Groupe de travail Réseau
**Request For Comments : 816**
Traduction Claude Brière de L'Isle

David D. Clark
MIT Laboratory for Computer Science
juillet 1982

# Isolement et récupération de faute

## 1. Introduction

À l'occasion, un réseau ou un routeur tombent en panne, et la séquence de bonds que prend le paquet de la source à la destination doit changer. L'isolement des fautes est l'action qu'entreprennent collectivement les hôtes et les routeurs pour déterminer que quelque chose ne va pas ; la récupération de faute est l'identification et le choix d'un chemin de remplacement qui va servir à reconnecter la source à la destination. En fait, les routeurs effectuent la plus grande partie de l'isolement et de la récupération de faute. Il y a cependant quelques actions que les hôtes doivent entreprendre si ils souhaitent assurer un niveau de service raisonnable. Le présent document décrit la portion de l'isolement et de la récupération de faute qui est de la responsabilité de l'hôte.

## 2. Ce que font les routeurs

Les routeurs mettent collectivement en œuvre un algorithme qui identifie le meilleur chemin entre toutes les paires de réseaux. Ils le font en échangeant des paquets qui contiennent la dernière opinion de chaque routeur sur l'état de fonctionnement de ses réseaux et routeurs du voisinage. En supposant que cet algorithme fonctionne correctement, on peut s'attendre à ce que les routeurs passent par une période de confusion immédiatement après que certains réseaux ou routeurs sont tombés en panne, mais on peut supposer qu'une fois passée une période de négociation, les routeurs sont équipés d'un modèle cohérent et correct de la connexité de l'internet. À présent, cette période de négociation peut en fait prendre plusieurs minutes, et de nombreuses mises en œuvre de TCP arrivent en fin de temporisation au sein de cette période, mais c'est un objectif de conception de l'algorithme final que les routeurs soient capables de reconstruire assez rapidement la topologie pour qu'une connexion TCP soit capable de survivre à une défaillance du chemin.

## 3. Algorithme d'hôte pour la récupération de faute

Comme les routeurs tentent toujours d'avoir un modèle cohérent et correct de la topologie de l'inter réseau, la stratégie d'hôte pour la récupération de faute est très simple. Chaque fois que l'hôte a le sentiment que quelque chose ne va pas, il demande conseil au routeur, et en supposant que l'avis est amical, il suit complètement cet avis. L'avis ne sera erroné que pendant la période transitoire de la négociation, qui suit immédiatement une panne, mais sera autrement le plus souvent correct.

En fait, il n'est jamais nécessaire qu'un hôte demande explicitement un conseil à un routeur, parce que le routeur va le donner lorsque c'est approprié. Lorsque un hôte envoie un datagramme à un réseau distant, l'hôte devrait être prêt à recevoir l'un des deux messages d'avis que le routeur peut envoyer. Le message ICMP "redirection" indique que le routeur auquel l'hôte a envoyé le datagramme n'est plus le meilleur routeur pour atteindre le réseau en question. Le routeur va devoir transmettre le datagramme, mais l'hôte devrait réviser son tableau d'acheminement pour avoir une adresse immédiate différente pour ce réseau. Le message ICMP "destination injoignable" indique que par suite d'une panne, il est actuellement impossible d'atteindre d'aucune façon le réseau ou l'hôte dont on a l'adresse. À réception de ce message, un hôte peut soit abandonner immédiatement la connexion sans autre retransmission, soit envoyer à nouveau lentement pour voir si la faute est corrigée dans un délai raisonnable.

Si un hôte pouvait supposer que ces deux messages ICMP vont toujours arriver lorsque quelque chose va de travers dans le réseau, aucune autre action ne serait alors nécessaire de la part de l'hôte pour tenir ses tableaux d'acheminement en condition optimale. Malheureusement, il y a deux circonstances dans lesquelles les messages ne vont pas arriver de la façon appropriée. D'abord, durant la période transitoire qui suit une défaillance, les messages qui arrivent peuvent ne pas représenter correctement l'état de la réalité. Donc, les hôtes doivent prendre avec précaution un message d'erreur isolé. (On discute plus en détails un peu plus loin de cette période transitoire.) Ensuite, si l'hôte a envoyé des datagrammes à un routeur particulier, et si ce routeur lui-même tombe en panne, tous les autres routeurs de l'internet vont alors reconstruire la topologie, mais le routeur en question sera toujours en panne, et ne pourra donc renvoyer aucun avis à l'hôte. Tant que l'hôte continue de diriger les datagrammes sur ce routeur mort, ceux-ci vont simplement disparaître de la surface du globe, et rien ne viendra en retour. Les hôtes doivent détecter cette défaillance.

