root/récupérer les ops sur un channel/lire l'email de quelqu'un ?
R : T'es vraiment désespéré pour vouloir faire de telles choses et un vrai crétin pour demander à un hacker de t'aider.

Bonnes et mauvaises questions

Pour terminer, je vais illustrer comment poser des questions de manière intelligente avec des exemples ; des paires de questions à propos du même problème, l'une posée de manière stupide, l'autre de manière intelligente.

Stupide :

> Où puis-je trouver des infos sur le Foonly Flurbamatic ?

Cette question ne mérite pas plus qu'un STFW comme réponse.

Intelligent :

> J'ai utilisé Google pour chercher "Foonly Flurbamatic 2600" sur le Web, mais je n'ai rien trouvé d'intéressant. Quelqu'un sait où je pourrais trouver des informations utiles pour ce périphérique ?

Cette personne a déjà STFW, et son problème semble réel.

Stupide :

> Je n'arrive pas à compiler le code du projet foo. Pourquoi ne marche-t-il pas ?

Il est persuadé que la faute vient de quelqu'un d'autre. C'est très arrogant de sa part.

Intelligent :

> Le code du projet foo ne compile pas sous Nulix version 6.2. J'ai lu la FAQ, mais il n'y a rien à propos des problèmes avec Nulix. Voici une transcription de ma tentative de compilation ; y a-t-il quelque chose que j'aurais dû faire ?

Il a indiqué son environnement, a lu la FAQ, montré l'erreur, et ne prétend pas que son problème est dû à une erreur de quelqu'un d'autre. Voila quelqu'un qui mérite de l'attention.

Stupide :

> J'ai des problèmes avec ma carte mère. Vous pouvez m'aider ?

La réponse du hacker moyen à cela sera sans doute quelque chose comme "C'est ça, et tu ne veux pas 100 balles et un mars non plus ?" suivi d'une série de caractères de la touche "Suppr".

Intelligent :

> J'ai essayé X, Y et Z sur la carte mère modèle S2464. Cela n'a pas marché, alors j'ai essayé A, B

> et C. Notez bien le symptôme assez curieux quand j'ai essayé C. De toute évidence, le florb a l'air de grommicker, mais les résultats ne sont pas ceux auxquels on pourrait s'attendre. Quelles sont les causes qui amènent la carte mère MP à grommicker ? Quelqu'un a des idées à propos de tests que je pourrais effectuer pour comprendre le problème ?

Cette personne, au contraire, semble mériter une réponse. Elle a montré qu'elle était capable de résoudre un problème par elle même au lieu d'attendre que la réponse tombe du ciel.

Dans la dernière question, notez la différence subtile mais importante entre "Donnez-moi une réponse" et "Merci de m'aider à trouver quels tests je pourrais effectuer pour trouver la réponse".

En fait, la forme de cette dernière question est basée sur un incident réel qui est arrivé en août 2001 sur la liste de diffusion du noyau Linux. J'avais (Eric) posé la question. Une carte mère Tyan S2464 plantait mystérieusement. Les membres de la liste m'ont donné les informations dont j'avais besoin pour résoudre le problème.

En posant la question de la manière dont je l'ai fait, j'ai donné aux gens quelque chose sur lequel ils pouvaient réfléchir ; je les ai encouragés à s'intéresser au problème qui était devenu attractif pour eux. J'ai montré du respect pour l'habileté de mes pairs et les ai invités à me considérer comme l'un d'eux. J'ai également montré du respect pour leur temps en leur expliquant par où j'étais déjà passé.

Une fois le problème résolu, quand j'ai remercié tout le monde et émis la remarque que le processus fonctionnait vraiment bien, un membre de la liste a dit qu'il pensait que cela avait marché non pas parce que je suis connu sur cette liste, mais parce que j'ai posé la question comme il le fallait.

Les hackers vivent dans une impitoyable méritocratie ; je suis sûr qu'il avait raison, et que si je m'étais comporté comme un parasite je me serais fait engueuler ou ignorer, peu importe qui j'étais. Sa suggestion de décrire l'incident au complet en tant qu'aide pour d'autres m'a directement amené à écrire ce guide.

Si vous ne pouvez pas obtenir de réponse

Si vous ne pouvez pas obtenir de réponse, ne le prenez pas personnellement, comme si nous ne voulions pas vous aider. Parfois les membres du groupe auquel vous posez votre question peuvent ne pas connaître la réponse. Pas de réponse ne veut pas dire que vous avez été ignoré, bien que j'avoue qu'il est difficile de faire la différence.

En général, re-poser votre question est une mauvaise idée. Cela sera perçu comme ennuyeux. Soyez patient : la personne avec votre réponse se trouve peut-être dans un autre fuseau horaire ou dort peut-être. Ou peut-être que la question n'était pas bien formulée.

Il y a d'autres sources d'informations vers lesquelles vous pouvez vous diriger, et bien souvent ces sources sont plus adaptées aux besoins d'un débutant.

Il y a beaucoup de groupes d'utilisateurs, en ligne ou localement, qui sont très enthousiastes à propos des logiciels, même s'ils n'ont pas forcément écrit de logiciels eux-mêmes. De tels groupes se forment souvent pour que les gens puissent s'aider les uns les autres et aider les nouveaux utilisateurs.

Il y a également beaucoup de sociétés commerciales auxquelles vous pouvez demander de l'aide, des grandes comme des petites (Red Hat et LinuxCare sont les deux plus connues ; il y en a plein d'autres). Ne soyez pas consterné à l'idée que vous devrez payer pour avoir un peu d'aide ! Après tout, si le moteur de votre voiture coule une bielle, il y a de grandes chances pour que vous deviez l'amener chez un garagiste et payer pour qu'elle soit réparée. Même si le logiciel ne vous a rien coûté, il ne faut pas vous attendre à ce que le support soit toujours gratuit.

Pour les systèmes populaires comme Linux, il y a au moins 10000 utilisateurs par développeur. Il est tout simplement impossible pour une personne de gérer les appels à l'aide de plus de 10000 utilisateurs. Rappelez-vous que même si vous devez payer pour le support, vous devrez toujours payer bien moins cher

que si vous aviez dû acheter le logiciel également (et le support pour les logiciels propriétaires est souvent plus cher et moins compétent que le support pour les logiciels open-source).

Comment répondre aux questions de manière utile

Soyez gentil.

Le stress ajouté aux problèmes peut rendre les gens impolis ou stupides même quand ils ne le sont pas.

Répondez à quelqu'un qui faute pour la première fois hors-ligne.

Il n'est pas nécessaire d'humilier publiquement quelqu'un qui a fait une erreur honnête. Un vrai débutant ne sait peut-être pas chercher dans les archives, ni où se trouve la FAQ.

Si vous n'êtes pas sûr, précisez-le!

Une réponse fausse et assurée est pire que pas de réponse du tout. N'envoyez pas quelqu'un dans la mauvaise direction parce que c'est cool d'avoir l'air de s'y connaître. Soyez humble et honnête ; donnez le bon exemple pour la personne qui pose la question et vos pairs.

Si vous ne pouvez pas aider, n'entravez pas.

Ne faites pas de plaisanteries sur des procédures qui pourraient casser le système de l'utilisateur -- le pauvre gars pourrait les interpréter comme des instructions à suivre.

Posez des questions pour obtenir plus de détails.

Si vous le faites bien, la personne qui demande apprendra quelque chose -- et vous aussi. Essayez de transformer la mauvaise question en bonne question ; rappelez-vous qu'on a tous été un jour débutant.

Même si hurler un RTFM est parfois justifié quand on répond à un fainéant, un pointeur vers la documentation (même s'il s'agit juste de mots-clés Google) est toujours meilleur.

Si vous pouvez répondre à la question, donnez une bonne réponse.

Ne suggérez pas des bidouillages infâmes quand quelqu'un utilise la mauvaise approche ou le mauvais outil. Suggérez d'autres outils. Reformulez la question.

Aidez votre communauté à apprendre de la question.

Quand vous repérez une bonne question, demandez-vous "Comment la documentation ou la FAQ devrait-elle changer pour que personne n'aie à se reposer cette question?". Puis envoyez un patch au mainteneur de la documentation.

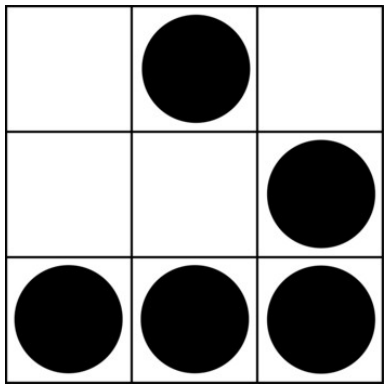
Soyez honnête sur l'origine de votre réponse

Si vous avez fait des recherches pour répondre à la question, démontrez vos talents plutôt que de faire comme si vous aviez sorti la réponse de votre cul. Répondre à une bonne question est comme donner un poisson à quelqu'un, mais leur apprendre à rechercher par l'exemple est comme leur apprendre à pêcher.

Comment Devenir un Hacker

Auteur : Eric S. Raymond
Traducteur : Erwan Monier

2001 ; traduit en juin 2004



Le texte qui suit est une traduction de How To Become A Hacker *de Eric Steven Raymond, version du 2 juin 2004 réalisée par Erwan Monier.*
Le document original se trouve à l'adresse http://catb.org/~esr/faqs/hacker-howto.html.

Pourquoi ce document ?

En tant qu'éditeur du Jargon File et auteur d'autres documents notoires de même nature, je reçois souvent des e-mails d'internautes débutants qui me demandent comment puis-je apprendre à devenir un hacker?. En 1996 j'ai remarqué qu'il ne semblait pas y avoir de FAQs ou de documents sur le Web qui répondaient à cette question vitale et j'en ai commencé un. De nombreuses personnes le considèrent comme une édition définitive, et je suppose que cela veut dire que ça l'est. Toutefois, je ne clame pas avoir une autorité exclusive sur le sujet et si vous n'aimez pas ce que vous lisez ici, rédigez votre propre document.

Si vous lisez une copie d'écran de ce document hors ligne, la version originale se trouve à http://catb.org/~esr/faqs/hacker-howto.html (et la traduction française à http://www.erwanhome.org/web/hacker.php).

Remarque : il y a une Foire Aux Questions à la fin de ce document. S'il vous plaît, lisez ces questions (deux fois) avant de m'envoyer des e-mails à propos de ce document.

De nombreuses traductions de ce document sont disponibles en : Bulgare, Catalan, Chinois (simplifié), Danois, Hollandais, Finnois Allemand, Hébreu, Hongrois, Italien, Japonais, Coréen, Polonais, Portugais, Espagnol, Turque, et Suédois. Comme ce document change occasionnellement, ces traductions peuvent refléter d'anciennes versions.

