Groupe de travail Réseau
Request for Comments : 1305
Rendues obsolètes : RFC-1119, -1059, -958
mars 1992

Editeur : David L. Mills,
Université du Delaware
Traduction Claude Brière de L'Isle

Protocole de l'heure du réseau (NTP) version 3, spécification, mise en œuvre et analyse

Résumé

Le présent document décrit le protocole de temps du réseau (NTP, *Network Time Protocol*), spécifie sa structure formelle et récapitule les informations utiles pour sa mise en œuvre. NTP donne des mécanismes pour synchroniser l'heure et coordonner sa distribution dans un grand Internet diversifié, fonctionnant à des débits allant du lent à l'ultra rapide. Il utilise un concept d'heure tournante dans lequel un sous réseau distribué de serveurs horaires fonctionnant dans une configuration autogérée avec hiérarchisation maître-esclave synchronise des horloges locales au sein du sous réseau et aux heures nationales standard via le fil ou la radio. Les serveurs peuvent aussi redistribuer l'heure de référence via des algorithmes d'acheminement locaux et des routines horaires.

Statut du présent Mémo

La présente RFC spécifie un protocole de normalisation IAB pour la communauté Internet et appelle à des discussions et suggestions pour son amélioration. Prière de se reporter à l'édition en cours des "IAB Official Protocol Standards" (*normes officielles du protocole Internet*) (STD 1) pour connaître l'état de la normalisation et le statut du présent protocole. La distribution du présent mémo n'est pas soumise à restriction.

Mots clé : synchronisation d'horloge réseau, distribution de l'heure standard, architecture à tolérance de pannes, estimation de vraisemblance maximum, oscillateur discipliné, protocole Internet, réseau à grande vitesse, spécification formelle.

Préface

Le présent document décrit la version 3 du protocole de temps du réseau (NTP). Il remplace la version 2 du protocole décrite dans la RFC-1119 datée de septembre 1989. Cependant, il n'en change pas le protocole de façon significative ni n'en rend obsolète les précédentes versions ou les mises en œuvre existantes. La principale raison de cette nouvelle version est de préciser l'analyse et les modèles de mise en œuvre pour de nouvelles applications à des vitesses de réseau bien supérieures au régime du gigabit par seconde et de répondre aux besoins de stabilité améliorée, de précision et de pertinence requis à de telles vitesses. En particulier, les sources d'erreur de temps et de fréquence ont été soigneusement examinées et les limites d'erreur établies afin d'améliorer les performances, de fournir un modèle pour des hypothèses correctes et indiquer la qualité de la précision horaire à l'utilisateur. La révision incorpore aussi deux caractéristiques facultatives nouvelles, (1) un algorithme pour combiner les décalages d'un certain nombre de serveurs horaires homologues afin d'en améliorer la précision et (2) améliorer les algorithmes d'horloge locale qui permettent que les intervalles d'interrogation sur tous les chemins de synchronisation soient substantiellement augmentés afin de réduire les frais pour le réseau. Ces changements, qui sont décrits en détail à l'Appendice D, font ci-après l'objet d'un survol rapide.

1. Dans la version 3, l'algorithme d'horloge locale a été révisé pour en améliorer la stabilité et la précision. L'Appendice G présente un modèle mathématique détaillé et un exemple de conception qui ont été affinés à l'aide de l'analyse de commande à rétroaction et de simulations extensives en utilisant les données collectées sur les canaux ordinaires de l'Internet. La section 5 de la RFC-1119 sur l'horloge locale NTP a été complètement réécrite pour décrire le nouvel algorithme. Comme celui-ci peut avoir pour résultat des débits de message très en dessous de ceux du vieux, il est vivement recommandé de les utiliser dans les nouvelles mises en œuvre. Noter que l'utilisation du nouvel algorithme n'affecte pas l'interopérabilité avec les versions antérieures ou avec les mises en œuvre existantes.

2. La version 3 présente à l'Appendice F un nouvel algorithme pour combiner les décalages d'un certain nombre de serveurs de temps homologues. Cet algorithme est modélisé sur ceux utilisés par les laboratoires de

normalisation nationaux pour combiner les décalages pondérés d'un certain nombre d'horloges standard pour construire une échelle de temps synthétique de laboratoire plus exacte que n'importe quelle horloge prise séparément. Il peut être utilisé dans une mise en œuvre NTP pour améliorer l'exactitude et la stabilité et réduire les erreurs dues aux voies asymétriques de l'Internet. Le nouvel algorithme a été simulé en utilisant des données collectées sur les canaux ordinaires de l'Internet, mis en œuvre et essayé, avec le nouvel algorithme d'horloge locale, dans les serveurs de temps Fuzzball qui fonctionnent maintenant sur l'Internet. Noter que l'utilisation du nouvel algorithme n'affecte pas interopérabilité avec les précédentes versions ou les mises en œuvre existantes.

3. Plusieurs incohérences et erreurs mineures des versions précédentes ont été corrigées dans la version 3. La description des procédures a été réécrite en pseudo code augmenté d'un commentaire pour le préciser et éviter les ambiguïtés. L'Appendice I a été ajouté pour illustrer les mises en œuvre en langage C des divers algorithmes de filtrage et de sélection suggérés pour NTP. Des informations supplémentaires sont incluses à la Section 5 et dans l'Appendice E, qui incluent le matériel didactique précédemment inclus dans la Section 2 de la RFC-1119, ainsi que de nouveaux éléments de clarification de l'interprétation des échelles de temps et des secondes sautées.

4. Des modifications mineures ont été apportées dans la version 3 aux algorithmes d'horloge locale pour éviter les problèmes observés lorsque les secondes sautées sont introduites dans l'échelle de temps UTC et aussi pour prendre en charge un oscillateur de précision auxiliaire, tel qu'une horloge au césium ou un récepteur de synchronisation, comme base de temps de précision. De plus, des changements ont été apportés à certaines procédures décrites à la Section 3 et dans les procédures de filtre d'horloge et de choix d'horloge décrites à la Section 4. Alors que ces changements ont été faits pour corriger des erreurs mineures révélées par l'expérience et sont recommandées pour les nouvelles mises en œuvre, elles n'affectent l'interopérabilité avec les versions précédentes ou les mises en œuvre existantes que de façon mineure (au moins jusqu'au prochain saut de seconde).

5. Des changements ont été faits dans la version 3 sur la façon dont sont définis, calculés et traités le retard, le décalage et la dispersion, afin de fixer des limites fiables aux erreurs inhérentes aux procédures de transfert de temps. En particulier, les accumulations d'erreur ont été déplacées du calcul du retard au calcul de la dispersion et inclus tous deux dans les procédures de filtre et de choix d'horloge. La procédure de choix d'horloge a été modifiée pour retirer la première des deux étapes de tri/élimination et la remplacer par un algorithme d'abord proposé par Marzullo et incorporé plus tard dans le service d'heure numérique *(Digital Time Service)*. Ces changements n'affectent pas significativement le fonctionnement ordinaire ni la compatibilité avec les différentes versions de NTP, mais ils donnent une base à des déclarations formelles d'exactitude telles que décrites à l'Appendice H.

