https://clx.asso.fr/spip/?Qui-a-casse-la-fenetre

TCP et linux 2.6.8

Qui a cassé la fenêtre ?

- Documentations -

Date de mise en ligne : mardi 23 novembre 2004

Puisqu'on vous dit que le Libre c'est bien : pourtant, qu'une modification du noyau empêche de consulter certains sites, et il ne faut pas moins de deux versions pour voir enfin corriger le problÚme, souligneront les mauvaises langues. Sauf qu'on n'est plus sensé, de nos jours, mettre à mort le porteur d'une mauvaise nouvelle (confondre le messager et le message lui-même). Lorsqu'il est nécessaire de « rouvrir la boîte noire », justement, rien ne vaut le Libre. Accessoirement, on peut même y lire, à code source ouvert, que la rhétorique, le social, le politique, l'affectif et toutes autres sortes de choses « impures » se mêlent inextricablement à la compilation du binaire. Illustration.

Le problème par les faits

Une modification de la gestion de la fenêtre TCP introduite dans le noyau 2.6.7 (eh oui ! le 2.6.7 posait déjà problème) a pour effet de bord de mettre en évidence un défaut de certains routeurs et pare-feux. Ce bug (qui, à proprement parler, ne ressortit pas au noyau lui-même mais à des tierces parties) se manifeste par l'impossibilité de contacter les sites Web (mais aussi bien tout autre type de serveur causant TCP) situés en amont de ces éléments défectueux.

Le problème par les causes

Comme chacun sait, la fenêtre TCP n'est rien d'autre qu'un tampon permettant d'accumuler les paquets entrants dans la pile, ce qui dispense d'avoir à les acquitter à la queue leu leu auprès de l'expéditeur avant de recevoir le suivant.

Ainsi, par exemple, au lieu d'attendre que le paquet nÂ° 10 soit acquitté par le récepteur avant d'envoyer le paquet nÂ° 11, puis le nÂ° 12, et ainsi de suite, l'émetteur connaissant la taille de la fenêtre du récepteur sait que ce dernier dispose d'un tampon de 65635 octets (le maximum hors extension) autorisant l'envoi de 43 (65635 / 1500) paquets avant saturation. Ce procédé permet d'accroître sensiblement la vitesse de transmission. L'idée est d'envoyer une rafale de paquets sans attendre l'acquittement de chaque paquet : si le récepteur accuse réception du paquet nÂ° 1024 (en pratique, cela revient à réclamer le nÂ° 1025), eh bien on considère qu'il acquitte du même coup les 1023 précédents !

La taille de cette fenêtre est codée sur 16 bits dans l'en-tête d'un paquet TCP, ce qui autorise un maximum de 64Ko. Sachant qu'un paquet

standard a une longueur totale de 1500 octets, ça ne fait finalement pas beaucoup sur une liaison rapide (on est alors très en dessous de la seconde). C'est pourquoi la RFC 1323 â€" comme par hasard co-rédigée par quelqu'un de chez *Cray*) a introduit, entre autres extensions destinées à améliorer les performances du protocole TCP, une option permettant d'accroître la taille de la fenêtre par une astuce consistant à ajouter un coefficient multiplicateur. Ainsi, malgré le fait que la « case » destinée à indiquer la taille de la fenêtre reste codée sur 16 bits, cet artifice permet de faire « comme si » elle en contenait 32 â€" soit une capacité sensiblement plus importante. Chose, encore une fois, particulièrement souhaitable sur des liaisons à large bande passante, mais plus encore à forte latence (satellites géostationnaires).

Si aucune des deux parties ne gère cette extension, tout va bien. La taille de la fenêtre passée dans l'en-tête des premiers paquets échangés n'a pas a être interprêtée autrement que littéralement. Idem si l'une seulement des deux parties est capable de la gérer : puisqu'il s'agit de préserver la compatibilité, les deux s'en passent. D'ailleurs, il suffit que le premier paquet (SYN) ne comporte pas l'option, pour que le partenaire soit dans l'obligation d'y renoncer aussi.