Le diagramme qui comporte 5 points dans 9 cases et qui décore ce document est nommé un glider. C'est un motif simple possédant des propriétés surprenantes dans une simulation mathématique, appelé Life, qui fascine les hackers depuis de nombreuses années. Je pense que le glider est un bon emblème visuel de ce que les hackers sont : abstraits, semblant à première vue mystérieux mais une passerelle vers un véritable monde ayant sa propre logique. Vous pouvez en lire plus sur le glider [NDT : visitez Le Glider - Emblème Universel des Hackers pour lire la version française].

Qu'est-ce qu'un hacker ?

Le Jargon File contient un grand nombre de définitions du terme hacker, la plupart se rapportant aux aptitudes techniques et au plaisir de résoudre des problèmes et à dépasser les limites. Cependant, si vous souhaitez savoir comment *devenir* un hacker, seules deux définitions sont pertinentes.

Il existe une communauté, une culture commune, d'experts en programmation et de gourous de la gestion de réseau dont les racines remontent à quelques décennies, au temps des mini-ordinateurs à exploitation partagée et des premières tentatives du réseau ARPAnet. Ce sont les membres de cette culture qui ont introduit le terme hacker. Les hackers ont construit Internet. Ils ont fait du système d'exploitation UNIX ce qu'il est aujourd'hui. Ils font tourner le réseau Usenet. Ils font fonctionner le Web. Si vous faites partie de cette culture, si vous y avez contribué et d'autres membres de cette communauté savent qui vous êtes et vous appellent un hacker, alors vous êtes un hacker.

L'état d'esprit d'un hacker ne se limite pas à cette culture informatique. Il y a des personnes qui appliquent cette attitude à d'autres domaines tels que l'électronique ou la musique. En fait, il est possible de la consacrer à l'état le plus avancé de tout art ou science. Les hackers informatiques reconnaissent ces esprits apparentés et peuvent parfois les appeler des hackers. Certains même estiment que cette démarche est totalement indépendante du support sur lequel le hacker travaille. Toutefois, dans la suite de ce document, nous allons concentrer notre attention aux aptitudes et au comportement des hackers informatiques, et aux traditions de cette culture commune qui a donné naissance au terme hacker.

Il y a un autre groupe de personnes qui s'autoproclament des hackers, mais qui n'en sont pas. Ces personnes (majoritairement des adolescents de sexe masculin) qui prennent leur pied à pirater les systèmes téléphoniques et à s'introduire à distance dans les ordinateurs. Les vrais hackers les appellent des crackers et ne veulent avoir rien à faire avec eux. Les vrais hackers pensent surtout que les crackers sont paresseux, irresponsables et pas très brillants. Ils réfutent l'idée qu'être capable de craquer des sécurités fasse de soi un

hacker tout comme démarrer une voiture en court-circuitant l'allumage fasse de soi un ingénieur automobile. Malheureusement, de nombreux journalistes et écrivains se sont laissés abuser à utiliser le mot hacker pour décrire les crackers, ce qui ne lasse pas d'irriter les hackers.

La différence fondamentale est : les hackers construisent des choses, les crackers les détruisent.

Si vous souhaitez être un hacker, continuez de lire ce document. Si vous voulez être un cracker, allez lire le newsgroup alt.2600 et préparez vous à un réveil brutal après avoir découvert que vous êtes moins intelligent que vous le croyez. C'est tout ce que j'ai à dire des crackers.

L'attitude du hacker

Les hackers résolvent des problèmes, construisent des choses et croient en la liberté et en l'entraide bénévole. Pour être accepté comme hacker, vous devez vous comporter comme si vous aviez vous-même cet état d'esprit. Et pour vous comporter comme si vous aviez cet état d'esprit, vous devez vraiment y croire.

Cependant, si vous pensez à adopter des comportements propres aux hackers dans le seul but d'être accepté dans leur culture, vous ne comprenez rien. Croire en ces choses est important parce qu'elles vous permettrons de rester motivé et vous aideront à apprendre. Comme dans toute forme d'art, le moyen le plus efficace de devenir un maître est d'imiter l'état d'esprit des maîtres, non seulement intellectuellement mais aussi émotionnellement.

Et comme dit le poème moderne Zen suivant :

> Pour suivre le chemin :
> Regarde le maître,
> Suis le maître,
> Marche à côté du maître,
> Regarde par les yeux du maître,
> Deviens le maître.

Donc si vous voulez être un hacker, répétez les phrases suivantes jusqu'à y croire réellement :

1. Le monde est rempli de problèmes fascinants qui n'attendent que d'être résolus.

C'est très amusant d'être un hacker, mais c'est le genre d'amusement qui demande beaucoup d'efforts. Et ces efforts nécessitent de la motivation. Les athlètes accomplis tirent leur motivation d'un certain plaisir physique à faire travailler leur corps, à dépasser leurs propres limites. De manière similaire, il faut, pour être un hacker, ressentir une excitation à résoudre des problèmes, à affûter ses compétences et à exercer son intelligence.

Si cette façon de penser ne vous semble pas naturelle, il vous faudra qu'elle le soit pour que vous deveniez un hacker. Autrement vous découvrirez que votre énergie est rongée par des distractions telles que le sexe, l'argent et la reconnaissance sociale.

Vous devez aussi développer une certaine foi en votre propre capacité d'apprentissage, la conviction que, même si vos connaissances ne vous permettent pas de résoudre un problème, le fait d'en attaquer une partie et d'en tirer quelque chose vous permettra de résoudre une autre partie, et cela jusqu'à résoudre le problème entier.

2. On ne devrait jamais avoir à résoudre un problème deux fois.

Les cerveaux créatifs sont une ressource de grande valeur, mais limitée. Ils ne devraient pas perdre leur temps à réinventer la roue alors qu'il existe tant de nouveaux problèmes fascinant qui les attendent.

Pour vous comporter comme un hacker, vous devez être convaincu que le temps de pensée des autres hackers est précieux. Si précieux qu'il est un devoir moral pour vous de partager vos informations, de résoudre des problèmes et d'en donner les solutions pour que d'autres hackers puissent se concentrer sur de

nouveaux problèmes au lieu de travailler perpétuellement sur d'anciens.

(Ne vous croyez pas obligé de diffuser *toutes* vos créations, bien que les hackers les plus respectés sont ceux qui le font. En vendre une partie pour avoir de quoi se nourrir, se loger et payer ses ordinateurs est compatible avec les valeurs des hackers. Vous pouvez aussi utiliser vos compétences pour entretenir votre famille ou même devenir riche tant que vous restez fidèle à votre art et à vos confrères hackers.)

3. L'ennui et les corvées sont maléfiques.

Les hackers (et les gens créatifs en général) ne devraient jamais s'ennuyer ou se consacrer à des taches répétitives, parce que dans ce cas, ils ne font pas ce qu'eux seuls savent faire : résoudre de nouveaux problèmes. Cette perte de temps nuit à tout le monde. Pour ces raisons, l'ennui et les corvées ne sont pas seulement déplaisants, ils sont maléfiques.

Pour vous comporter comme un hacker, vous devez vous en convaincre suffisamment pour vouloir automatiser au maximum les tâches ennuyeuses de votre travail, non seulement pour vous-même, mais aussi pour tous les autres (et particulièrement les autres hackers).

(Il existe toutefois une exception à cette règle. Un hacker va parfois faire des choses qu'un observateur trouverait répétitives ou ennuyeuses dans le but de se vider l'esprit, d'acquérir une nouvelle compétence ou de l'expérience. Mais c'est toujours par choix : toute personne capable de penser ne devrait jamais être forcée d'effectuer un travail ennuyeux.)

4. La liberté est bonne.

Les hackers sont naturellement anti-autoritaires. Quiconque peut vous donner des ordres peut vous empêcher de résoudre les problèmes qui vous fascinent. Étant donné la façon dont les esprits autoritaires fonctionnent, ils trouveront toujours un prétexte stupide pour le faire. Vous devez donc combattre les comportements autoritaires partout où vous les rencontrez, de crainte qu'ils ne vous réduisent au silence, vous et les autres hackers.

(Ce n'est pas pareil que de combattre toute forme d'autorité. Les enfants ont besoins d'être guidés et les criminels d'être contenus. Un hacker peut accepter un certain type d'autorité pour obtenir quelque chose qui vaut bien le temps passé à suivre des ordres. Mais ce doit être une négociation réfléchie et limitée. La soumission totale requise par l'autoritarisme n'est pas une option.)

Les autoritaristes se méfient de l'entraide volontaire et du partage d'informations. Ils n'apprécient la coopération que lorsqu'ils la contrôlent. De ce fait, pour vous comporter en hacker, vous devez devenir instinctivement hostile à la censure, au secret et l'emploi de la force ou de la tromperie pour contraindre des adultes responsables. Et vous devez être prêt à agir conformément à cette conviction.

5. L'attitude ne remplace pas la compétence.

Pour être un hacker, vous devez acquérir certains de ces traits de caractère. Mais simplement adopter un état d'esprit ne fera de vous un hacker, pas plus qu'un champion sportif ou une rock star. Devenir un hacker nécessite de l'intelligence, de l'expérience, du dévouement et un travail acharné.

Par conséquent, vous devez apprendre à vous méfier des attitudes et à respecter les aptitudes, quelles qu'elles soient. Les hackers ne perdront pas leur temps avec les m'as-tu-vu mais ils admirent les compétences, particulièrement liées aux hackers, mais toute compétence est bonne. Elle est particulièrement appréciée lorsqu'elle touche des domaines astreignants que peu de personnes maîtrisent et plus encore des domaines qui nécessitent de l'acuité mentale, du savoir-faire et de la concentration.

Si vous vénérez les aptitudes, vous prendrez plaisir à en développer. Le travail acharné et le dévouement prendront la forme d'un jeu intense et non d'une corvée. Ce trait de caractère est vital pour devenir un hacker.

Les compétences de base du hacker

L'état d'esprit du hacker est primordial, mais ses aptitudes le sont plus encore. Son attitude ne peut remplacer ses compétences et il y a un ensemble de connaissances de base que vous devez posséder avant de pouvoir être considéré comme un hacker par la communauté.

Ces connaissances évoluent lentement au fil des années avec la technologie, certains outils devenant obsolètes, d'autres voyant le jour. Par exemple, la programmation en langage machine était autrefois indispensable alors que le HTML est tout récent. De nos jours, les aptitudes que vous devez posséder sont les suivantes :

1. Apprenez à programmer.

C'est, évidemment, la compétence fondamentale du hacker. Si vous ne connaissez pas de langage de programmation, je vous recommande de débuter avec Python. Il est clairement structuré, bien documenté et relativement simple pour les débutants. Bien qu'il s'agisse d'un bon langage de départ, il ne s'agit pas d'un simple jouet. C'est un outil très puissant, flexible et particulièrement efficace pour les grands projets. J'ai rédigé une évaluation de Python plus détaillée. De bon tutoriels sont disponibles sur le site Web de Python.

Java est aussi un bon langage pour apprendre à programmer. Il est plus difficile que Python mais produit du code plus rapide. Je pense qu'il fait un excellent second langage.