Table des matières

1 Introduction

Le présent document constitue une spécification formelle de la version 3 du protocole de temps du réseau (NTP, *Network Time Protocol*), qui sert à synchroniser la conservation du temps parmi un ensemble distribué de serveurs et clients de l'heure. Il définit les architectures, algorithmes, entités et protocoles utilisés par NTP et est principalement destiné aux développeurs. Un document parallèle [MIL91a] résume les exigences, modèles analytiques, analyses algorithmiques et performances dans les conditions normales de l'Internet. Un autre document [MIL91b] décrit l'échelle de temps NTP et ses relations avec les autres échelles de temps standard utilisées actuellement. NTP a été décrit pour la première fois dans la RFC-958 [MIL85c], mais a depuis évolué de façon significative, culminant avec la plus récente version 2 de NTP décrite dans la RFC-1119 [MIL89]. Il est construit sur le protocole Internet (IP) [DAR81a] et le protocole de datagramme d'utilisateur (UDP, *User Datagram Protocol*) [POS80], qui fournit un mécanisme de transport sans connexion ; cependant, il est d'ores et déjà adaptable aux autres suites de protocole. NTP est tiré du protocole de temps [POS83b] et du message d'horodatage ICMP [DAR81b], mais il est spécifiquement conçu pour entretenir l'exactitude et la robustesse, même lorsque utilisé sur des canaux Internet normaux qui impliquent plusieurs passerelles, des retards accroissant la dispersion et des réseaux non fiables.

L'environnement de service consiste en un modèle de mise en œuvre et un modèle de service décrits à la section 2. Le modèle de mise en œuvre se fonde sur une architecture de système de fonctionnement multitraitement, bien que d'autres architectures puissent aussi bien être utilisées. Le modèle de service se fonde sur une conception de temps renversable qui dépend uniquement des décalages d'horloge mesurés, mais n'exige pas de livraison fiable de message. Le sous réseau de synchronisation utilise une configuration hiérarchisée maître/esclave auto gérée, avec des chemins de synchronisation déterminés par un arbre d'expansion à pondération minimum. Alors que plusieurs maîtres (serveurs primaires) peuvent exister, il n'y a pas d'exigence d'un choix de protocole.

Le protocole NTP est décrit à la Section 3. Il fournit les mécanismes de protocole pour synchroniser le temps en principe jusqu'à des précisions de l'ordre de la nanoseconde tout en préservant une date non ambiguë loin en avant dans le prochain siècle. Le protocole inclut des dispositions pour spécifier les caractéristiques et estimer les erreurs d'horloge locale et le serveur horaire auquel elle peut être synchronisée. Il inclut aussi des dispositions pour fonctionner avec un certain nombre de sources de référence principale hiérarchiquement distribuées, mutuellement suspectes, telles que les horloges radio-synchronisées.

La Section 4 décrit des algorithmes utiles pour le dérayage *(deglitching)* et le lissage des échantillons de décalage d'horloge collectés sur une base continue. Ces algorithmes découlent de ceux suggérés dans [MIL85a], et ont été affinés par suite des expériences décrites dans [MIL85b] puis ont évolué dans les conditions normales de fonctionnement pendant les trois dernières années. De plus, par suite de l'expérience acquise dans le fonctionnement de sous réseaux à plusieurs serveurs, y compris les horloges radio de plusieurs sites des USA et de clients aux USA et en Europe, des algorithmes fiables pour le choix de bonnes horloges ont été développés à partir d'une population qui peut en inclure des exemplaires hors d'usage [DEC89], [MIL91a] et ils sont décrits à la Section 4.

Les précisions qu'on peut obtenir avec NTP dépendent fortement de la précision du matériel d'horloge locale, de la sévérité des contrôles et du traitement des délais de latence. Les dispositions doivent inclure l'ajustement de l'heure et de la fréquence d'horloge logique en réponse aux corrections produites par NTP. La Section 5 décrit un concept d'horloge locale découlant de la mise en œuvre Fuzzball décrite dans [MIL83b] et [MIL88b]. Ce concept inclut des mécanismes de balayage de décalage, de compensation de fréquence et de dérayage capables de donner une exactitude de l'ordre d'une milliseconde, même après de longues périodes de perte de synchronisation avec les sources de référence principales.

Les détails spécifiques des formats de paquets NTP utilisés avec le protocole Internet (IP) et le protocole de datagramme d'utilisateur (UDP) sont présentés à l'Appendice A, et les détails du message auxiliaire de contrôle NTP suggéré, qui peut être utilisé lorsque des facilités complètes de surveillance réseau ne sont pas disponibles, sont présentés à l'Appendice B. L'Appendice C contient des détails de spécification et de mise en œuvre d'un mécanisme d'authentification facultatif qui peut être utilisé pour contrôler l'accès et empêcher des modifications de données non autorisées, et l'Appendice D contient une liste des différences entre la version 3 de NTP et les versions précédentes. L'Appendice E s'étend sur les questions d'échelle de temps de précision et de date calendaire particulières aux réseaux d'ordinateurs et à NTP. L'Appendice F décrit un algorithme facultatif pour améliorer l'exactitude en combinant les décalages de temps d'un certain nombre d'horloges. L'Appendice G présente un modèle mathématique détaillé et l'analyse des algorithmes d'horloge locale de NTP. L'Appendice H analyse les sources et la propagation des erreurs et

présente les principes d'exactitude qui se rapportent au service de transfert de l'heure. L'Appendice I illustre des segments de code en langage C pour le filtre d'horloge, le choix d'horloge et les algorithmes qui s'y rapportent, décrits à la Section 4.

1.1 Technologie concernée

D'autres mécanismes ont été spécifiés dans la suite de protocole Internet pour enregistrer et transmettre l'heure à laquelle est survenu un événement, parmi lesquels le protocole Daytime *(heure du jour)* [POS83a], le protocole Time *(heure)* [POS83b], le message d'horodatage ICMP [DAR81b] et l'option d'horodatage IP [SU81]. Les résultats expérimentaux sur les décalages d'horloge mesurés et sur les délais d'aller-retour dans l'Internet sont discutés dans [MIL83a], [MIL85b], [COL88] et [MIL88a]. D'autres algorithmes de synchronisation sont discutés dans [LAM78], [GUS84], [HAL84], [LUN84], [LAM85], [MAR85], [MIL85a], [MIL85b], [MIL85c], [GUS85b], [SCH86], [TRI86], [RIC88], [MIL88a], [DEC89] et [MIL91a], tandis que les protocoles qui se fondent sur eux sont décrits dans [MIL81a], [MIL81b], [MIL83b], [GUS85a], [MIL85c], [TRI86], [MIL88a], [DEC89] et [MIL91a]. NTP utilise des techniques qui dérivent à la fois d'eux et de méthodologies de systèmes linéaires et de consensus. Les méthodes linéaires pour la synchronisation du réseau téléphonique numérique sont récapitulées dans [LIN80], et les méthodes de consensus pour la synchronisation d'horloge sont rassemblées dans [LAM85].

Le service d'heure numérique (DTS, *Digital Time Service*) [DEC89] a beaucoup des objectifs de service de NTP. La conception de DTS insiste beaucoup sur la gestion de la configuration et sur les principes d'exactitude pour le fonctionnement dans un environnement de LAN ou grappe de LAN gérés, alors que NTP met largement l'accent sur la précision et la stabilité du service effectué dans un environnement non géré d'Internet mondial. En DTS, un sous réseau de synchronisation consiste en employés, serveurs, courriers et fournisseurs d'heure. Par rapport à la nomenclature NTP, un fournisseur d'heure est une source de référence primaire, un courrier est un serveur secondaire destiné à importer l'heure d'un ou plusieurs serveurs primaires distants pour la redistribution locale et un serveur est destiné à fournir l'heure à des nœuds d'extrémité ou employés pouvant être nombreux. A la différence de NTP, DTS n'a pas besoin, ni n'utilise, d'informations de mode ou de strate dans le choix d'horloge et n'inclut pas de dispositions pour filtrer le bruit de synchronisation, choisir la plus exacte dans un ensemble d'horloges supposées correctes ou compenser des erreurs de fréquences intrinsèques.