Si c'est un routeur qui se trouve à de nombreux bonds qui est en panne, cela ne pose pas de problème à l'hôte, car la

découverte de la défaillance est de la responsabilité des routeurs de son voisinage immédiat, qui vont effectuer cette action d'une manière invisible pour l'hôte. Le problème n'apparaît que si le tout premier routeur, celui auquel l'hôte est l'envoyeur immédiat des datagrammes, tombe en panne. On identifie donc une tâche unique que l'hôte doit effectuer au titre de sa part dans l'isolement de faute dans l'internet : l'hôte doit avoir une stratégie de détection de la mort du routeur auquel il envoie des datagrammes.

Supposons pour le moment que l'hôte mette en œuvre un algorithme pour détecter les routeurs défaillants ; on reviendra plus tard à la discussion de ce que devrait être cet algorithme. D'abord, examinons ce que l'hôte devrait faire lorsque il a déterminé que le routeur est mort. En fait, à l'exception d'un petit problème, l'action que devrait effectuer l'hôte est extrêmement simple. L'hôte devrait choisir quelque autre routeur, et essayer de lui envoyer le datagramme. Si on suppose que le routeur est actif, cela va produire un résultat correct ou un message ICMP. Comme on suppose que, en ignorant les périodes temporaires qui suivent immédiatement une panne, tout routeur est capable de donnée un avis correct, une fois que l'hôte a reçu l'avis de n'importe quel routeur, cet hôte est dans une condition aussi bonne qu'il pourrait l'espérer.

Il y a toujours la possibilité déplaisante que lorsque l'hôte essaye un routeur différent, celui-ci soit également en panne. Donc, quel que soit l'algorithme que l'hôte utilise pour détecter un routeur en panne, celui-ci doit être continuellement appliqué car l'hôte essaye tour à tour chaque routeur qu'il connaît.

La seule partie difficile de cet algorithme est de spécifier les moyens par lesquels l'hôte entretient le tableau de tous les routeurs auxquels il a un accès immédiat. Actuellement, la spécification du protocole Internet n'élabore aucun message par lequel un hôte puisse demander qu'on lui fournisse un tel tableau. La raison en est que les différents réseaux peuvent utiliser des mécanismes très différents pour remplir ce tableau. Par exemple, si le réseau est un réseau de diffusion, comme un ethernet ou un réseau en anneau, chaque routeur peut simplement diffuser de temps en temps un tel tableau, et l'hôte n'a rien à faire d'autre que d'écouter pour obtenir les informations requises. Autrement, le réseau peut fournir le mécanisme d'adressage logique, par lequel tout un ensemble de machines peut être alimenté avec une seule adresse de groupe, à laquelle une demande peut être envoyée pour obtenir assistance. En dehors de ces deux schémas, l'hôte peut construire un tableau des routeurs du voisinage en se souvenant de tous les routeurs desquels il a déjà reçu un message. Finalement, dans certains cas, il peut être nécessaire pour ce tableau, ou à tout le moins pour les entrées initiales du tableau, qu'il soit construit manuellement par un gestionnaire ou opérateur sur le site. Dans les cas où le réseau en question ne fournit absolument aucun support pour cette sorte d'interrogation d'hôte, au moins quelque intervention manuelle sera requise pour le démarrage, afin que l'hôte puisse découvrir au moins un routeur.

## 4. Algorithmes d'hôte pour l'isolement de faute

Revenons maintenant à la question soulevée plus haut. Quelle stratégie l'hôte devrait-il utiliser pour détecter qu'il parle à un routeur mort, afin qu'il puisse savoir qu'il doit passer à un autre routeur de sa liste. En fait, il y a plusieurs algorithmes qui peuvent être utilisés. Tous sont d'une simplicité de mise en œuvre raisonnable, mais ils ont des implications très différentes quant à la redondance chez l'hôte, le routeur, et le réseau. Donc, dans une certaine mesure, l'algorithme retenu doit dépendre des détails du réseau et de l'hôte.