Mais soyez conscient que vous n'obtiendrez pas les aptitudes d'un hacker ou même d'un programmeur en connaissant seulement un ou deux langages. Vous devez apprendre à aborder les problèmes de programmation de manière générale, quelque soit le langage que vous utilisez. Pour être un vrai hacker, vous devez pouvoir apprendre un nouveau langage en quelques jours, simplement en appliquant les règles du manuel à vos connaissances en matière de programmation. Par conséquent il vous faudra maîtriser plusieurs langages très différents.

Si vous vous lancez sérieusement dans la programmation, vous devrez étudier le C, le langage noyau d'Unix. Le langage C++ en est très proche. Ainsi, si vous connaissez l'un, vous n'aurez pas de difficulté à apprendre l'autre. Cependant, il n'est pas bon d'en faire votre premier langage. D'ailleurs, plus vous pouvez éviter de programmer en C, plus vous serez productif.

Le langage C est très efficace et ménage particulièrement les ressources de votre machine. Malheureusement, cet atout nécessite une gestion manuelle conséquente des ressources, telles que la mémoire. Tout ce code de bas niveau est complexe et enclin aux bogues. Vous devrez passer une grande partie de votre temps à le déboguer. De nos jours, avec la puissance des machines, c'est en général un mauvais compromis. Il est plus astucieux de choisir un langage qui demandera moins de votre temps que celui de votre machine. Pour cela, Python est un bon choix.

D'autres langages de programmation particulièrement chers aux hackers comprennent Perl et LISP. Apprendre le Perl est intéressant pour des raisons pratiques. Il est très largement utilisé dans les pages Web actives et l'administration de système. Ainsi, même si vous ne l'employez jamais, vous devriez savoir le lire. Par ailleurs, nombre de personnes s'en servent comme je vous conseille d'utiliser Python, afin d'éviter la programmation en C pour des projets qui ne nécessitent pas l'efficacité des machines C. Vous aurez besoin de comprendre leur code.

L'apprentissage de LISP mérite votre attention car sa maîtrise offre une expérience révélatrice et profonde. Elle fera de vous un meilleur programmeur pour le reste de vos jours, même si vous n'utilisez plus vraiment le langage LISP. (Il est assez facile d'obtenir des bonnes bases avec LISP en codant ou en modifiant des mode d'édition de l'éditeur de texte Emacs.)

Le mieux, en fait, est de les apprendre tous les cinq (Python, Java, C/C++, Perl et LISP). En plus d'être les langages les plus importants pour les hackers, ils représentent des approches très diverses de la programmation, et chacun vous apportera des connaissances très profitables.

Je ne peux pas vous donner ici d'instructions complètes sur l'apprentissage de la programmation, il

s'agit d'une exercice complexe. Mais je peux vous dire que les cours et les livres n'y suffisent pas (de nombreux, et peut-être même *la plupart* des meilleurs hackers sont autodidactes). Vous pouvez apprendre dans les livres certaines fonctionnalités de langage, quelques connaissances, mais l'état d'esprit qui les transformera en réelles compétences ne peut s'acquérir que par des années d'apprentissage et de pratique. Ce qu'il vous faut, c'est (a) *lire du code* et (b) *écrire du code*.

Apprendre à programmer, c'est comme apprendre à écrire naturellement et correctement dans n'importe quelle langue. La meilleure façon d'y arriver, c'est de lire les maîtres, d'écrire soi-même un peu, de lire beaucoup plus, d'en écrire un peu plus, et ce jusqu'à ce que votre écriture montre la même force et simplicité que vos modèles.

Il était autrefois difficile aux apprentis hackers de trouver du bon code à lire et à bricoler, parce que les sources des gros programmes étaient rarement disponibles. Cette situation a vraiment changé. Aujourd'hui, les logiciels à code source ouvert, les outils de programmation et systèmes d'exploitations open-source, tous écris par des hackers, sont très largement accessibles. Ce qui m'amène à notre sujet suivant ...

2. Procurez-vous un Unix à code source ouvert et apprenez à vous en servir.

Je vais supposer que vous possédez ou que avez accès à un ordinateur personnel (de nos jours, c'est si facile :-)). Le tout premier pas pour n'importe quel aspirant hacker pour obtenir des compétences de hacker est d'obtenir une copie de Linux ou d'un des Unix de BSD, de l'installer sur sa machine et la faire tourner.

Certes, il existe d'autres systèmes d'exploitation dans le monde qu'Unix. Le problème est qu'ils sont distribués en code binaire; vous ne pouvez ni le lire ni le modifier. Essayer de bidouiller sur une machine Windows, sous MacOS ou tout autre système binaire, c'est comme tenter d'apprendre à danser alors que vous êtes entièrement plâtré.

Il est possible de bricoler avec OS/X. Toutefois, seule une partie du système est à code source ouvert. Vous risquez donc de rencontrer de nombreux murs, et vous devez rester prudent et ne pas prendre la mauvaise habitude de dépendre d'un code qui est la propriété d'Apple. Si vous vous concentrez sur le système Unix, sous son capot, vous pourrez apprendre quelques astuces très utiles.

Unix est le système d'exploitation d'Internet. Alors que vous pouvez apprendre à vous servir d'Internet sans connaître Unix, vous ne pouvez pas être un hacker d'Internet sans comprendre Unix. C'est pour cette raison que, de nos jours, la culture des hackers est très fortement centrée sur Unix. Ce n'a pas toujours été le cas, et quelques hackers rétro n'en sont toujours pas très contents. Cependant, la symbiose entre Unix et Internet est devenue tellement solide que même la toute puissance de Microsoft n'a pas l'air de l'entailler.

Alors, installez Unix. Personnellement, j'aime Linux, mais il y a d'autres choix possibles (et oui, vous *pouvez* faire tourner Linux et Microsoft Windows sur la même machine). Apprenez-le. Faites-le tourner. Bidouillez avec. Communiquez avec Internet grâce à lui. Lisez le code. Modifiez le code. Vous trouverez de meilleurs outils de programmation (dont C, LISP, Python et Perl) que sous n'importe quel système d'exploitation de Microsoft. Vous vous amuserez et vous assimilerez plus de connaissances que vous ne le croyez, jusqu'à ce que vous réalisiez que vous êtes un maître hacker.

Pour en savoir plus sur Unix, voir The Loginataka. Vous voudrez aussi jeter un coup d'oeil sur l'essai The Art Of Unix Programming.

Pour mettre la main sur une copie de Linux, voir le site Linux Online!. Vous pouvez y effectuer un téléchargement ou, et c'est sans doute une meilleure idée, trouver un groupe local d'utilisateurs de Linux qui vous aidera pour votre installation. Du point de vue d'un nouvel utilisateur, toutes les distributions de Linux sont plus ou moins équivalentes.

Vous pouvez trouver l'aide et les ressources de BSD Unix à www.bsd.org.

J'ai écris un premier jet sur les bases d'Unix et d'Internet.

(Remarque : Je ne conseillerais pas à un novice d'installer Linux ou BSD tout seul. Pour Linux, tachez de trouver un groupe d'utilisateur pas loin de chez vous et demandez-leur de l'aide.)

3. Apprenez à utiliser le World Wide Web et à écrire en HTML.

La plupart des créations de la communauté des hackers fonctionnent dans l'ombre. Elles permettent de faire tourner des usines, des bureaux et des universités sans impact apparent sur les vies des non-hackers.

Le Web est la seule grosse exception, ce jouet des hackers, brillant et démesuré, dont même les *politiciens* admettent qu'il est en train de changer le monde. Rien que pour cette raison, et pour bien d'autres encore, vous devez apprendre à vous servir du Web.

Cela ne signifie pas uniquement savoir utiliser un navigateur (tout le monde pour le faire) mais aussi apprendre à écrire en HTML, le langage de balisage du Web. Si vous ne savez pas programmer, écrire en HTML vous enseignera quelques habitudes mentales qui vous y aideront. Réalisez donc votre page personnelle. Tâchez de vous cantonner au XHTML, qui est un langage plus propre que le HTML classique. (Il existe de bon tutoriels pour débutants sur le Web : en voici un). [NDT : pour un tutoriel en français, visitez Guide HTML / XHTML].

Toutefois, avoir son propre site Web ne fera pas de vous un hacker. Le Web est plein de pages personnelles. Un grand nombre sont sans intérêt, un lisier sans contenu; certes un lisier qui tape à l'oeil, mais pour autant insipide (pour plus de détails à ce propos, visitez The HTML Hell Page). [NDT : Qu'Est-ce qu'une Bonne Page Web est un document similaire mais en français].

Pour qu'elle en vaille la peine, votre page doit posséder du *contenu*. Elle doit être intéressante et/ou utile pour les autres hackers. Ce qui nous amène au sujet suivant...

4. Si vous ne maîtrisez pas l'anglais technique, apprenez-le.

En tant qu'américain et qu'anglophone de naissance, j'ai d'abord eu des réserves quant à suggérer ceci, de peur que ce soit pris comme une sorte d'impérialisme culturel. Pourtant, plusieurs personnes, d'autre langue natale, m'ont exhorté à souligner que l'anglais est la langue pratique d'Internet et de la culture hacker. Vous en aurez donc besoin pour communiquer avec la communauté des hackers.

Tout ceci est véridique. En 1991, j'ai appris que de nombreux hackers, dont l'anglais est la seconde langue, l'utilise dans des discussions techniques, même s'ils ont la même langue natale. On m'a rapporté à l'époque que l'anglais possède un vocabulaire technique plus riche que tout autre langue, ce qui en fait un outil plus adapté. Pour des raisons similaires, les traductions de livres techniques rédigés en anglais laissent souvent à désirer (lorsqu'elles existent).

Linus Torvalds, un finlandais, commente son code en anglais (apparemment, il ne lui est jamais venu à l'esprit de faire autrement). Sa fluidité en anglais fut un facteur primordial dans sa capacité à rassembler une communauté internationale de développeurs pour Linux. C'est un exemple qui vaut d'être suivi.

Les statuts dans la culture hacker

Comme la plupart des cultures sans économie fiduciaire, la réputation fait loi dans la communauté des hackers. Vous essayez de résoudre des problèmes intéressants, mais seuls vos égaux ou vos supérieurs d'un point de vue technique sont à même de juger l'intérêt réel de ces problèmes et la qualité de vos solutions.

Par conséquent, lorsque vous jouez au hacker, vous apprenez à tenir les scores en fonction de ce que les autres hackers pensent de vos aptitudes; et c'est pourquoi vous n'appartiendrez à la communauté hacker qu'un fois reconnu par ses membres. Ce fait est masqué par l'image de solitude de l'activité du hacker, ainsi que par un tabou culturel (en déclin actuellement mais toujours présent) au sein même des hackers, qui refusent que l'orgueil ou la validation extérieure ne fassent partie de leurs motivations.