En fait, les dernières révisions de NTP ont adopté certaines caractéristiques de DTS afin de prendre en charge les principes de justesse. Cela inclut des mécanismes pour limiter les erreurs maximum inhérentes aux procédures de transfert de l'heure et l'utilisation d'un mécanisme à exactitude démontrable (soumis à des hypothèses établies) pour rejeter les homologues inappropriés dans les procédures de sélection d'horloge. Ces caractéristiques sont décrites à la Section 4 et à l'Appendice H du présent document.

Le protocole d'acheminement Fuzzball [MIL83b], parfois appelé Hellospeak, incorpore la synchronisation de l'heure directement dans la conception du protocole d'acheminement. Un ou plusieurs processus se synchronisent à une source de référence externe, telle qu'une horloge radio ou un démon NTP, et l'algorithme d'acheminement construit un arbre d'expansion à pondération minimum qui s'enracine dans ces processus. Les décalages d'horloge sont alors distribués le long des arcs de l'arbre d'expansion à tous les processus du système et les diverses horloges de traitement sont corrigées en utilisant la procédure décrite à la Section 5 du présent document. Alors qu'il est visible que la conception de Hellospeak a fortement influencé la conception de NTP, Hellospeak lui-même n'est pas un protocole Internet et n'est pas adapté pour une utilisation en dehors de son environnement de réseau local.

Le démon temporel Unix 4.3bsd synchronisé [GUS85a] utilise un seul démon maître d'heure pour mesurer les décalages d'un certain nombre d'hôtes esclaves et leur envoyer des corrections périodiques. Dans ce modèle, le maître est fait pour utiliser un algorithme choisi [GUS85b] conçu pour éviter les situations où aucun maître n'est choisi ou où plus d'un maître est choisi. Le processus de choix exige une capacité de diffusion, qui n'est pas une caractéristique que l'on rencontre partout sur l'Internet. Alors que ce modèle a été étendu pour prendre en charge des configurations hiérarchiques dans lesquelles un esclave sur un réseau sert de maître sur l'autre [TRI86], le modèle exige des tableaux de configuration manuels afin d'établir la hiérarchie et éviter les mises en boucle. En plus de la redondance lourde, mais qu'on peut supposer peu fréquente, du processus de choix, le processus de mesure/correction du décalage exige deux fois plus de messages que NTP par mise à jour.

Un schéma présentant des caractéristiques similaires à celles de NTP est décrit dans [KOP87]. Ce schéma est destiné aux LAN multi serveurs où chacun des serveurs d'un ensemble de serveurs horaires (qui peuvent être nombreux) détermine son décalage de temps local par rapport à chacun des autres serveurs de l'ensemble en utilisant des messages horodatés périodiques, puis détermine la correction d'horloge locale en utilisant l'algorithme Fault-Tolerant Average (FTA, *tolérance moyenne de faute*) de [LUN84]. L'algorithme FTA, qui est utile lorsque jusqu'à *k* serveurs peuvent être en faute, trie les décalages, élimine les *k* plus forts et plus faibles et moyenne le reste. Le schéma, tel que décrit dans [SCH86], est très convenable dans les environnements de LAN qui acceptent la diffusion mais produirait des redondances inacceptables dans un environnement Internet. De plus, pour des raisons données à la Section 4 du présent

document, les propriétés statistiques de l'algorithme FTA ne sont vraisemblablement pas optimales dans un environnement Internet ayant des retards très dispersés.

De nombreux travaux de recherche ont été effectués sur la question de la maintenance du temps exact dans une communauté où certaines horloges ne sont pas fiables. Une vraie chimère *(truechimer)* est une horloge qui conserve l'exactitude de l'heure sur la base d'un standard précédemment publié (et de confiance), alors qu'un faux tic-tac *(falseticker)* est une horloge qui ne le fait pas. Déterminer si une horloge est une vraie chimère ou un faux tic-tac est un intéressant problème abstrait qu'on peut aborder à l'aide des méthodes par consensus décrites dans [LAM85] et [SRI87].

Une fonction de convergence fonctionne selon les décalages entre les horloges d'un système pour accroître la précision en réduisant ou en éliminant les erreurs causées par les faux tic-tac. Il y a deux classes de fonctions de convergence, celles qui impliquent des algorithmes de convergence interactive et celles qui impliquent des algorithmes de cohérence interactive. Les algorithmes de convergence interactive utilisent des techniques de regroupement statistique telles que l'algorithme de moyenne de tolérance aux fautes de [HAL84], l'algorithme CNV de [LUN84], l'algorithme de sous ensemble de majorité de [MIL85a], l'algorithme non byzantin de [RIC88], l'algorithme égocentrique de [SCH86], l'algorithme d'intersection de [MAR85] et [DEC89] et l'algorithme de la Section 4 du présent document.

Les algorithmes de cohérence interactive sont conçus pour détecter les processus d'horloge fautifs qui pourraient indiquer des décalages grossièrement incohérents sur des lectures successives ou pour des lecteurs différents. Ces algorithmes utilisent un protocole de consensus qui implique des phases successives de lectures, qui peuvent être relayées et éventuellement augmentées de signatures numériques. Des exemples sont ceux des algorithmes de feu d'artifice de [HAL84] et l'algorithme de l'optimum de [SRI87]. Cependant, ces algorithmes requièrent un grand nombre de messages, particulièrement lorsqu'un grand nombre d'horloges sont impliquées, et ils sont conçus pour détecter des fautes qui se rencontrent rarement dans l'expérience de l'Internet. Pour ces raisons, ils ne seront pas approfondis dans le présent document.

Il n'est pas possible en pratique de discriminer les "truechimers" des "falsetickers" autrement que sur une base statistique, particulièrement avec des configurations hiérarchiques et un Internet statistiquement bruyant. Alors qu'il est possible d'encadrer les erreurs maximum dans les procédures de transfert de l'heure, en supposant des tolérances suffisamment généreuses pour les composants matériels, il en résulte généralement une faible exactitude et une médiocre stabilité. L'approche suivie par le concept de NTP et ses prédécesseurs implique des oscillateurs mutuellement couplés, une estimation de vraisemblance maximum, et des procédures de choix d'horloge, conjointement à une conception qui permet des hypothèses démontrables sur les limites d'erreur, à faire par rapport à des hypothèses établies sur l'exactitude des sources de référence primaires. Du point de vue analytique, le système d'homologues NTP distribués fonctionne comme un ensemble d'oscillateurs à verrouillage de phase couplés, avec l'algorithme de mise à jour fonctionnant comme un détecteur de phase et l'horloge locale comme un oscillateur discipliné, mais avec des limites d'erreur déterministes calculées à chaque étape du processus de transfert de l'heure.