### 4.1 Détection au niveau réseau

De nombreux réseau, en particulier l'Arpanet, effectuent précisément la fonction requise en interne au réseau. Si un hôte envoie un datagramme à un routeur mort sur l'Arpanet, le réseau va retourner un message "hôte mort", qui est précisément l'information dont l'hôte a besoin afin de passer sur un autre routeur. Certaines mises en œuvre précoces de l'Internet sur l'Arpanet éliminent ces messages. C'est une très mauvaise idée.

### 4.2 Interrogation continue

Le protocole ICMP fournit un mécanisme d'écho par lequel un hôte peut solliciter une réponse de la part du routeur. Un hôte pourrait simplement envoyer ce message à un intervalle raisonnable, pour s'assurer en continu que le routeur est encore actif. Cela fonctionne, mais, comme le message doit être envoyé très souvent pour détecter une faute dans un délai raisonnable, cela peut impliquer une redondance insupportable pour l'hôte lui-même, pour le réseau, et pour le routeur.

Cette stratégie est prohibée sauf lorsque une analyse spécifique a indiqué que la redondance est tolérable.

### 4.3 Interrogation déclanchée

Si l'utilisation de l'interrogation pouvait être restreinte aux seules fois où quelque chose semble aller de travers, la redondance serait alors acceptable. Pourvu qu'on puisse avoir l'avis approprié de la part d'un des protocoles de niveau supérieur, il est possible de mettre en œuvre une telle stratégie. Par exemple, on pourrait programmer le niveau TCP de telle sorte que chaque fois qu'il retransmet un segment plus d'une fois, il envoie une indication à la couche IP qui déclanche l'interrogation. Cette stratégie n'a pas une redondance excessive, mais pose le problème que l'hôte peut être un peu long à répondre à une erreur, car c'est seulement après que l'interrogation a commencé que l'hôte va être capable de confirmer que quelque chose ne va pas, et entre temps le TCP au dessus peut être déjà arrivé à expiration.

Les deux formes d'interrogation souffrent d'un handicap mineur. Les hôtes comme les routeurs répondent aux messages d'écho ICMP. Donc, l'interrogation ne peut pas être utilisée pour détecter d'erreur sur une adresse étrangère qu'on pense être celle d'un routeur et qui est en fait celle d'un hôte. Une telle confusion peut avoir lieu si l'adresse physique des machines est réarrangée.

### 4.4 Resélection déclanchée

Il y a une stratégie qui fait usage d'une indication provenant d'un niveau supérieur, comme le faisait la stratégie précédente, mais qui évite tout à fait l'interrogation. Chaque fois qu'un niveau supérieur se plaint que le service semble défectueux, la couche Internet peut prendre le prochain routeur sur la liste des routeurs disponibles, et y passer le trafic. En supposant que ce routeur est actif, aucun dommage réel ne peut découler de cette décision, même si elle était erronée, le pire qui puisse arriver est un message de redirection qui ordonne à l'hôte de revenir au routeur utilisé à l'origine. Si, d'un autre côté, le routeur d'origine est bien en panne, cela donne immédiatement un nouveau chemin, de sorte que le délai jusqu'à la récupération en est raccourci. Cette dernière stratégie semble particulièrement habile, et est probablement la plus généralement convenable pour les cas où le réseau lui-même ne fournit pas d'isolement de faute. (J'ai malheureusement oublié qui m'a suggéré cette idée. Elle n'est pas de mon invention.)

### 4.5 Détection de faute de niveau supérieur

L'exposé précédent se concentrait sur la détection et récupération de faute à la couche IP. Ce paragraphe examine ce que devraient faire les couches supérieures telles que TCP.