Explicitement, le monde des hackers est ce que les anthropologues appellent une *culture du don*. Votre statut et votre réputation ne se fondent pas sur votre potentiel à dominer les autres, sur votre beauté ou sur vos possessions qui font des envieux, mais sur votre capacité à donner, en particulier votre temps, votre créativité et le résultat de vos compétences.

Il existe en fait cinq différentes actions que vous pouvez entreprendre pour gagner le respect des

hackers :

1. Écrivez des programmes à code source ouvert.

La première action, la plus essentielle et traditionnelle, est d'écrire des programmes que les autres hackers trouvent amusants ou utiles, et d'en diffuser le code source à toute la communauté.

(Nous avions l'habitude de les appeler des free software [NDT: logiciels libres, en français], mais cela troublait trop de personnes qui n'étaient pas certain du véritable sens de free [NDT: en anglais, le terme free signifie à la fois libre et gratuit]. La plupart des hackers, au moins deux tiers selon des analyses sur le contenu du Web, préfèrent désormais utiliser le terme logiciel open-source [NDT: en français, logiciel à code source ouvert].)

[NDT : Il existe aussi en français deux termes, logiciel libre et logiciel à code source ouvert, même s'il ne semble pas qu'un terme ait pris l'ascendant sur l'autre. Par ailleurs, il existe une réelle différence entre les deux termes anglais. Le Mouvement du logiciel libre, Free Software Foundation, fondé par Richard Stallman, et le Mouvement open source, Open Source Initiative, initié par Eric S. Raymond, coïncident sur la méthodologie des logiciels mais diffèrent principalement sur l'éthique de leurs mouvements, le premier se voulant être un mouvement social.]

Les demi-dieux les plus vénérés sur la planète hacker sont ceux qui ont écrit des programmes importants répondant à des besoins répandus et les ont diffusé gratuitement, si bien que maintenant tout le monde les utilise.

2. Aidez à tester et à déboguer des logiciels à code source ouvert.

Les hackers aident quiconque en a besoin et déboguent les logiciels à code source ouvert. Dans ce monde imparfait, nous passerons fatalement la majorité du temps de développement d'un logiciel à le déboguer. C'est pour cette raison que tous les auteurs de programmes à code source vous diront qu'un bon bêta-testeur (qui sait décrire des symptômes clairement, bien localiser des problèmes, tolérer des bogues dans une mise à jour éclair et qui est prêt à lancer quelques procédures de diagnostic simples) vaut son pesant d'or. Il suffit d'un seul bêta-testeur pour faire la différence entre une phase de déboguage prolongée, épuisante et cauchemardesque et une besogne profitable.

Si vous êtes novice, tâchez de trouver un programme en développement qui vous intéresse et d'en être un bon bêta-testeur. La progression se fait naturellement en commençant par aider à tester les programmes, puis à les déboguer pour finalement participer à leur modification. Vous apprendrez beaucoup en procédant de la sorte, et vous vous ferez de bonnes relations avec des personnes qui vous aideront plus tard.

3. Publiez des informations utiles.

Une autre bonne chose à faire est de réunir et de trier des informations utiles et intéressantes et de les publier sur des pages Web ou des documents tels que des Foires Aux Questions (FAQ) de sorte qu'elles soient disponibles au public.

Ceux qui maintiennent les principales FAQs techniques récoltent presque autant de respect que les auteurs de programmes à code source ouvert.

4. Aidez à maintenir l'infrastructure vivante.

La culture hacker (et donc le développement technique d'Internet) est basée sur le bénévolat. Il existe de nombreuses tâches nécessaires quoique peu séduisantes pour qu'elle se maintienne : administrer des listes de diffusion, modérer des forums de discussion, gérer de grands sites d'archivage de logiciels, développer des RFCs [NDT : Request For Comment, publications de référence portant sur le réseau Internet et rédigées par les experts du réseau, *définition du grand dictionnaire terminologique*] et d'autres normes techniques.

Les personnes qui maintiennent cette infrastructure sont très respectées parce que tout le monde sait

que c'est un travail qui demande beaucoup de temps et qui n'est pas aussi amusant que de jouer avec du code. Y contribuer est une preuve de dévouement.

5. Contribuez à la culture hacker.

Enfin, pour pouvez participer à la transmission de la culture hacker elle-même (par exemple, en rédigeant un livre pour enfants complet sur comment devenir un hacker :-)). Avant d'en arriver là, vous devrez avoir roulé votre bosse dans la communauté et vous être illustré dans l'une des quatre actions ci-dessus.

Le monde des hackers n'a pas de chefs, à proprement parler, mais il possède des héros et des anciens de la tribu, des historiens et des porte-parole. Quand vous aurez passé suffisamment de temps dans les tranchées, vous pourrez peut-être devenir l'un deux. Mais attention, les hackers se méfient de l'ego flagrant des anciens de leur tribu; il est apparemment dangereux d'être en quête de ce statut. Plutôt que de vous y efforcer, vous devez vous mettre sur les bancs et attendre que ce statut vous tombe naturellement dans le creux de la main et rester modeste et abordable.

La relation hacker/nerd

Contrairement au mythe populaire, vous n'avez pas besoin d'être un nerd [NDT : le nerd est un accroc de l'informatique qui passe des nuits entières devant son écran d'ordinateur, n'a pas de vie sociale et ressemble à un adolescent un peu attardé, lunette à double foyer, acné, peu athlétique.] pour être un hacker. Pourtant cela aide, et nombre de hackers sont en réalité des nerds. Être un rejeté social vous permet de rester concentré sur la seule chose qui importe, comme penser et bidouiller comme un hacker.

C'est pourquoi beaucoup de hackers ont adopté l'étiquette de nerd et endossent même avec fierté le badge encore plus rude de geek. C'est une manière de déclarer leur indépendance de toute attente sociale. Pour une discussion plus complète sur le sujet, vous pouvez vous rendre sur The Geek Page.

Si vous arrivez à passer suffisamment de temps pour devenir un bon hacker et continuer à avoir une vie à côté, tant mieux. Cela est plus simple de nos jours que dans les années 70s, quand j'étais un novice. Le courant de pensée de la société actuelle est plus complaisant vis-à-vis des techno-nerds. Il y a même un nombre croissant de personnes qui réalisent que les hackers font souvent d'excellents amants et de très bons maris.

Si vous souhaitez devenir un hacker parce que vous n'avez pas de vie, pas de problème; au moins, vous n'aurez pas de difficulté à vous concentrer. Et puis vous aurez peut-être une vie plus tard.

Remarques sur le style de vie

De nouveau, pour être un hacker, il faut entrer dans l'état d'esprit du hacker. Pour se faire, il existe quelques petites choses que vous pouvez faire lorsque vous n'êtes pas devant votre ordinateur. Ces activités ne remplacent pas le bidouillage informatique mais la plupart des hackers les pratiquent et se sentent fondamentalement connectés à l'essence même de ce qui fait le hacker.

- Maîtrisez l'écrit de votre langue natale. Bien que le stéréotype veuille que le programmeur ne sache pas bien écrire, un nombre étonnant de hackers (parmi lesquels les plus éminents que je connaisse) sont des écrivains compétents.
- Lisez de la science-fiction. Allez aux conventions SF (un bon moyen de rencontrer des hackers et des proto-hackers).
- Étudiez le zen et/ou pratiquez les arts martiaux (la discipline mentale semble remarquablement similaire).
- Développez une oreille musicale aiguisée. Apprenez à apprécier des genres particuliers de musique.

Apprenez à bien jouer d'un instrument ou à bien chanter.

- Stimulez votre goût pour les calembours et les jeux de mots.

Plus vous pratiquez déjà ces activités, plus il est probable que vous soyez à même de devenir un bon hacker. Pourquoi ces disciplines en particulier sont importantes n'est pas tout à fait clair, à part pour le fait qu'elles mettent en jeu un dosage, entre des aptitudes du lobe gauche et droit du cerveau, qui semble important (les hackers doivent savoir à la fois raisonner logiquement et franchir la barrière de la logique apparente d'un problème au moment opportun).

Travaillez aussi intensément que vous jouez et jouez aussi intensément que vous travaillez. Pour les véritables hackers, les frontières entre jeu, travail, science et art ont tendance à disparaître et à fusionner avec un entrain créatif incomparable. Aussi, ne vous limitez pas à une palette de compétences restreinte. Bien que la plupart des hackers se disent être des programmeurs, il y a de fortes chances pour qu'ils soient plus qu'expérimentés dans un grand nombre de domaines liés : administration de système, conception de sites Web, et dépannage de matériel informatique font partie des plus communs. Un hacker qui est administrateur de système, à son tour, sera sans doute compétent en programmation de script et en conception de sites Web. Les hackers ne font rien à moitié. S'ils s'investissent dans une activité, ils deviennent souvent très doués.

Pour conclure, voici une liste de choses à ne pas faire :

- N'utilisez pas de pseudonymes ou de nom d'utilisateur stupide ou grandiloquent.
- Ne prenez pas part aux dissensions frénétiques sur le réseau Usenet (ni où que ce soit).
- Ne vous autoproclamez pas cyberpunk, et ne perdez pas votre temps avec ceux qui le font.
- Ne postez pas de messages et n'envoyez pas d'e-mails remplis de fautes d'orthographe et de grammaire.

La seule réputation que vous vous ferez en agissant de la sorte est celle d'un crétin. Les hackers ont la mémoire longue. Il vous faudra peut-être des années pour que ces erreurs soient oubliées.

Le problème des pseudonymes et des noms d'utilisateur mérite quelques approfondissements. Dissimuler son identité derrière un pseudonyme est un comportement immature et idiot, caractéristique des crackers, des warez d00dz [NDT : une sous-culture majeure des crackers], et d'autres formes de vie inférieures. Les hackers ne le font pas. Ils sont fiers de leurs réalisations et veulent que leur *vrai* nom y soit associé. Donc si vous avez un pseudonyme, laissez-le tomber. Il ne pourra que vous faire passer pour un looser dans la communauté hacker.

Autres ressources

Peter Seebach maintient un excellent Hacker FAQ pour les managers qui ne comprennent pas comment s'y prendre avec les hackers. Si le site de Peter n'est pas en ligne, vous pouvez en trouver une copie sur Excite search.

Il existe un document appelé How To Be A Programmer qui est un excellent complément au précédent. Il contient de précieux conseils non seulement sur la programmation et les compétences requises mais aussi sur comment travailler dans une équipe de programmeurs.

J'ai également écris *Brief History Of Hackerdom* [NDT : Une Brève Histoire des Hackers].

J'ai rédigé l'article *The Cathedral and the Bazaar*, qui explique pas mal de choses sur le fonctionnement de Linux et de la culture open-source. Ce sujet est traité plus précisément dans sa suite : *Homesteading the Noosphere*.

Rick Moen a publié un excellent document sur comment gérer un groupe d'utilisateurs Linux.

Rick Moen et moi-même avons collaboré à la conception d'un autre document : How To Ask Smart Questions. Il vous aidera à demander de l'aide d'une manière qui vous permettra d'en recevoir.