Le choix particulier d'une procédure de mesure et de calcul de décalage décrit à la Section 3 est une variante du système de l'heure retournable utilisé dans certains réseaux téléphoniques numériques [LIN80]. Les algorithmes de filtre et de choix d'horloge sont conçus de telle sorte que le sous réseau de synchronisation d'horloge s'auto organise en une configuration hiérarchisée maître-esclave [MIT80]. Par rapport à la conservation de l'exactitude et de la stabilité, les similarités de NTP avec les systèmes de téléphonie numérique ne sont pas accidentelles car ces systèmes ont été très étudiés [LIN80], [BRA80]. Ce qui rend le modèle NTP unique est sa configuration adaptative, la consultation *(polling)*, le filtrage, les mécanismes de sélection et d'exactitude qui structurent la dynamique du système pour qu'il s'adapte à l'environnement universel de l'Internet.

2 Architecture du système

Dans le modèle NTP un certain nombre de sources de référence primaires, synchronisées par fil ou radio aux heures nationales standard, sont connectées à des ressources largement accessibles, telles que des passerelles du réseau dorsal, et fonctionnent comme serveurs horaires principaux. L'objet de NTP est de convoyer les informations de conservation du temps de ces serveurs vers les autres serveurs horaires via l'Internet et aussi de faire une vérification croisée des horloges et d'atténuer les erreurs dues aux équipements ou à des défaillances de propagation. Un certain nombre d'hôtes ou passerelles de réseau local, agissant comme serveurs horaires secondaires, font tourner NTP avec un ou plusieurs serveurs primaires. Afin de réduire les redondances du protocole, les serveurs secondaires distribuent l'heure via NTP aux hôtes de réseau local restants. Dans l'intérêt de la fiabilité, des hôtes choisis peuvent être équipés horloges radio moins précises mais moins coûteuses, utilisées pour la récupération en cas de défaillance des serveurs primaires et/ou secondaires ou des canaux de communication entre eux.

Tout au long du présent document, une nomenclature standard a été adoptée : la stabilité d'une horloge est la façon dont elle peut maintenir une fréquence constante, l'exactitude *(accuracy)* est la façon dont sa fréquence et son heure peuvent être comparées à l'heure nationale standard, et la précision est comment ces quantités peuvent être précisément maintenues dans un système particulier de conservation du temps. Sauf indication contraire, le décalage de deux horloges est la différence de temps entre elles, alors que le biais *(skew)* est la différence de fréquence (dérivée première du décalage par rapport au temps) entre elles. Les horloges réelles affichent une variation de biais (dérivée seconde du décalage par rapport au temps), qui est appelée dérive *(drift)* ; cependant dans la présente version de NTP, la dérive est supposée à zéro.

NTP est conçu pour donner trois produits : du décalage d'horloge, du temps *(ou délai)* d'aller-retour, et de la dispersion, qui se rapportent tous à une horloge de référence choisie. Le décalage d'horloge représente la quantité *(de temps)* qu'il faut apporter à l'horloge locale pour la mettre en correspondance avec l'horloge de référence. Le délai d'aller-retour donne la capacité de lancer un message pour qu'il arrive à l'horloge de référence à une heure spécifiée. La dispersion représente l'erreur maximum d'une horloge locale par rapport à l'horloge de référence. Comme la plupart des serveurs horaires hôtes vont se synchroniser via un autre serveur horaire homologue, il y a deux composants dans chacun de ces trois produits, ceux déterminés par l'homologue par rapport à la source de référence primaire de l'heure standard et ceux mesurés par l'hôte par rapport à l'homologue. Chacun de ces composants est maintenu séparément dans le protocole afin de faciliter le contrôle des erreurs et la gestion du sous réseau lui-même. Ils ne procurent pas seulement des mesures de précision des décalages et des délais, mais aussi des limites définitives d'erreur maximum, de sorte que l'interface d'utilisateur peut déterminer non seulement l'heure, mais aussi la qualité de l'heure.

Il n'y a pas, dans NTP, de dispositions pour la découverte des homologues ou la gestion des circuits virtuels. L'intégrité des données est assurée par les sommes de contrôle de IP et de UDP. Aucun contrôle de flux ni facilité de retransmission n'est fourni ni n'est nécessaire. La duplication de détection est inhérente aux algorithmes de traitement. Le service peut fonctionner dans un mode symétrique, dans lequel les serveurs et les clients sont indistinguables, et pourtant maintenir une petite quantité d'informations d'état, ou en mode client/serveur, dans lequel les serveurs n'ont besoin de maintenir aucun état autre que celui contenu dans la demande du client. Il n'est inclus une légère capacité d'association-gestion, incluant l'accessibilité dynamique et des mécanismes de taux d'interrogation variables, que pour gérer les informations d'état et réduire les exigences de ressource. Comme un seul format de message NTP est utilisé, le protocole est facilement mis en œuvre et peut être utilisé dans divers mécanismes de consultation sollicités ou non sollicités.

Il faut reconnaître que la synchronisation d'horloge exige par nature de longues périodes et de nombreuses comparaisons afin de maintenir une exacte conservation de l'heure. Alors que seulement quelques mesures sont habituellement suffisantes pour déterminer de façon fiable l'heure locale à une seconde près, des périodes de nombreuses heures et des dizaines de mesures sont nécessaires pour résoudre un biais d'oscillateur et maintenir l'heure locale à une milliseconde près. Et donc, l'exactitude réalisée est directement dépendante du temps mis à la réaliser. Heureusement, la fréquence des mesures peut être assez faible et presque toujours de façon non intrusive pour les opérations normales du réseau.

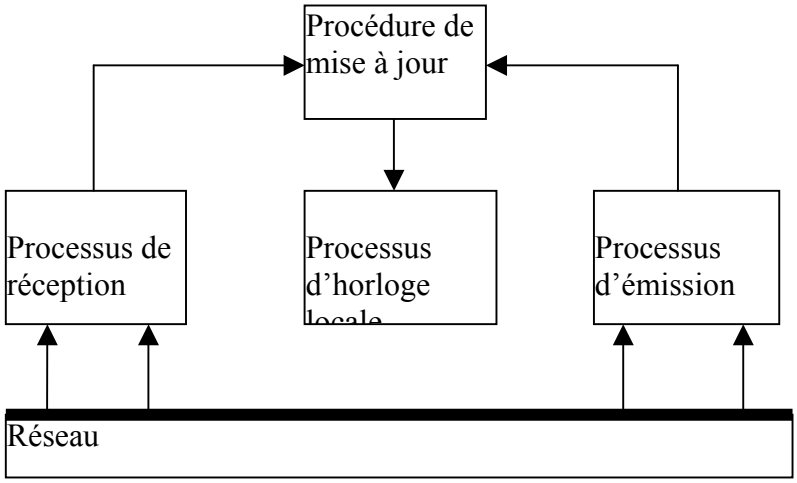

Figure 1. Modèle de mise en oeuvre

2.1 Modèle de mise en œuvre

Dans ce qui peut être le modèle client/serveur le plus courant, un client envoie un message NTP à un ou plusieurs serveurs et traite la réponse dès réception. Le serveur échange les adresses et les ports, réécrit certains champs du message, recalcule la somme de contrôle et retourne immédiatement le message. Les informations incluses dans le message NTP permettent au client de déterminer l'heure du serveur par rapport à l'heure locale et d'ajuster en conséquence l'horloge locale. De plus, le message inclut les informations pour calculer l'exactitude et la fiabilité attendues de la conservation de l'heure, ainsi que de choisir le meilleur parmi plusieurs serveurs possibles.