Même ainsi, la taille des fenêtres peut différer de chaque côté : la fenêtre maximale du client est indiquée dans le premier paquet (SYN), tandis que celle du serveur est indiquée dans le paquet d'acquittement (SYN+ACK) correspondant. (Contrairement à ce qui se passe en UDP, on ne commence pas à envoyer les paquets en rafale avant d'avoir complété une première séquence d'acquittement en bonne et due forme.) Enfin le client accuse à son tour réception de la taille de fenêtre annoncée par le serveur (ACK), avant que ne puisse effectivement débuter la session TCP.

C'est seulement lorsque l'une des parties se méprend sur les intentions de l'autre que les choses se gâtent. Cela se produit si, par exemple, un pare-feu situé entre le client et le serveur remet à zéro la valeur d'échelle de la fenêtre TCP des paquets SYN et SYN+ACK qui passent entre ses pattes (de goret ;o) sans pour autant désactiver l'option (tcp window scaling). Entre-nous, c'est une curieuse manière de faire, car l'option tient sur trois octets contigüs : le premier indique le type d'option (type 3), le second la longueur requise (3 octets), et enfin le troisième contient le coefficient. Pourquoi tripoter seulement le dernier octet, ce qui revient à indiquer qu'on gère l'option sans pour autant souhaiter en bénéficier (troisième octet remis à zéro) ? Tss... tss... ça sent la parano obtuse â€" ou à tout le moins la configuration mal maîtrisée. On peut imaginer plusieurs jeux de règles de pare-feu produisant involontairement ce facheux résultat. En particulier si ce dernier se mêle de recréer de toute pièce les premiers paquets d'une session TCP sans être d'accord sur la taille de fenêtre acceptable avec le serveur qu'il protège dans la DMZ. (Tip : chercher les paquets susceptibles de passer sans générer d'état).

À priori, on pourrait considérer le fait que l'une des parties infère faussement que l'autre renonce au coefficient multiplicateur comme gênant mais pas rédhibitoire pour la connexion. Cela devrait seulement réduire la taille de fenêtre possible et ralentir (plus ou moins significativement, selon la bande passante) la communication.

Pourtant, en pratique, cela se passera mal la plupart du temps. Certes le serveur proprement dit gère l'option, mais comme son pare-feu lui présente un paquet dont le coefficient multiplicateur de taille de fenêtre est toujours à zéro, il prend pour argent comptant la taille de la fenêtre et ne l'ajuste pas en conséquence. Ainsi, si le client souhaite (et croit) annoncer une fenêtre de 128Ko (soit le double de ce qu'elle pourrait être si l'extension était désactivée â€" ce qui n'est déjà pas mal en terme d'occupation mémoire), il place la valeur 1024 dans le champ réservé à la taille de fenêtre, et 7 dans celui réservé au multiplicateur. Le calcul de la valeur réelle s'effectue par une rotation de bits sur la gauche : sept rotations égalent 128, ce qui devrait donner 1024 * 128 = 128Ko. Las ! le serveur ne voit pas le coefficient 7 mais zéro. Il considère par conséquent que la taille de fenêtre du client est de 1024 octets seulement, soit moins qu'un paquet (1500) et n'envoie donc rien du tout. (En réalité, chaque paquet placé dans la pile compte moins de 1500 octets puisque la partie IP a déjà été déjà décapsulée : pour ceux qui suivent, c'est bien de l'avoir noté, mais cela représente une très faible différence dans le calcul).

Et voilà, Madame, pourquoi votre fille est muette.

On voit que la valeur 7 retenue en pratique depuis le noyau 2.6.7 est critique. Tant que le noyau adoptait une valeur inférieure, cela se passait bien dans l'immense majorité des cas, puisqu'il y avait très peu de chances que la valeur de la fenêtre (même faussement divisée du fait que l'autre partie ne « voyait » pas le coefficient) soit inférieure à l'équivalent d'au moins un paquet. Avec une valeur de coefficient égale à 2, cela revient à diviser à tort la taille de la fenêtre par quatre. Avec une valeur de coefficient égale à 7 par défaut, cela revient cette fois à la diviser par 128. Dès lors, il y a beaucoup plus de chance que le résultat soit interprêté comme inférieur à la taille d'un seul paquet !