Si vous avez besoin d'instructions de base concernant les ordinateurs personnels, Unix et l'Internet,

allez voir The Unix and Internet Fundamentals HOWTO.

Lorsque vous diffusez un logiciel ou que vous écrivez un patch pour un programme, essayez de suivre les principes de Software Release Practice HOWTO.

Si vous avez aimé le poème zen, vous apprécierez sans doute Rootless Root: The Unix Koans of Master Foo.

Foire Aux Questions

Q: Pouvez-vous m'apprendre à être un hacker ?

R: Depuis que j'ai publié cette page pour la première fois, je reçois chaque semaine des messages (souvent plusieurs par jour) de personnes me demandant apprenez-moi tout ce qu'il faut savoir pour être un hacker. Malheureusement, je n'ai ni le temps ni l'énergie pour le faire. Mes propres bidouillages et mes déplacements en tant que défenseur du mouvement open-source prennent 100% de mon temps.

Et même si j'avais le temps, on ne peut pas enseigner l'attitude et les compétences qui font un hacker, et qu'il faut acquérir par soi-même. Vous découvrirez que les hackers veulent vous aider, mais qu'ils ne vous respecteront pas si vous les implorer de vous nourrir à la petite cuillère.

Commencez par apprendre quelques petits trucs. Montrez que vous faîtes des efforts, que vous êtes capable d'apprendre par vous-même. Après, posez des questions précises aux hackers.

Si vous demandez de l'aide à un hacker par e-mail, voici deux choses que vous devez savoir au préalable. Tout d'abord, nous avons remarqué que les personnes qui sont paresseuses et négligentes dans leur façon d'écrire le sont aussi dans leur manière de penser et ne font pas de bons hackers. Faites donc attention à votre orthographe, grammaire et ponctuation, sinon vous serez sans doute ignoré. Ensuite, n'*envisagez* même pas de demander une réponse sur une adresse e-mail différente de celle que vous avez utilisée pour votre message; en général, ceux qui le demandent sont des truands qui utilisent des comptes piratés, et nous n'avons aucun intérêt à récompenser ou aider le piratage.

Q: Par quoi puis-je commencer, alors ?

R: Le meilleur moyen de démarrer est de vous rendre à une session d'un GUL (groupe d'utilisateur de Linux). Vous pouvez trouver un GUL sur LDP General Linux Information Page [NDT : vous trouverez des GUL francophones sur le site de l'Association Francophone des Utilisateurs de Linux et des Logiciels Libres]. Il y a de fortes chances pour qu'il y en ait un près de chez vous, probablement associé à une université ou un autre établissement d'enseignement supérieur. Ses membres vous donneront vraisemblablement une version de Linux si vous le leur demandez, et vous aideront certainement à l'installer et à débuter.

Q: Quand faut-il commencer ? Est-il trop tard pour moi ?

R: Peu importe votre âge si vous êtes motivé. Beaucoup de hackers semblent s'y intéresser entre 15 et 20 ans, mais je connais des exceptions dans les deux sens.

Q: Comment de temps me faudra-t-il pour devenir un hacker ?

R: Tout dépend si vous êtes doué et si vous vous travaillez sérieusement. La plupart des gens peuvent acquérir un ensemble de compétence convenable en dix-huit mois ou deux ans, s'ils s'investissent. Mais

ne pensez pas que ça s'arrête là. Si vous êtes un vrai hacker, vous passerez le reste de vos jours à apprendre et perfectionner votre art.

Q: Est-ce que le Visual Basic et le C# sont de bons langages pour débuter ?

R: Si vous posez cette question, c'est à coup sûr que vous pensez à bidouiller sous Microsoft Windows. C'est une mauvaise idée en soi. Lorsque je comparais apprendre à bidouiller sous Windows à danser en étant complètement plâtré, je ne plaisantais pas du tout. Ne suivez pas cette route. Elle est répugnante et n'en finit pas de l'être.

Le Visual Basic et le C# présentent des problèmes spécifiques, principalement parce qu'ils ne sont pas transférables. Bien que des prototypes de versions open-source de ces langages existent, les normes de l'ECMA ne couvrent qu'une partie de leurs interfaces de programmation. Sous Windows, la plupart de leurs bibliothèques sont la propriété d'un seul distributeur (Microsoft). Ainsi, si vous n'êtes pas extrêmement prudent sur les caractéristiques que vous utilisez, plus prudent qu'un novice n'en est capable, vous serez réduit aux plateformes que Microsoft a choisi de fournir. Par contre, si vous démarrez sous Unix, de meilleurs langages avec d'excellentes bibliothèques sont disponibles.

Le Visual Basic est particulièrement atroce. Tout comme d'autres versions de Basic, il est très mal conçu et vous donnera de mauvaises habitudes de programmation. Non, *ne* me demandez *pas* de vous les décrire en détails, cela remplirait un livre entier. Apprenez un langage bien conçu à la place.

Une de ces mauvaises habitudes est de devenir dépendant des bibliothèques, des gadgets et des outils de développement d'un seul distributeur. En général, tout langage qui n'est pas disponible sous Linux ou l'un des BSD, et/ou au moins trois systèmes d'exploitation de distributeurs différents, n'est pas un bon langage pour s'exercer à devenir un hacker.

Q: Pourriez-vous m'aider à craquer un système, ou m'apprendre à pirater ?

R: Non. Quiconque pose cette question après avoir lu cette FAQ est trop stupide pour être éduqué, même si j'avais le temps de l'encadrer. Toute requête de ce genre par e-mail sera ignorée ou recevra une réponse particulièrement rude.

Q: Comment puis-je obtenir le mot de passe du compte de quelqu'un d'autre ?

R: C'est du piratage. Fiche le camp, abruti.

Q: Comment puis-je pénétrer/lire/surveiller la boîte e-mail d'un autre utilisateur ?

R: C'est du piratage. Barre-toi, idiot.

Q: Comment puis-je voler les privilèges d'un opérateur de canal IRC.

R: C'est du piratage. Dégage, crétin.

Q: J'ai été piraté. Pourriez-vous m'aider à me défendre contre de nouvelles attaques ?

R: Non. À chaque fois qu'on me pose cette question, c'est un pauvre idiot qui utilise Microsoft Windows. Il n'est pas possible de sécuriser efficacement Windows contre les attaques de pirates. Son code et son architecture possèdent tout simplement trop de défauts. Sécuriser Windows serait comme vouloir

renflouer un bateau percé comme une passoire. Le seul moyen de se prévenir contre toute attaque est d'opter pour Linux ou un autre système qui a été conçu pour, au moins, pouvoir être sécurisé.

Q: J'ai des problèmes avec mes logiciels Windows. Pourriez-vous m'aider ?

R: Oui. Ouvrez une fenêtre de commande DOS et tapez format c:. Tout problème que vous rencontrez devrait cesser d'ici quelques minutes.

Q: Où puis-je trouver de vrais hackers à qui parler ?

R: Le meilleur moyen est de trouver un groupe d'utilisateurs d'Unix ou de Linux près de chez vous et d'aller à leurs réunions (vous trouverez des liens vers plusieurs listes de groupes d'utilisateurs sur le site LDP hébergé par ibiblio).

Q: Pourriez-vous me conseiller des ouvrages consacrés au monde des hackers ?

R: Je maintiens The Linux Reading List HOWTO [NDT : Bibliographie Linux] qui peut vous servir. Vous pourrez aussi trouver le Loginataka intéressant.

Pour une introduction à Python, allez voir les documents introductifs sur le site de Python.

Q: Est-ce que j'ai besoin d'être fort en mathématiques pour devenir un hacker ?

R: Non. Bien que vous deviez être à même de réfléchir logiquement, vous n'aurez besoin que de très peu d'arithmétique et de mathématiques formelles.

En particulier, vous n'aurez pas besoin de trigonométrie, de calcul ou de mathématiques analytiques (nous laissons tout ça aux électroniciens :-)). Certaines bases en mathématiques discrètes (ce qui comprend l'algèbre booléenne, la théorie des ensembles finis, l'analyse combinatoire et la théorie des graphes) peuvent s'avérer utiles.

Q: Quel langage dois-je apprendre pour commencer ?

R: Le XHTML (le dialecte le plus récent de HTML), si vous ne le connaissez pas déjà. Il existe malheureusement peu de bons ouvrages sur le HTML et un grand nombre de très beaux mais très *mauvais* livres dont la publicité est extravagante et intensive. Celui que je préfère est HTML: The Definitive Guide.

Mais le HTML n'est pas un langage de programmation complet. Lorsque vous serez prêt à programmer, je vous conseille de commencer avec Python. Vous entendrez beaucoup de personnes préconiser Perl, et Perl est toujours plus populaire que Python, mais il est plus dur à apprendre et n'a pas été, à mon avis, aussi bien conçu.

Le C est très important, mais il est aussi plus difficile que Python ou Perl. N'essayez pas de l'apprendre en premier.

Quant aux utilisateurs de Windows, *n*'optez *pas* pour le Visual Basic. Il vous donnera de mauvaises habitudes et il n'est pas transférable hors de Windows. À éviter.

Q: De quel type de matériel ai-je besoin ?

R: Autrefois, je pensais que les ordinateurs personnels n'ont pas assez de mémoire et ne possèdent pas de moteurs assez puissants, de sorte qu'ils limitent artificiellement l'apprentissage du hacker. Cela a cessé d'être vrai il y a quelques années. N'importe quelle machine supérieure à un Intel 486DX50 est plus que suffisamment puissante pour faire du développement, faire fonctionner un système X Window et utiliser Internet. Par ailleurs, les plus petits disques durs que vous pouvez acheter de nos jours font amplement l'affaire.

L'étape la plus importante dans le choix d'une machine est de vérifier que le matériel est compatible avec Linux (ou BSD, si vous décidez de suivre ce chemin). Encore une fois, la plupart des ordinateurs personnels modernes devraient l'être. Le seul domaine vraiment poisseux est celui des modems. Certaines machines ont du matériel spécifique Windows qui ne marche pas sous Linux.

Il existe une FAQ concernant la compatibilité du matériel informatique, dont voici la dernière version.

Q: Je souhaite contribuer. Pourriez-vous m'aider à choisir un projet sur lequel travailler ?

R: Non, parce que je ne connais ni vos compétences ni vos intérêts. Vous devez être motivé sinon vous n'irez pas au bout. C'est pour cela que ne pas choisir soi-même son projet ou la direction de son travail ne marche jamais ou presque.

Essayez plutôt ceci : regardez pendant quelques jours la liste des projets annoncés sur Freshmeat. Lorsque vous en voyez un qui vous fait penser Cool! J'aimerais bien bosser là-dessus!, rejoignez le projet.

Q: Dois-je haïr et cogner sur Microsoft ?

R: Non, vous n'êtes pas obligé. Non pas que Microsoft n'est pas écoeurant, mais il y avait une culture hacker bien avant Microsoft et il y en aura une bien après que Microsoft aura été de l'histoire ancienne. Toute énergie dépensée à haïr Microsoft serait mieux employée par vos compétences et votre passion. Écrivez du bon code, cela brisera suffisamment Microsoft sans polluer votre karma.