Alors que le modèle client/serveur peut suffire pour une utilisation sur des réseaux locaux impliquant un serveur public et peut-être de nombreuses stations de travail clientes, la pleine généralité de NTP exige une participation distribuée d'un certain nombre de client/serveurs ou homologues arrangés dans une configuration hiérarchiquement distribuée reconfigurable de façon dynamique. Elle exige aussi des algorithmes sophistiqués pour la gestion associée, la manipulation des données et le contrôle d'horloge locale. Tout au long du reste du présent document, le terme "hôte" se réfère à une instance du protocole sur un processeur local, alors que le terme "homologue" se réfère à l'instance de protocole sur un processeur distant connecté par un canal réseau.

La Figure 1 montre un modèle de mise en œuvre pour un hôte qui inclut trois processus partageant une base de données partitionnée, avec une partition dédiée à chaque homologue, et interconnectés par un système de passage de messages. Le processus de transmission, piloté par des temporisateurs indépendants pour chaque homologue, collecte les informations dans la base de données et envoie les messages NTP aux homologues. Chaque message contient l'horodatage local de l'envoi du message, ainsi que les horodatages précédemment reçus et d'autres informations nécessaires pour déterminer la hiérarchie et gérer l'association. Le débit de transmission de messages est déterminé par l'exactitude requise par l'horloge locale, ainsi que par l'exactitude de ses homologues.

Le processus de réception reçoit les messages NTP et peut-être les messages dans les autres protocoles, ainsi que les informations des horloges radio directement connectées. Lorsqu'un message NTP est reçu, le décalage entre l'horloge homologue et l'horloge locale est calculé et incorporé dans la base de données avec les autres informations utiles pour la détermination d'erreurs et le choix d'homologue. Un algorithme de filtrage décrit à la Section 4 amélioré l'exactitude en éliminant les données inférieures.

La procédure de mise à jour est initialisée par la réception d'un message mais aussi à d'autres moments. Elle traite des données de décalage provenant de chaque homologue et choisit le meilleur en utilisant les algorithmes de la Section 4. Cela peut impliquer de nombreuses observations de quelques homologues ou quelques observations de nombreux homologues, selon l'exactitude requise.

Le processus d'horloge locale fonctionne d'après les données de décalage produites par la procédure de mise à jour et elle ajuste la phase et la fréquence de l'horloge locale en utilisant les mécanismes décrits à la Section 5. Il peut en résulter un changement d'étape ou un ajustement de phase graduel de l'horloge locale pour réduire le décalage à zéro. L'horloge locale fournit une source stable d'informations horaires aux autres utilisateurs du système et sert de référence ultérieure à NTP lui-même.

2.2 Configurations réseau

Le sous réseau de synchronisation est un réseau connecté de serveurs horaires primaires et secondaires, de client et de canaux de transmission qui les interconnectent. Un serveur horaire primaire est directement synchronisé à une source de référence primaire, habituellement une horloge radio. Un serveur horaire secondaire déduit la synchronisation, éventuellement via d'autres serveurs secondaires, d'un serveur primaire sur des canaux du réseau éventuellement partagés avec d'autres services. Dans les circonstances normales, il est prévu que le sous réseau de synchronisation des serveurs primaires et secondaires suppose une configuration hiérarchisée maître-esclave avec les serveurs primaires à la racine et les serveurs secondaires d'exactitude décroissante en progressant selon les branches de l'arborescence.

Suivant les conventions établies par l'industrie du téléphone [BEL86], l'exactitude de chaque serveur est définie par un nombre appelé la strate *(stratum)*, avec un (1) alloué au niveau le plus élevé (serveurs primaires) et à chaque niveau en dessous (serveurs secondaires) dans la hiérarchie est alloué un de plus que le niveau précédant. Avec la technologie actuelle et les horloges radio disponibles, des exactitudes à échantillon unique de l'ordre de la milliseconde peuvent être réalisées à l'interface réseau d'un serveur primaire. Des exactitudes de cet ordre exigent des attentions particulières dans la conception et la mise en œuvre du système d'exploitation et du mécanisme d'horloge locale, tels que décrits à la Section 5.

Lorsque la strate augmente de un, l'exactitude d'échantillon simple se dégrade en fonction des canaux du réseau et de la stabilité des horloges locales. Afin d'éviter les calculs fastidieux [BRA80] nécessaires pour estimer les erreurs de chaque configuration spécifique, il est utile de supposer que les erreurs de mesure moyennes s'accumulent

approximativement en proportion du délai et de la dispersion mesurés par rapport à la racine du sous réseau de synchronisation. L'Appendice H contient une analyse des erreurs, y compris une déduction de l'erreur maximum en fonction du délai et de la dispersion, où cette dernière quantité dépend de la précision du système de conservation de l'heure, de la tolérance de fréquence de l'horloge locale et de divers résidus. En supposant que les serveurs primaires sont synchronisés sur l'heure standard avec une exactitude connue, cela donne une spécification déterministe fiable de l'exactitude de la conservation du temps tout au long du sous réseau de synchronisation.

En s'appuyant encore sur l'expérience de l'industrie du téléphone, qui a appris de telles leçons à de considérables dépens [ABA89], la topologie du sous réseau de synchronisation devrait être organisée de façon à produire l'exactitude la plus élaborée, sans jamais être autorisé à se mettre en boucle. Un facteur supplémentaire est que chaque incrément de strate implique un serveur horaire potentiellement non fiable qui introduit des erreurs de mesure supplémentaires. L'algorithme de choix utilisé dans NTP utilise une variante de l'algorithme d'acheminement distribué de Bellman-Ford [37] pour calculer les arbres d'expansion à pondération minimum qui s'enracinent sur les serveurs primaires. La métrique des distances utilisée par l'algorithme consiste en strates (échelonnées) plus la distance de synchronisation, qui consiste elle-même en la dispersion plus la moitié du délai absolu. Et donc, le chemin de synchronisation va toujours prendre le nombre minimum de serveurs vers la racine, avec les nœuds résolus sur la base de l'erreur maximum.

Il résulte de cette conception que le sous réseau reconfigure automatiquement dans une configuration hiérarchique maître esclave pour produire l'heure la plus exacte et la plus fiable, même lorsque un ou plusieurs serveurs primaires ou secondaires, ou les canaux réseau entre eux, subissent une défaillance. Ceci inclut le cas où tous les serveurs primaires normaux (par exemple, une horloge radio WWVB de haute précision à la plus faible distance de synchronisation) sur un sous réseau qui peut être partitionné, subissent une défaillance, mais où un ou plusieurs serveurs primaires de sauvegarde (par exemple une horloge radio WWV de moindre précision fonctionnant à une distance de synchronisation plus élevée) continue de fonctionner. Cependant, si tous les serveurs primaires du sous réseau devaient subir une défaillance, les serveurs secondaires restants se synchroniseraient entre eux tandis que les distances s'augmenteraient par paliers jusqu'à un maximum prédéterminé infini du fait des propriétés bien connues de l'algorithme de Bellman-Ford. En atteignant le maximum sur tous les chemins, un serveur va abandonner le sous réseau et courir librement en utilisant ses dernières heure et fréquence déterminées. Comme ces calculs sont supposés être très précis, particulièrement en fréquence, même une période d'interruption longue peut avoir pour résultat des erreurs de conservation de l'heure non supérieures à quelques millisecondes par jour avec des oscillateurs stabilisés de façon appropriée (voir à la Section 5).