TCP a une seule action de récupération de faute ; il retransmet de façon répétitive un segment jusqu'à ce qu'il obtienne un accusé de réception ou que son temporisateur de connexion arrive à expiration. Comme exposé ci-dessus, il peut utiliser la retransmission comme événement qui déclanche une demande de récupération de faute à la couche IP. Dans l'autre direction, les informations peuvent provenir de la couche IP, faisant rapport de choses comme les messages ICMP Destination injoignable ou Erreur, provenant du réseau rattaché. La seule question subtile sur TCP et les fautes est ce que TCP devrait faire lorsque un tel message d'erreur arrive ou que son temporisateur de connexion arrive à expiration.

La spécification TCP discute du temporisateur. Dans la description de l'appel ouvert, le temporisateur est décrit comme une valeur facultative que le client de TCP peut spécifier ; si un segment reste non acquitté à la fin de cette période, TCP devrait interrompre la connexion. La valeur par défaut du temporisateur est de 30 secondes. Les premiers TCP étaient souvent mis en œuvre avec un intervalle de temporisation fixé, mais cela ne fonctionnait pas bien en pratique, comme peut le suggérer la discussion qui suit.

Les clients de TCP peuvent être divisés en deux classes : ceux qui fonctionnent directement au nom d'un humain, tels que Telnet, et ceux qui prennent en charge un programme, comme un envoyeur de messagerie. Les humains requièrent une réponse sophistiquée aux erreurs. Selon exactement ce qui ne va pas, ils peuvent vouloir abandonner la connexion tout de suite, ou attendre longtemps pour voir si les choses s'améliorent. Les programmes n'ont pas cette impatience humaine, mais manquent aussi du pouvoir de prendre des décisions complexes sur la base des détails de la condition d'erreur exacte. Pour eux, une simple fin de temporisation est raisonnable.

Sur la base de ces considérations, au moins deux modes de fonctionnement sont nécessaires dans TCP. D'abord, pour les programmes, les abandons de connexion sans exception si le temporisateur TCP arrive à expiration. L'autre mode, convenable pour les personnes, ne jamais abandonner la connexion de sa propre initiative, mais faire rapport à la couche supérieure lorsque le temporisateur expire. Donc, l'utilisateur humain peut voir les messages d'erreur venant de toutes les couches pertinentes, TCP et ICMP, et peut demander à TCP d'interrompre si c'est approprié. Ce second mode requiert que TCP soit capable d'envoyer un message asynchrone jusqu'au client pour faire rapport des fins de temporisations, et il exige que les messages d'erreur qui arrivent aux couches inférieures s'écoulent également à travers TCP.

Au niveau au dessus de TCP, la détection de faute est aussi requise. Un des deux événements suivants peut se produire.

D'abord, le client étranger de TCP peut avoir une défaillance, même si TCP fonctionne toujours, de sorte que les données sont quand même acquittées et que le temporisateur n'arrive jamais à expiration. Autrement, le chemin de communication peut avoir une défaillance, sans que le temporisateur TCP n'arrive à expiration, parce que le client local n'a pas de données à envoyer. Ces deux cas ont causé des problèmes.

L'envoi de messagerie donne un exemple du premier cas. Lorsqu'on envoie un message avec SMTP, il y a un accusé de réception au niveau de SMTP qui est retourné lorsque un élément de message est bien livré. Plusieurs programmes anciens de réception de messagerie vont tomber en panne juste au moment où ils ont reçu la totalité du texte du message (de sorte que TCP n'a pas détecté de fin de temporisation du fait de données en instance non acquittées) mais avant que le message ait été acquitté au niveau SMTP. Cette défaillance va faire que les envoyeurs de vieux systèmes de messagerie vont attendre indéfiniment l'accusé de réception de niveau SMTP. Le remède évident était d'établir un temporisateur au niveau SMTP, mais la première tentative pour le faire n'a pas fonctionné, parce qu'il n'y avait pas de façon simple pour choisir l'intervalle de temporisation. Si l'intervalle choisi était court, il arrivait à expiration dans le temps de fonctionnement normal lors de l'envoi d'un gros fichier à un hôte lent. Un intervalle de plusieurs minutes était nécessaire pour empêcher de fausses fins de temporisation, mais cela voulait dire que les défaillances n'étaient détectée que très lentement. La solution actuelle chez plusieurs systèmes de messagerie est de prendre un intervalle de temporisation proportionnel à la taille du message.