Q: Les logiciels à code source ouvert ne vont-ils pas empêcher les programmeurs de gagner leur pain ?

R: Cela paraît peu probable. Jusqu'à présent, l'industrie des logiciels à code source ouvert semble créer plus d'emplois qu'elle n'en détruit. Dès lors que la création d'un programme représente un profit net sur son inexistence, un programmeur sera payé, que le code source du logiciel soit ouvert ou non une fois écrit. Par ailleurs, quelque soit le nombre de logiciels libres écrit, il y aura toujours une demande pour de nouvelles applications et des programmes personnalisés. J'en ai écrit plus à ce sujet dans le site Web Open Source.

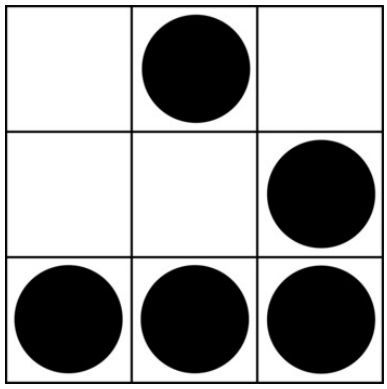
Q: Comment puis-je démarrer ? Où puis-je trouver un Unix gratuit ?

R: Il existe ailleurs sur cette page des liens vers les sites Web où vous pourrez obtenir les Unix gratuits les plus couramment utilisés. Pour devenir un hacker, il vous faut de la motivation, de l'initiative et la capacité d'être autodidacte. Commencez maintenant ...

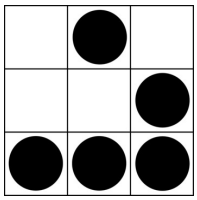
Le Glider - Emblème Universel des Hackers

Auteur : Eric S. Raymond
Traducteur : Erwan Monier

2001 ; traduit en juin 2004



Le texte qui suit est une traduction de The Glider: A Universal Hacker Emblem *de Eric Steven Raymond, version du 23 octobre 2005 réalisée par Erwan Monier.*
Le document original se trouve à l'adresse http://www.catb.org/~esr/hacker-emblem/.



Pourquoi cet emblème ?

Les gars de chez Linux ont leur pingouin et les adeptes des différents BSD ont leur démon. Perl a un chameau, la Fondation pour le logiciel libre (FSF) a son gnou, et l'Initiative Open-Source (OSI) a un logo open-source. Ce que nous n'avons jamais eu, c'est un emblème qui représente l'*entière* communauté des hackers, dont tous ces groupes font partie. Je propose que l'on adopte un emblème - le glider [NDT : en français, planeur], une des structures du Jeu de la Vie.

Près de la moitié des hackers sur lesquels j'ai testé cette idée se sont instantanément écriés «Wow! Cool!» sans demander de plus amples explications. Si vous ne savez pas ce qu'est un glider, ou pourquoi il ferait un bon emblème, ou vous si vous êtes tout simplement sceptique à l'idée d'un emblème, je vous invite à lire la Foire Aux Questions à Propos du Glider.

J'ai proposé cet emblème pour la première fois en octobre 2003. Depuis, son utilisation est relativement répandue, comme vous pouvez le constater par le nombre de traductions internationales (allemand, arabe, espagnol, hongrois, italien, polonais, portugais et serbe). Certes, cet emblème n'est pas universel, du fait que de nombreux hackers refusent par principe la simple idée d'un emblème, mais il semble que ce soit un concept couronné de succès.

Quel est le message derrière cet emblème ?

Lorsque vous affichez le glider sur votre page Web, vos habits, ou quelque autre support, vous vous associez à la culture des hackers. Ce n'est pas pareil que de se revendiquer un hacker. En général, on ne s'autoproclame pas hacker, c'est un titre d'honneur qu'on se voit décerner par ses confrères. En utilisant cet emblème, vous témoignez de votre solidarité envers les hackers, leurs objectifs, leur éthique et leur façon de vivre. Vous pouvez lire la Foire Aux Questions pour plus d'explications.

C'est vrai, à peine quatre jours après avoir proposé cet emblème, il existait déjà des mugs et des T-shirts. Je tiens à préciser que je n'ai rien à voir avec ces produits et que je ne reçois aucune royalties. La recette est en fait reversée à l'organisation Electronic Frontier Foundation. Il existe aussi un autre site Web avec d'autres motifs.

Qui *ne* doit *pas* l'utiliser ?

Si vous pensez que le hacking c'est s'introduire sur des ordinateurs à l'insu de leurs propriétaires, ce logo ne vous est pas destiné. Ceux pour qui cet emblème a été créé ne veulent pas que vous l'utilisiez. Inventez votre propre emblème, bande de crackers. Nous trouverons le moyen de vous discréditer et répudier publiquement si vous souillez le nôtre.

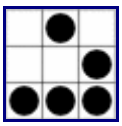
J'avais d'abord interdit l'usage commercial de cet emblème mais de nombreuses personnes m'ont convaincu que c'est impossible à mettre en œuvre et quelque peu arbitraire. Donc, si vous souhaitez utiliser le glider dans un but commercial, que cela reste élégant, ou bien vous vous ferez incendier.

Comment puis-je l'utiliser ?

Le glider n'est pas une marque de commerce et n'est pas protégé par des droits d'auteur. Il est recommandé de l'utiliser sur une page Web, avec une image et un lien vers cette page ou bien directement vers How To Become A Hacker [NDT : vous pouviez aussi faire un lien vers la version française Comment Devenir un Hacker, si vous le souhaitez]. Voici un bout de code en XHTML que vous pouvez copier sur dans un document HTML :

```
<a href='http://www.catb.org/hacker-emblem/'>
<img src='http://www.catb.org/hacker-emblem/glider.png' alt='hacker emblem' /></a>
```

Voilà ce que cela donne :



Vous êtes libre d'agrandir ou de réduire l'image à votre guise. Le fichier PNG a été généré à partir d'une source PIC, puis réduit de moitié. Vous pouvez aussi télécharger une version SVG, une version PostScript encapsulé (EPS) et même un code source TEX.

Variantes

Avant de créer votre propre variante, je vous prie de lire la Foire Aux Questions. Voici quelques versions qu'on m'a envoyéees :

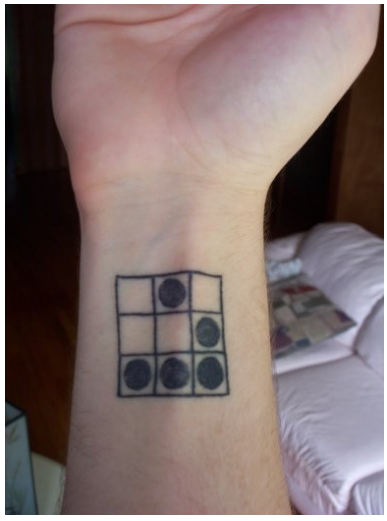
```
. O .
. . O
O O O
```

```
|_|0|_|
|_|_|0|
|0|0|0|
```

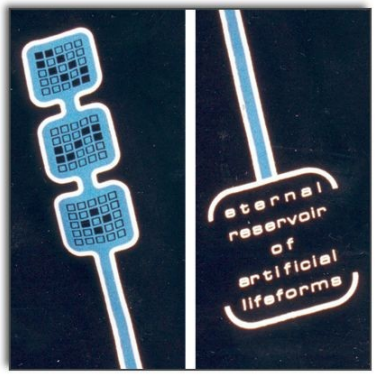
Voici un thème karamba pour afficher l'emblème des hackers sur son bureau KDE.

Voici une version ico, et en voilà une autre. Si vous nommez l'un des ces fichiers favicon.ico et le copiez dans le répertoire racine de votre site, il deviendra l'icône de votre site Web.

Ce tatouage est impressionnant, mais peut-être un peu excessif.



L'idée d'utiliser les structures du Jeu de la Vie comme emblème fut inspirée par des hackers en Argentine.



Foire Aux Questions

Que représente cet emblème?

Le graphique en haut de la page s'appelle un Glider. Il s'agit d'un modèle tiré d'une simulation mathématique appelé le Jeu de la Vie. Dans cette simulation, des règles très simples induisant le comportement de points sur une grille va donner lieu à des phénomènes émergents merveilleusement complexes. Le Glider est le plus simple de ces phénomènes, et le plus immédiatement reconnaissable de tous les modes de vie.

Pourquoi adopter un emblème?

Pour certains hackers, le fait d'avoir un emblème semble trop relever d'un mode de pensée unique. Mais la communauté hacker est réellement une communauté, soudée par des liens de confiance sur Internet. Une chose que nous avons apprise depuis 1991 est que les emblèmes de la communauté sont tout aussi précieux pour les pirates qu'ils le sont à d'autres communautés humaines. Ils nous aident à nous reconnaître les uns les autres, à affirmer des valeurs communes et à coopérer plus étroitement. Ils sont utiles, socialement parlant.

L'utilisation de cet emblème signifie quelque chose d'un peu différent que de se présenter simplement comme un fan de Linux, ou d'un groupement Perl, ou comme membre d'une quelconque tribu de hackers qui ont connues tant de succès depuis le milieu des années 1990. Ceux-ci sont le développement relativement récent d'une tradition qui remonte à des décennies. Qui remonte aux expérimentateurs de micro-ordinateur au début des années 1970, aux premiers développeurs et ingénieurs Unix ARPANET en 1969, et à la SPACEWAR hackers du MIT en 1961.

Les hackers, au sens le plus large, sont les gens qui ont construit l'Internet, Unix, et le World Wide Web; nos rêves de liberté ont changé le monde. Voyez « Comment devenir un hacker » pour une approche approfondie de ce que cela signifie. Si vous vous trouvez en accord avec ce document, vous faites partie des gens qui devraient utiliser cet emblème.

Quel est cet emblème?

Le Glider est un emblème à plusieurs niveaux. Début de l'histoire: le jeu de la vie a tout d'abord été publiquement décrit dans le Scientific American, en 1970. Il est né à peu près au même moment que l'Internet et Unix. Il a fasciné les hackers depuis.

Dans le Jeu de la Vie, des règles simples créent l'inattendu, provoquent des modèles complexes, qui

ne sont pas liés à des schémas connus de phénomènes pré-établis. Il s'agit d'un joli parallèle sur l'inattendu et surprenant phénomène du développement de la communauté Open Source, et de l'émergente communauté des hackers.

Le Glider remplit tous les critères d'un bon logo. Il est simple, visible, difficile à confondre avec n'importe quoi d'autre, et facile à imprimer sur une tasse ou T-shirt. Il pourrait être modifié, combiné à d'autres emblèmes, ou répété à l'infini pour être utilisé comme arrière-plan.

Mais que faire si les mauvaises personnes commencent à l'utiliser?