Dans le cas de plusieurs serveurs primaires, le calcul de l'arbre d'expansion va normalement choisir le serveur se trouvant à la distance de synchronisation minimum. Cependant, lorsque ces serveurs sont approximativement à la même distance, le calcul peut avoir pour résultat des choix aléatoires parmi ceux-ci à cause des délais normaux de dispersion. Ordinairement, cela ne dégrade pas l'exactitude tant que toute distorsion entre les serveurs primaires est petite comparée à la distance de synchronisation. Si elle ne l'est pas, les algorithmes de filtre et de sélection vont choisir le meilleur des serveurs disponibles et évacuer les valeurs divergentes comme prévu.

3 Le protocole de l'heure du réseau (NTP)

La présente section consiste en une définition formelle du protocole de l'heure du réseau, incluant ses formats de données, ses entités, ses variables d'état, ses événements et ses procédures de traitement d'événement. La spécification se fonde sur le modèle de mise en œuvre illustré à la Figure 1, mais elle n'est pas destinée à ce que ce modèle soit le seul sur lequel une spécification puisse se fonder. En particulier, la spécification est destinée à illustrer et préciser les opérations intrinsèques de NTP, ainsi qu'à servir de fondement à une spécification plus rigoureuse, plus complète et plus vérifiable.

3.1 Formats de données

Toutes les opérations mathématiques exprimées ou impliquées ici sont en arithmétique en complément à deux, à virgule fixe. Les données sont spécifiées comme des quantités d'entiers ou à virgule fixe, avec les positions de bits dénombrées à la mode gros boutien à partir de zéro en commençant à gauche, ou d'ordre élevé. Comme diverses mises en œuvre peuvent étalonner des quantités déduites de sources externes pour un usage interne, ni la précision ni le placement de la virgule pour les quantités à virgule fixe ne sont spécifiées. Sauf spécification contraire, toutes les quantités sont arithmétiques *(non signées)* et peuvent occuper la largeur de champ complète avec un zéro implicite précédant le bit zéro. Les paquetages de matériel et logiciel conçus pour fonctionner avec des quantités algébriques *(signed)* donneront des résultats étonnants lorsque le bit de plus fort poids (le signe) est établi. Il est suggéré que les quantités non signées déduites de l'extérieur avec virgule fixe, telles que les horodatages, soient décalées à droite d'un bit pour utilisation interne, car la précision représentée par la largeur de champ complète est rarement justifiée.

Comme les horodatages NTP sont des données précieuses, et en fait, représentent le principal produit du protocole, un format d'horodatage spécial a été établi. Les horodatages NTP sont représentés comme des nombres non signés (arithmétiques) de 64 bits à virgule fixe, en secondes par rapport au premier janvier 1900 à 0 heure. La partie entière est dans les 32 premiers bits et la partie décimale est dans les 32 derniers bits. Ce format permet une arithmétique à précisions multiples convenable et la conversion en représentation du protocole de l'heure (en secondes), mais complique la conversion en représentation de message d'horodatage ICMP (en millisecondes). La précision de cette représentation est d'environ 200 picosecondes, ce qui devrait être adéquat pour les exigences les plus extravagantes.

Les horodatages sont déterminés en copiant la valeur en cours de l'horloge locale sur un horodatage lors d'un événement significatif, tel que l'arrivée d'un message. Afin de maintenir la plus stricte exactitude, il est important que cela soit fait aussi près que possible du pilote matériel ou logiciel associé à l'événement. En particulier, les horodatages de départ devraient être redéterminés pour chaque retransmission de niveau liaison. Dans certains cas, un horodatage particulier peut n'être pas disponible, comme lorsque l'hôte est réinitialisé ou que le protocole démarre pour la première fois. Dans ces cas, le champ de 64 bits est mis à zéro, indiquant que la valeur est invalide ou indéfinie.

Noter que depuis une certaine date en 1968, le bit de plus fort poids (le bit 0 de la partie entière) a été établi, et que le champ de 64 bits va déborder quelque part en 2036. Si NTP devait être encore utilisé en 2036, des moyens externes seraient nécessaires pour qualifier l'heure par rapport à 1900 et l'heure par rapport à 2036 (et d'autres multiples de 136 ans). Les données d'horodatage exigeant une telle qualification seront si précieuses que les moyens appropriés devraient être rendus disponibles. Il va exister un intervalle de 200 picosecondes, ignoré dorénavant, chaque 136 ans lorsque le champ de 64 bits sera à zéro et donc considéré comme invalide.

3.2 Variables et paramètres d'état

Ci-après figure un résumé des divers paramètres et variables d'état utilisés par le protocole. Ils sont répartis en classes de variables de système, qui se rapportent à l'environnement de système d'exploitation et au mécanisme d'horloge locale ; les variables d'homologues, qui représentent l'état de la machine de protocole spécifique de chaque homologue ; les variables de paquet, qui représentent le contenu du message NTP ; et les paramètres, qui représentent des constantes de configuration fixes pour toute mise en œuvre de la version en cours. Pour chaque classe, la description de la variable est suivie de son nom et de la procédure ou valeur qui la contrôle. Noter que les variables sont en minuscules, alors que les paramètres sont en majuscules. Des détails supplémentaires sur les formats et les utilisations sont présentés dans les sections et Appendices suivants.

3.2.1 Variables communes

Les variables suivantes sont communes à deux ou plus classes de système, d'homologue et de paquet. Des variables supplémentaires sont spécifiques du mécanisme facultatif d'authentification décrit à l'Appendice C. Lorsqu'il est nécessaire de distinguer entre variables communes du même nom, on utilisera l'identifiant de variable.

Adresse d'homologue (peer.peeraddr, pkt.peeraddr), Port d'homologue (peer.peerport, pkt.peerport) : ce sont l'adresse Internet de 32 bits et le numéro de port de 16 bits de l'homologue.

Adresse d'hôte (peer.hostaddr, pkt.hostaddr), Port d'hôte (peer.hostport, pkt.hostport) : Ce sont l'adresse Internet de 32 bits et le numéro de port de 16 bits de l'hôte. Ils sont inclus parmi les variables d'état pour prendre en charge le multi rattachement.

Indicateur de saut (sys.leap, peer.leap, pkt.leap) : C'est un code d'avertissement de deux bits d'un saut de seconde à venir à insérer dans l'échelle de temps NTP. Les bits sont mis avant 23:59 le jour de l'insertion et remis à zéro après 00:00 le jour suivant. Cela provoque l'augmentation ou la diminution d'un du nombre de secondes (intervalle de débordement) le jour de l'insertion. Dans le cas de serveurs primaires, les bits sont établis par l'intervention d'un opérateur, alors que dans le cas de serveurs secondaires, les bits sont établis par le protocole. Les deux bits, bit 0 et bit 1, sont respectivement codés comme suit :

00	pas d'avertissement
01	la dernière minute a 61 secondes
10	la dernière minute a 59 secondes
11	condition d'alarme (horloge non synchronisée)

Sauf en condition d'alarme (11_2), NTP lui –même ne fait rien de ces bits, sauf les passer aux routines de conversion d'heure qui ne font pas partie de NTP. La condition d'alarme survient lorsque, pour une raison quelconque, l'horloge locale n'est pas synchronisée, comme lors d'une première mise en route ou après une longue période lorsque aucune source de référence primaire n'est disponible.