Le serveur Telnet fournit un exemple de l'autre sorte de défaillance. Il peut facilement arriver que la liaison de communications tombe en panne alors qu'il ne s'écoule pas de trafic, peut-être parce que l'usager réfléchit. Finalement, l'usager va tenter de taper quelque chose, et il va découvrir à ce moment que la connexion est morte et il va l'interrompre. Mais l'extrémité hôte de la connexion, qui n'a rien à envoyer, ne va rien découvrir d'anormal, et va rester éternellement en attente. Dans certains systèmes, il n'y a aucun moyen pour un usager engagé dans un processus différent de détruire ou résilier un tel processus en suspens, de sorte qu'il n'y a pas moyen de récupérer.

Une solution serait que le serveur hôte Telnet interroge l'extrémité utilisateur de temps en temps, pour voir si elle est toujours active. (Telnet n'a pas de dispositif explicite d'interrogation, mais l'hôte pourrait négocier une option sans importance, qui devrait produire en retour soit un accord, soit un désaccord.) Le seul problème serait qu'un intervalle d'échantillon raisonnable, si il est appliqué à tous les usagers sur un grand système, peut générer une quantité de trafic et de redondance système inacceptables. Un serveur Telnet intelligent n'utiliserait cette interrogation que lorsque quelque chose semble aller mal, peut-être lorsque il n'y a pas eu d'activité d'utilisateur pendant u certain temps.

Dans ces deux cas, la conclusion générale est qu'une détection d'erreur au niveau du client est nécessaire, et que les détails du mécanisme sont très dépendants de l'application. Les programmeurs d'applications doivent faire attention au problème des défaillances, et doivent comprendre que la détection d'erreur au niveau TCP ou inférieur ne peut pas résoudre pour eux la totalité du problème.

## 6. Savoir quand abandonner

Il n'est pas évident, lorsque arrivent des messages d'erreur comme une Destination injoignable ICMP, si TCP devrait abandonner la connexion. La raison pour laquelle les messages d'erreur sont difficiles à interpréter est que, comme on l'a exposé ci-dessus, après la défaillance d'un routeur ou d'un réseau, il y a une période transitoire durant laquelle les routeurs peuvent avoir des informations incorrectes, de sorte que des messages d'erreur non pertinents ou incorrects peuvent parfois être retournés. Un Destination injoignable ICMP isolé peut arriver à un hôte, par exemple, si un paquet est envoyé durant la période pendant laquelle les routeurs essayent de trouver un nouveau chemin. Pour abandonner une connexion TCP sur la base de l'arrivée d'un tel message serait ignorer les dispositions précieuses de l'Internet qui font que pour de nombreuses défaillances internes il reconstruit sa fonction sans aucune perturbation des points d'extrémité.

Mais si des messages de défaillance n'impliquent pas une défaillance, à quoi servent ils ? En fait, les messages d'erreur servent à plusieurs objets importants. D'abord, si ils arrivent en réponse à l'ouverture d'une nouvelle connexion, ils sont probablement causés par l'ouverture incorrecte de la connexion (par exemple, à une adresse non existante) plutôt que par une défaillance transitoire du réseau. Ensuite, ils fournissent des informations précieuses, après qu'est survenue une fin de temporisation TCP, sur la cause probable de la défaillance. Finalement, certains messages, comme le message ICMP Problème de paramètre, impliquent un possible problème de mise en œuvre. En général, les messages d'erreur donnent des informations précieuses sur ce qui est allé de travers, mais ils ne sont pas à prendre absolument au pied de la lettre. Un mécanisme général d'alerte, comme la fin de temporisation TCP exposé ci-dessus, fournit une bonne indication que, quelle que soit ce qui va de travers, il y a un problème sérieux, mais sans les messages indicatifs pour augmenter la temporisation, il n'y a pas de moyen pour le client de savoir comment répondre à l'erreur. La combinaison du temporisateur et de l'avis provenant des messages d'erreur donne un ensemble raisonnable de faits pour que la couche client le fasse. Il est important que les messages d'erreur de toutes les couches soient passés au module client de façon utile et cohérente.