Beaucoup de personnes pensent que cet emblème deviendra pire qu'inutile parce que script kiddies, crashers et lamers seront les principaux à l'utiliser, car majoritaires. Oui, c'est un risque, mais d'autres emblèmes, comme le signe de paix ou le A-pour-anarchie, qui ont subi des risques similaires, ont conservé leurs valeurs. Si cela peut aider, j'ai reçu beaucoup de messages de gens que je sais appartenir au « noyau dur » des pirates, et je n'ai vu presque aucun abus.

Allez-vous soutenir ma version modifiée de l'emblème?

Probablement pas. L'objectif d'un tel logo est la reconnaissance immédiate. Cet objectif ne peut être atteint que si les variantes sont simples. Les deux propositions les plus fréquentes ont été la suppression du quadrillage et le changement de direction du Glider. Peut-être que ces variantes seraient préférables, ou une sophistication de ma proposition. Cependant, le logo tel qu'il est a certainement un avantage important de par sa simplicité. Par conséquent, je vais réunir uniquement des images sur lesquels le logo normal, comme indiqué plus haut sur cette page, est clairement reconnaissable.

Pourquoi vous?

Parce que j'entretiens les documents « Comment devenir un hacker », « A Brief History of Hackerdom », « le Jargon File », et que je suis plus ou moins l'historien/anthropologue de la communauté des hackers. C'est mon travail de penser à ces choses, si tant est que ce soit le travail de quelqu'un.

Annexe A - Licence de documentation libre GNU

Licence de documentation libre GNU

Disclaimer

This is an unofficial translation of the GNU Free Documentation License into French. It was not published by the Free Software Foundation, and does not legally state the distribution terms for documentation that uses the GNU FDL--only the original English text of the GNU FDL does that. However, we hope that this translation will help French speakers understand the GNU FDL better.

Ceci est une traduction française non officielle de la Licence de documentation libre GNU. Elle n'a pas été publiée par la Free Software Foundation, et ne fixe pas légalement les conditions de redistribution des documents qui l'utilisent -- seul le texte original en anglais le fait. Nous espérons toutefois que cette traduction aidera les francophones à mieux comprendre la FDL GNU.

Traduction française *non officielle* de la GFDL Version 1.1 (Mars 2000)

Pour la traduction :
Version 1.0 FR (Jean-Luc Fortin, juillet 2000)
Version 1.1 FR (Christian Casteyde, mars 2001)
Version 1.1.1 FR (César Alexanian, mars 2001)
Version 1.1.2r2 FR (Christian Casteyde et César Alexanian, juin 2001)

0 - PRÉAMBULE

L'objet de cette Licence est de rendre tout manuel, livre ou autre document écrit « libre » au sens de la liberté d'utilisation, à savoir : assurer à chacun la liberté effective de le copier ou de le redistribuer, avec ou sans modifications, commercialement ou non. En outre, cette Licence garantit à l'auteur et à l'éditeur la reconnaissance de leur travail, sans qu'ils soient pour autant considérés comme responsables des modifications réalisées par des tiers.

Cette Licence est une sorte de « copyleft », ce qui signifie que les travaux dérivés du document d'origine sont eux-mêmes « libres » selon les mêmes termes. Elle complète la Licence Publique Générale GNU, qui est également une Licence copyleft, conçue pour les logiciels libres.

Nous avons conçu cette Licence pour la documentation des logiciels libres, car les logiciels libres ont besoin d'une documentation elle-même libre : un logiciel libre doit être accompagné d'un manuel garantissant les mêmes libertés que celles accordées par le logiciel lui-même. Mais cette Licence n'est pas limitée aux seuls manuels des logiciels ; elle peut être utilisée pour tous les documents écrits, sans distinction particulière relative au sujet traité ou au mode de publication. Nous recommandons l'usage de cette Licence principalement pour les travaux destinés à des fins d'enseignement ou devant servir de documents de référence.

1 - APPLICABILITÉ ET DÉFINITIONS

Cette Licence couvre tout manuel ou tout autre travail écrit contenant une notice de copyright autorisant la redistribution selon les termes de cette Licence. Le mot « **Document** » se réfère ci-après à un tel manuel ou

travail. Toute personne en est par définition concessionnaire et est référencée ci-après par le terme « **Vous** ».

Une « **Version modifiée** » du Document désigne tout travail en contenant la totalité ou seulement une portion de celui-ci, copiée mot pour mot, modifiée et/ou traduite dans une autre langue.

Une « **Section secondaire** » désigne une annexe au Document, ou toute information indiquant les rapports entre l'auteur ou l'éditeur et le sujet (ou tout autre sujet connexe) du document, sans toutefois être en rapport direct avec le sujet lui-même (par exemple, si le Document est un manuel de mathématiques, une **Section secondaire** ne traitera d'aucune notion mathématique). Cette section peut contenir des informations relatives à l'historique du Document, des sources documentaires, des dispositions légales, commerciales, philosophiques, ou des positions éthiques ou politiques susceptibles de concerner le sujet traité.

Les « **Sections inaltérables** » sont des sections secondaires considérées comme ne pouvant être modifiées et citées comme telles dans la notice légale qui place le Document sous cette Licence.

Les « **Textes de couverture** » sont les textes courts situés sur les pages de couverture avant et arrière du Document, et cités comme tels dans la mention légale de ce Document.

Le terme « **Copie transparente** » désigne une version numérique du Document représentée dans un format dont les spécifications sont publiquement disponibles et dont le contenu peut être visualisé et édité directement et immédiatement par un éditeur de texte quelconque, ou (pour les images composées de pixels) par un programme de traitement d'images quelconque, ou (pour les dessins) par un éditeur de dessins courant. Ce format doit pouvoir être accepté directement ou être convertible facilement dans des formats utilisables directement par des logiciels de formatage de texte. Une copie publiée dans un quelconque format numérique ouvert mais dont la structure a été conçue dans le but exprès de prévenir les modifications ultérieures du Document ou dans le but d'en décourager les lecteurs n'est pas considérée comme une Copie Transparente. Une copie qui n'est pas « **Transparente** » est considérée, par opposition, comme « **Opaque** ».

Le format de fichier texte codé en ASCII générique et n'utilisant pas de balises, les formats de fichiers Texinfo ou LaTeX, les formats de fichiers SGML ou XML utilisant une DTD publiquement accessible, ainsi que les formats de fichiers HTML simple et standard, écrits de telle sorte qu'ils sont modifiables sans outil spécifique, sont des exemples de formats acceptables pour la réalisation de Copies Transparentes. Les formats suivants sont opaques : PostScript, PDF, formats de fichiers propriétaires qui ne peuvent être visualisés ou édités que par des traitements de textes propriétaires, SGML et XML utilisant des DTD et/ou des outils de formatage qui ne sont pas disponibles publiquement, et du code HTML généré par une machine à l'aide d'un traitement de texte quelconque et dans le seul but de la génération d'un format de sortie.

La « **Page de titre** » désigne, pour les ouvrages imprimés, la page de titre elle-même, ainsi que les pages supplémentaires nécessaires pour fournir clairement les informations dont cette Licence impose la présence sur la page de titre. Pour les travaux n'ayant pas de Page de titre comme décrit ci-dessus, la « **Page de titre** » désigne le texte qui s'apparente le plus au titre du document et situé avant le texte principal.

2 - COPIES CONFORMES

Vous pouvez copier et distribuer le **Document** sur tout type de support, commercialement ou non, à condition que cette Licence, la notice de copyright et la notice de la Licence indiquant que cette Licence s'applique à ce **Document** soient reproduits dans toutes les copies, et que vous n'y ajoutiez aucune condition restrictive supplémentaire. Vous ne pouvez pas utiliser un quelconque moyen technique visant à empêcher ou à contrôler la lecture ou la reproduction ultérieure des copies que vous avez créées ou distribuées. Toutefois, vous pouvez solliciter une rétribution en échange des copies. Si vous distribuez une grande quantité de copies, référez-vous aux dispositions de la section 3.

Vous pouvez également prêter des copies, sous les mêmes conditions que celles suscitées, et vous pouvez afficher publiquement des copies de ce **Document**.

3 - COPIES EN NOMBRE

Si vous publiez des copies imprimées de ce **Document** à plus de 100 exemplaires et que la Licence du

Document indique la présence de **Textes de couverture**, vous devez fournir une couverture pour chaque copie, qui présente les **Textes de couverture** des première et dernière pages de couverture du **Document**. Les première et dernière pages de couverture doivent également vous identifier clairement et sans ambiguïté comme étant l'éditeur de ces copies. La première page de couverture doit comporter le titre du **Document** en mots d'importance et de visibilité égales. Vous pouvez ajouter des informations complémentaires sur les pages de couverture. Les copies du **Document** dont seule la couverture a été modifiée peuvent être considérées comme des copies conformes, à condition que le titre du **Document** soit préservé et que les conditions indiquées précédemment soient respectées.

Si les textes devant se trouver sur la couverture sont trop importants pour y tenir de manière claire, vous pouvez ne placer que les premiers sur la première page et placer les suivants sur les pages consécutives.

Si vous publiez plus de 100 **Copies opaques** du **Document**, vous devez soit fournir une **Copie transparente** pour chaque **Copie opaque**, soit préciser ou fournir avec chaque **Copie opaque** une adresse réseau publiquement accessible d'une **Copie transparente** et complète du **Document**, sans aucun ajout ou modification, et à laquelle tout le monde peut accéder en téléchargement anonyme et sans frais, selon des protocoles réseau communs et standards. Si vous choisissez cette dernière option, vous devez prendre les dispositions nécessaires, dans la limite du raisonnable, afin de garantir l'accès non restrictif à la **Copie transparente** durant une année pleine après la diffusion publique de la dernière **Copie opaque** (directement ou *via* vos revendeurs).

Nous recommandons, mais ce n'est pas obligatoire, que vous contactiez l'auteur du **Document** suffisamment tôt avant toute publication d'un grand nombre de copies, afin de lui permettre de vous donner une version à jour du **Document**.