Mode (peer.mode, pkt.mode) : Entier indiquant le mode d'association, avec des valeurs codées comme suit :

0 non spécifié
1 symétrie active
2 symétrie passive
3 client
4 serveur
5 diffusion
6 réservé pour messages de contrôle NTP
7 réservé pour utilisation privée

Strate (sys.stratum, peer.stratum, pkt.stratum) : C'est un entier indiquant la strate de l'horloge locale, avec des valeurs définies comme suit :

0 non spécifié
1 référence primaire (par exemple, horloge atomique calibrée, horloge radio)
2 à 255 référence secondaire (via NTP)

Pour les comparaisons une valeur de zéro est considérée comme plus grande que toute autre valeur. Noter que la valeur maximum de l'entier codé comme une variable de paquet est limitée par le paramètre NTP.MAXSTRATUM.

Intervalle de consultation (sys.poll, peer.hostpoll, peer.peerpoll, pkt.poll) : C'est un entier signé qui indique l'intervalle minimum entre les messages transmis, en secondes comme une puissance de deux. Par exemple, une valeur de six indique un intervalle minimum de 64 secondes.

Précision (sys.precision, peer.precision, pkt.precision) : C'est un entier signé qui indique la précision des diverses horloges, en secondes à la plus proche puissance de deux. La valeur doit être arrondie à la plus grande puissance de deux suivante, par exemple, on allouera la valeur -5 (31,25 ms) à une horloge de puissance-fréquence 50 Hz (20 ms) ou 60 Hz (16,67 ms), alors qu'on allouera la valeur -9 (1,95 ms) à une horloge à contrôle cristal de 1000 Hz (1 ms).

Délai de racine (sys.rootdelay, peer.rootdelay, pkt.rootdelay) : C'est un nombre algébrique à virgule fixe indiquant le délai d'aller-retour total de la source de référence primaire à la racine du sous réseau de synchronisation, en secondes. Noter que cette variable peut prendre des valeurs positives et négatives, selon la précision et le biais de l'horloge.

Dispersion de racine (sys.rootdispersion, peer.rootdispersion, pkt.rootdispersion) : C'est un nombre algébrique à virgule fixe indiquant l'erreur maximum par rapport à la source de référence primaire à la racine du sous réseau de synchronisation, en secondes. Seules les valeurs positives supérieures à zéro sont possibles.

Identifiant d'horloge de référence (sys.refid, peer.refid, pkt.refid) : C'est un code de 32 bits qui identifie l'horloge de référence particulière. Dans le cas de strate 0 (non spécifiée) ou strate 1 (source de référence primaire), c'est une chaîne ASCII de quatre octets, justifiée à gauche, bourrée avec des zéros, par exemple (voir la liste complète à l'Appendice A) :

Strate	**Code**	**Signification**
0	DCN	protocole d'acheminement DCN
0	TSP	protocole d'heure TSP
1	ATOM	horloge atomique (calibrée)
1	WWVB	radio LF WWVB (bande 5)
1	GOES	satellite UHF GOES (bande 9)
1	WWV	radio HF WWV (bande 7)

Dans le cas de la strate 2 et au-dessus (référence secondaire) c'est l'adresse Internet de quatre octets de l'homologue choisi pour la synchronisation.

Variables de système	**Nom**	**Procédure**
Indicateur de saut	sys.leap	Mise à jour d'horloge
Strate	sys.stratum	Mise à jour d'horloge
Précision	sys.precision	système
Délai racine	sys.rootdelay	Mise à jour d'horloge
Dispersion racine	sys.rootdispersion	Mise à jour d'horloge
Identifiant d'horloge de référence	sys.refid	Mise à jour d'horloge
Horodatage de référence	sys.reftime	Mise à jour d'horloge
Horloge locale	sys.clock	Mise à jour d'horloge
Source d'horloge	sys.peer	sélection
Intervalle de consultation	sys.poll	horloge locale

Clés cryptographiques	sys.keys	authentification

Tableau 1. Variables système

Horodatage de référence (sys.reftime, peer.reftime, pkt.reftime) : C'est l'heure locale, en format d'horodatage, lorsque l'horloge locale a été mise à jour pour la dernière fois. Si l'horloge locale n'a jamais été synchronisée, la valeur est zéro.

Horodatage d'origine (peer.org, pkt.org) : C'est l'heure locale, en format d'horodatage, à l'homologue, lorsque son dernier message NTP a été envoyé. Si l'homologue devient injoignable, la valeur est réglée à zéro.

Horodatage reçu (peer.rec, pkt.rec) : C'est l'heure locale, en format d'horodatage, d'arrivée du dernier message NTP. Si l'homologue devient injoignable, la valeur est réglée à zéro.

Horodatage d'émission (peer.xmt, pkt.xmt) : C'est l'heure locale, en format d'horodatage, à laquelle le message NTP est parti de chez l'expéditeur.

3.2.2 Variables de système

Le Tableau 1 montre l'ensemble complet des variables de système. En plus des variables communes décrites précédemment, les variables suivantes sont utilisées par le système d'exploitation afin de synchroniser l'horloge locale.

Horloge locale (sys.clock) : C'est l'heure locale en cours, en format d'horodatage. L'heure locale est déduite de l'horloge matérielle de la machine particulière et s'incrémente à des intervalles qui dépendent de la conception utilisée. Une conception appropriée, comprenant des mécanismes de biaisage et de compensation de biais, est décrite à la Section 5.

Source d'horloge (sys.peer) : C'est un sélecteur qui identifie la source de synchronisation en cours. Ce sera habituellement un pointeur sur une structure contenant les variables d'homologue. La valeur spéciale NULL indique qu'il n'y a pas de source de synchronisation valide en cours.

3.2.3 Variables d'homologue

Le Tableau 2 montre l'ensemble complet de variables d'homologue. En plus des variables communes décrites précédemment, les variables suivantes sont utilisées par les fonctions de gestion et de mesure d'homologue.

Bit configuré (peer.config) : C'est un bit qui indique que l'association a été créée à partir d'informations de configuration qu'elle ne devrait pas être dissoute si l'homologue devient inatteignable.

Mise à jour d'horodatage (peer.update) : C'est l'heure locale, en format d'horodatage, à laquelle le plus récent message NTP a été reçu. Elle est utilisée pour calculer la dispersion du biais.

Registre d'atteignabilité (peer.reach) : C'est un registre à décalage de NTP.WINDOW bits utilisé pour déterminer l'état d'atteignabilité de l'homologue, avec les bits entrant de l'extrémité de plus faible poids (la plus à droite). Un homologue est considéré comme atteignable si au moins un bit dans ce registre est mis à un.

Temporisateur d'homologue (peer.timer) : C'est un compteur d'entiers utilisé pour contrôler l'intervalle entre les messages NTP transmis. Une fois réglé à une valeur différente de zéro, le compteur décrémente à intervalles d'une seconde jusqu'à atteindre zéro, moment où la procédure d'émission est appelée. Noter que le fonctionnement de ce temporisateur est indépendant des mises à jour d'horloge locale, ce qui implique que le système de conservation de l'heure et l'architecture du système d'intervalle de temporisation doivent être indépendants l'un de l'autre.