Le bug pré-existait (sur certains routeurs ou pare-feux), il faisait déjà problème, mais les dégats étaient limités à un ralentissement plus ou moins perceptible. C'est le coup de force qui a consisté à passer le « TCP win scale » à 7 par défaut (valeur sans doute intéressante pour les liaisons à large bande passante mais peut-être pas pour l'ADSL) qui a servi de révélateur en cassant complètement certaines connexions déjà foireuses.

Solutions possibles

Méthode individuelle et bûcheronne (peut-être sans douleur si l'on ne dispose que d'une liaison DSL et pas d'un réseau local à 100Mb), désactiver complètement le tcp window scaling.

```
# sysctl -w net.ipv4.tcp_window_scaling=0
```

Il est possible de rendre cela permanent en éditant le fichier */etc/sysctl.conf* qui devrait se trouver dans toute distribution raisonnable.

L'option à pour effet d'annoncer aux serveurs contactés que l'on ne supporte pas l'option et doit marcher dans la plupart des cas car, du même coup, on invite le noyau à annoncer une taille de fenêtre jamais inférieure à un certain nombre de paquets.

Il est évidemment hors de question de coder cette valeur en dur dans le noyau, cela reviendrait à tirer un trait sur une fonctionnalité qui date de plusieurs années (la RFC 1323 a été publiée en mai 1992 !) Néanmoins, de nombreux utilisateurs se plaignent de ne pouvoir accéder à certains sites.

Conscient du problème, Stephane Hemminger, contributeur régulier de la liste linux-netdev a proposé une première rustine le 6 juillet dernier. Bien qu'il ait pris la précaution d'annoncer que son patch ne visait qu'à « prendre en compte la corruption du monde », celui-ci a été tout d'abord refusé par David S. Miller, le respsonsable de la partie réseau du noyau. L'argument était le suivant : si le patch est adopté, alors le bug ne sera plus apparent mais continuera d'exister (comme avant). Il est finalement préférable qu'il apparaisse en plein jour, afin de contraindre à le traiter.

Le noyau 2.6.8 (puis rapidement le 2.6.8.1, son quasi-clone à un bug NFS près) fut donc publié sans la correction, bien que le problème ait été déjà largement repéré lors de sa phase de mise au point. Évidemment, les plaintes des utilisateurs se sont multipliées, et cela a généralement contribué à mettre le bug « des autres » en évidence ;o) Aussi, surfant peut-être un peu sur l'expression de ces plaintes au sein même de la liste linux-netdev, Stephane Hemminger est revenu à la charge le 26 août (soit moins de deux mois plus tard) avec un nouveau patch destiné cette fois au noyau 2.6.9, en argumentant toujours sur la nécessité de survivre dans un monde imparfait, mais en mettant surtout en avant une nouvelle méthode de calcul de la taille de fenêtre par défaut (laquelle, certes, prend sournoisement en compte le cas des routeurs/pare-feux indélicats) et qui dispense de coder le modificateur (la fameuse valeur 7) en dur dans le noyau. « Je capitule ! » a enfin lâché le mainteneur de la partie réseau du noyau. C'est ainsi que le 2.6.9 « corrige » le problème. Et tant pis pour les distributions qui sont « sorties » avec le 2.6.8.

Tout est bien qui finit bien ? Certainement pas ! On imagine que le problème ressurgira tôt ou tard de manière cruciale, puisqu'il n'est pas vraiment résolu mais seulement contourné. Dans l'intervalle, la « boîte noire » de la gestion de la fenêtre TCP peut être refermée pour la plupart des utilisateurs â€" tant que son imperfection (résultat d'un compromis entre ce qui serait théoriquement souhaitable et le monde réel) ne se rappelle pas au bon souvenir des humains...

Moralité : comme on s'en serait douté, dans le bazar il ne suffit pas d'être un programmeur avisé pour voir une proposition de modification du noyau adoptée ; encore faut-il faire montre de rhétorique (diplomatie ?) Ensuite, ce n'est pas seulement dans les commentaires qu'on peut repérer la rhétorique dans le code source, mais bien déjà dans le code lui-même. La technique n'est ni « neutre », ni « pure » et, comme en politique, tout un chacun à son avis à donner, même s'il n'y est pas dûment autorisé par un diplôme ou je ne sais quelle « compétence ». Dans le logiciel libre, celà peut se lire « à code source ouvert ».