4 - MODIFICATIONS

Vous pouvez copier et distribuer une **Version modifiée** du **Document** en respectant les conditions des sections 2 et 3 précédentes, à condition de placer cette **Version modifiée** sous la présente Licence, dans laquelle le terme « **Document** » doit être remplacé par les termes « **Version modifiée** », donnant ainsi l'autorisation de redistribuer et de modifier cette **Version modifiée** à quiconque en possède une copie. De plus, vous devez effectuer les actions suivantes dans la **Version modifiée** :

1. Utiliser sur la **Page de titre** (et sur la page de couverture éventuellement présente) un titre distinct de celui du **Document** d'origine et de toutes ses versions antérieures (qui, si elles existent, doivent être mentionnées dans la section « **Historique »** du **Document**). Vous pouvez utiliser le même titre si l'éditeur d'origine vous en a donné expressément la permission.
2. Mentionner sur la **Page de titre** en tant qu'auteurs une ou plusieurs des personnes ou entités responsables des modifications de la **Version modifiée**, avec au moins les cinq principaux auteurs du **Document** (ou tous les auteurs s'il y en a moins de cinq).
3. Préciser sur la **Page de titre** le nom de l'éditeur de la **Version modifiée**, en tant qu'éditeur du **Document**.
4. Préserver intégralement toutes les notices de copyright du **Document**.
5. Ajouter une notice de copyright adjacente aux autres notices pour vos propres modifications.
6. Inclure immédiatement après les notices de copyright une notice donnant à quiconque l'autorisation d'utiliser la **Version modifiée** selon les termes de cette Licence, sous la forme présentée dans l'annexe indiquée ci-dessous.
7. Préserver dans cette notice la liste complète des **Sections inaltérables** et les **Textes de couverture** donnés avec la notice de la Licence du Document.
8. Inclure une copie non modifiée de cette Licence.
9. Préserver la section nommée « **Historique** » et son titre, et y ajouter une nouvelle entrée décrivant le titre, l'année, les nouveaux auteurs et l'éditeur de la **Version modifiée**, tels que décrits sur la **Page de titre**, ainsi qu'un descriptif des modifications apportées depuis la précédente version.
10. Conserver l'adresse réseau éventuellement indiquée dans le **Document** permettant à quiconque

d'accéder à une **Copie transparente** du **Document**, ainsi que les adresses réseau indiquées dans le **Document** pour les versions précédentes sur lesquelles le **Document** se base. Ces liens peuvent être placés dans la section « **Historique** ». Vous pouvez ne pas conserver les liens pour un travail datant de plus de quatre ans avant la version courante ou si l'éditeur d'origine vous en accorde la permission.

11. Si une section « **Dédicaces** » ou une section « **Remerciements** » sont présentes, les informations et les appréciations concernant les contributeurs et les personnes auxquelles s'adressent ces remerciements doivent être conservées, ainsi que le titre de ces sections.
12. Conserver sans modification les **Sections inaltérables** du **Document**, ni dans leurs textes, ni dans leurs titres. Les numéros de sections ne sont pas considérés comme faisant partie du texte des sections.
13. Effacer toute section intitulée « **Approbations** ». Une telle section ne peut pas être incluse dans une **Version modifiée**.
14. Ne pas renommer une section existante sous le titre « **Approbations** » ou sous un autre titre entrant en conflit avec le titre d'une **Section inaltérable**.

Si la **Version modifiée** contient de nouvelles sections préliminaires ou de nouvelles annexes considérées comme des **Sections secondaires** et que celles-ci ne contiennent aucun élément copié à partir du Document, vous pouvez à votre convenance en désigner une ou plusieurs comme étant des **Sections inaltérables**. Pour ce faire, ajoutez leurs titres dans la liste des **Sections inaltérables** au sein de la notice de Licence de la version Modifiée. Ces titres doivent êtres distincts des titres des autres sections.

Vous pouvez ajouter une section nommée « **Approbations** » à condition que ces approbations ne concernent que les modifications ayant donné naissance à la **Version modifiée** (par exemple, comptes rendus de revue du document ou acceptation du texte par une organisation le reconnaissant comme étant la définition d'un standard).

Vous pouvez ajouter un passage comprenant jusqu'à cinq mots en première page de couverture, et jusqu'à vingt-cinq mots en dernière page de couverture, à la liste des **Textes de couverture** de la **Version modifiée**. Il n'est autorisé d'ajouter qu'un seul passage en première et en dernière pages de couverture par personne ou groupe de personnes ou organisation ayant contribué à la modification du **Document**. Si le Document comporte déjà un passage sur la même couverture, ajouté en votre nom ou au nom de l'organisation au nom de laquelle vous agissez, vous ne pouvez pas ajouter de passage supplémentaire ; mais vous pouvez remplacer un ancien passage si vous avez expressément obtenu l'autorisation de l'éditeur de celui-ci.

Cette Licence ne vous donne pas le droit d'utiliser le nom des auteurs et des éditeurs de ce **Document** à des fins publicitaires ou pour prétendre à l'approbation d'une **Version modifiée**.

5 - FUSION DE DOCUMENTS

Vous pouvez fusionner le **Document** avec d'autres documents soumis à cette Licence, suivant les spécifications de la section 4 pour les **Versions modifiées**, à condition d'inclure dans le document résultant toutes les **Sections inaltérables** des documents originaux sans modification, et de toutes les lister dans la liste des **Sections inaltérables** de la notice de Licence du document résultant de la fusion.

Le document résultant de la fusion n'a besoin que d'une seule copie de cette Licence, et les **Sections inaltérables** existant en multiples exemplaires peuvent être remplacées par une copie unique. S'il existe plusieurs **Sections inaltérables** portant le même nom mais de contenu différent, rendez unique le titre de chaque section en ajoutant, à la fin de celui-ci, entre parenthèses, le nom de l'auteur ou de l'éditeur d'origine, ou, à défaut, un numéro unique. Les mêmes modifications doivent être réalisées dans la liste des **Sections inaltérables** de la notice de Licence du document final.

Dans le document résultant de la fusion, vous devez rassembler en une seule toutes les sections « Historique » des documents d'origine. De même, vous devez rassembler les sections « Remerciements » et « Dédicaces ». Vous devez supprimer toutes les sections « Approbations ».

6 - REGROUPEMENTS DE DOCUMENTS

Vous pouvez créer un regroupement de documents comprenant le **Document** et d'autres documents soumis à cette Licence, et remplacer les copies individuelles de cette Licence des différents documents par une unique copie incluse dans le regroupement de documents, à condition de respecter pour chacun de ces documents l'ensemble des règles de cette Licence concernant les copies conformes.

Vous pouvez extraire un document d'un tel regroupement et le distribuer individuellement sous couvert de cette Licence, à condition d'y inclure une copie de cette Licence et d'en respecter l'ensemble des règles concernant les copies conformes.

7 - AGRÉGATION AVEC DES TRAVAUX INDÉPENDANTS

La compilation du **Document** ou de ses dérivés avec d'autres documents ou travaux séparés et indépendants sur un support de stockage ou sur un média de distribution quelconque ne représente pas une **Version modifiée** du **Document** tant qu'aucun copyright n'est déposé pour cette compilation. Une telle compilation est appelée « agrégat » et cette Licence ne s'applique pas aux autres travaux indépendants compilés avec le **Document** s'ils ne sont pas eux-mêmes des travaux dérivés du **Document**.

Si les exigences de la section 3 concernant les **Textes de couverture** sont applicables à ces copies du **Document**, et si le **Document** représente un volume inférieur à un quart du volume total de l'agrégat, les **Textes de couverture** du **Document** peuvent être placés sur des pages de couverture qui n'encadrent que le **Document** au sein de l'agrégat. Dans le cas contraire, ils doivent apparaître sur les pages de couverture de l'agrégat complet.

8 - TRADUCTION

La traduction est considérée comme une forme de modification, vous pouvez donc distribuer les traductions du **Document** selon les termes de la section 4. Vous devez obtenir l'autorisation spéciale des auteurs des **Sections inaltérables** pour les remplacer par des traductions, mais vous pouvez inclure les traductions des **Sections inaltérables** en plus des textes originaux. Vous pouvez inclure une traduction de cette Licence à condition d'inclure également la version originale en anglais. En cas de contradiction entre la traduction et la version originale en anglais, c'est cette dernière qui prévaut.

9 - RÉVOCATION

Vous ne pouvez pas copier, modifier, sous-licencier ou distribuer le **Document** autrement que selon les termes de cette Licence. Tout autre acte de copie, modification, sous-Licence ou distribution du **Document** est sans objet et vous prive automatiquement des droits que cette Licence vous accorde. En revanche, les personnes qui ont reçu de votre part des copies ou les droits sur le document sous couvert de cette Licence ne voient pas leurs droits révoqués tant qu'elles en respectent les principes.

10 - RÉVISIONS FUTURES DE CETTE LICENCE

La Free Software Foundation peut publier de temps en temps de nouvelles versions révisées de cette Licence. Ces nouvelles versions seront semblables à la présente version dans l'esprit, mais pourront différer sur des points particuliers en fonction de nouvelles questions ou nouveaux problèmes. Voyez <A HREF="http://www.gnu.org/copyleft/">http://www.gnu.org/copyleft/</A> pour plus de détails.

Chaque version de cette Licence est dotée d'un numéro de version distinct. Si un **Document** spécifie un numéro de version particulier de cette Licence, et porte la mention « ou toute autre version ultérieure », vous pouvez choisir de suivre les termes de la version spécifiée ou ceux de n'importe quelle version ultérieure publiée par la Free Software Foundation. Si aucun numéro de version n'est spécifié, vous pouvez choisir n'importe quelle version officielle publiée par la Free Sofware Foundation.

Comment utiliser cette Licence pour vos documents

Pour utiliser cette Licence avec un document que vous avez écrit, incorporez une copie du texte de cette Licence en anglais et placez le texte ci-dessous juste après la page de titre :

Copyright (c) **ANNÉE VOTRE NOM**

Permission vous est donnée de copier, distribuer et/ou modifier ce document selon les termes de la Licence GNU Free Documentation License, Version 1.1 ou ultérieure publiée par la Free Software Foundation ; avec les sections inaltérables suivantes :

LISTE DES TITRES DES SECTIONS INALTÉRABLES

Avec le texte de première page de couverture suivant :

TEXTE DE PREMIÈRE PAGE DE COUVERTURE

Avec le texte de dernière page de couverture suivant :

TEXTE DE DERNIÈRE PAGE DE COUVERTURE

Une copie de cette Licence est incluse dans la section appelée **GNU Free Documentation License** de ce document.

Si votre Document ne comporte pas de section inaltérable, de textes de première et dernière pages de couverture, veuillez insérer les mentions suivantes dans les sections adéquates :

- pas de section inaltérable -

- pas de texte de première page de couverture -

- pas de texte de dernière page de couverture -

Vous pouvez également fournir une traduction de la Licence GNU FDL dans votre document, mais celle-ci ne doit pas remplacer la version anglaise. La section intitulée **GNU Free Documentation License** doit contenir la version anglaise de la Licence GNU FDL, c'est la seule qui fait foi.

Si votre Document contient des exemples non triviaux de code programme, nous recommandons de distribuer ces exemples en parallèle sous Licence GNU General Public License, qui permet leur usage dans les logiciels libres.

Annexe B - Texte original de la licence GNU FDL

Version 1.2, November 2002

SOMMAIRE

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively

with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

A. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
B. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
C. State on the Title page the name of the publisher of the Modified Version, as the publisher.
D. Preserve all the copyright notices of the Document.
E. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
F. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
G. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
H. Include an unaltered copy of this License.
I. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
J. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it

was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.

K. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.

L. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.

M. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.

N. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.

O. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties--for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements."

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided for under this License. Any other attempt to copy, modify, sublicense or distribute the Document is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation.

External links

- Official GNU FDL webpage

Pour la Liberté