Variables d'homologue	**Nom**	**Procédure**
Bit configuré	peer.config	initialisation
Adresse d'homologue	peer.peeraddress	réception
Port d'homologue	peer.peerport	réception
Adresse d'hôte	peer.hostaddress	réception
Port d'hôte	peer.hostport	réception
Indicateur de saut	peer.leap	paquet
Mode	peer.mode	paquet
Strate	peer.stratum	paquet
Intervalle de consultation d'homologue	peer.peerpoll	paquet
Intervalle de consultation d'hôte	peer.hostpoll	mise à jour de consultation
Précision	peer.precision	paquet

Délai racine	peer.rootdelay	paquet
Dispersion racine	peer.rootdispersion	paquet
Identifiant d'horloge de référence	peer.refid	paquet
Horodatage de référence	peer.reftime	paquet
Horodatage originel	peer.org	paquet, libération
Horodatage de réception	peer.rec	paquet, libération
Horodatage d'émission	peer.xmt	émission, libération
Mise à jour d'horodatage	peer.update	filtre, libération
Registre d'atteignabilité	peer.reach	paquet, émission, libération
Temporisateur d'homologue	peer.timer	réception, émission
Registre de filtre	peer.filter	mise à jour de consultation
Compteur de données valides	peer.valid	filtre
Délai	peer.delay	émission
Décalage	peer.offset	filtre
Dispersion	peer.dispersion	filtre
Bit authentifié activé	peer.authenable	filtre
Bit authentifié	peer.authentic	authentification
Identifiant de clé d'hôte	peer.hostkeyid	authentification
Identifiant de clé d'homologue	peer.peerkeyid	authentification

Tableau 2. Variables d'homologue

3.2.4 Variables de paquet

Variables de paquet	Nom	Procédure
Adresse d'homologue	pkt.peeraddress	émission
Port d'homologue	pkt.peerport	émission
Adresse d'hôte	pkt.hostaddress	émission
Port d'hôte	pkt.hostport	émission
Indicateur de saut	pkt.leap	émission
Numéro de version	pkt.version	émission
Mode	pkt.mode	émission
Strate	pkt.stratum	émission
Intervalle de consultation	pkt.poll	émission
Précision	pkt.precision	émission
Délai racine	pkt.rootdelay	émission
Dispersion racine	pkt.rootdispersion	émission
Identifiant d'horloge de référence	pkt.refid	émission
Horodatage de référence	pkt.reftime	émission
Horodatage originel	pkt.org	émission
Horodatage de réception	pkt.rec	émission
Horodatage d'émission	pkt.xmt	émission
Identifiant de clé	pkt.keyid	authentification
Somme de contrôle cryptée	pkt.check	authentification

Tableau 3. Variables de paquet

Le Tableau 3 montre l'ensemble complet des variables de paquet. En plus des variables communes décrites précédemment, les variables suivantes sont définies.

Numéro de version (pkt.version) : C'est un entier qui indique le numéro de version de l'expéditeur. Les messages NTP seront toujours envoyés avec le numéro de version NTP.VERSION en cours et seront toujours acceptés si le numéro de version correspond à NTP.VERSION. Des exceptions peuvent être acceptées au cas par cas lorsque le numéro de version a changé. Des lignes directrices spécifiques pour l'interfonctionnement entre la présente version et les versions précédentes de NTP sont exposées à l'Appendice D.

3.2.5 Variables de filtre d'horloge

Lorsqu'on utilise les algorithmes de filtre et de sélection suggérés à la Section 4, les variables d'état suivantes sont définies en plus des variables décrites précédemment.

Registre de filtre (peer.filter) : C'est un registre à décalage d'étapes NTP.SHIFT, où chaque étape emmagasine un triplet comportant le délai mesuré, le décalage mesuré et la dispersion calculée associés à une seule observation. Ces triplets entrent à partir l'extrémité droite de plus fort poids (le plus à gauche) et sont déplacé vers l'extrémité de moindre poids (la plus à droite) et finalement éliminés lorsque arrivent de nouvelles observations.

Compteur de données valides (peer.valid) : C'est un compteur d'entiers qui indique les échantillons valides restant dans le registre de filtre. Il sert à déterminer l'état d'atteignabilité et quand l'intervalle de consultation devrait être augmenté ou diminué.

Décalage (peer.offset) : C'est un nombre algébrique à virgule fixe qui indique le décalage de l'horloge homologue par rapport à l'horloge locale, en secondes.

Délai (peer.delay) : C'est un nombre algébrique à virgule fixe qui indique le délai d'aller-retour de l'horloge homologue par rapport à l'horloge locale sur le chemin réseau entre elles, en secondes. Noter que cette variable peut prendre des valeurs positives et négatives, selon la précision de l'horloge et l'accumulation d'erreurs de biais.

Dispersion (peer.dispersion) : C'est un nombre algébrique à virgule fixe qui indique l'erreur maximum de l'horloge homologue par rapport à l'horloge locale sur le chemin réseau entre elles, en secondes. Seules les valeurs positives supérieures à zéro sont possibles.

3.2.6 Variables d'authentification

Lorsqu'on utilise le mécanisme d'authentification suggéré à l'Appendice C, les variables d'état suivantes sont définies en plus des variables décrites précédemment. Ces variables ne servent que si le mécanisme facultatif d'authentification décrit à l'Appendice C est mis en œuvre.

Bit d'authentification activée (peer.authenable) : C'est un bit qui indique que l'association va fonctionner dans le mode authentifié.

Bit authentifié (peer.authentic) : C'est un bit qui indique que le dernier message reçu de l'homologue a été correctement authentifié.

Identifiant de clé (peer.hostkeyid, peer.peerkeyid, pkt.keyid) : C'est un entier qui identifie la clé cryptographique utilisée pour générer le code d'authentification du message.

Clés cryptographiques (sys.key) : C'est un ensemble de clés DES de 64 bits. Chaque clé est construite comme dans les distributions Unix Berkeley, qui consistent en huit octets, où les sept bits de plus faible poids de chaque octet correspondent aux bits 1 à 7 de DES et le bit de plus fort poids correspond au bit 8 de parité impair de DES.

Crypto-Checksum (pkt.check) : C'est une somme de contrôle cryptée calculée par la procédure de chiffrement.

3.2.7 Paramètres

Le Tableau 4 montre les paramètres supposés pour toute mise en œuvre fonctionnant dans le système Internet. Il est nécessaire de s'accorder sur les valeurs de ces paramètres afin d'éviter des redondances inutiles dans les réseaux et des associations stables d'homologues. Les paramètres suivants sont supposés fixés et applicables à toutes les associations.

Numéro de version (NTP.VERSION) : C'est le numéro (3) de version NTP en cours.

Port NTP (NTP.PORT) : C'est le numéro de port (123) alloué par l'Autorité d'allocation des numéros de l'Internet à NTP.

Strate maximum (NTP.MAXSTRATUM) : C'est la valeur de strate maximum qui peut être codée comme variable de paquet, aussi interprétée comme <169>infini<170> ou inatteignable par l'algorithme d'acheminement de sous réseau.

Age maximum d'horloge (NTP.MAXAGE) : C'est l'intervalle maximum pendant lequel une horloge de référence sera considéré comme valide après sa dernière mise à jour, en secondes.

Biais maximum (NTP.MAXSKEW) : C'est l'erreur de décalage maximum due au biais de l'horloge locale sur l'intervalle déterminé par NTP.MAXAGE, en secondes. Le ratio $\varphi = \frac{NTP.MAXSKEW}{NTP.MAXDISTANCE}$ est interprété comme taux de biais maximum possible dû à toute cause.

Distance maximum (NTP.MAXDISTANCE) : Lorsque l'algorithme de choix suggéré à la Section 4 est utilisé, c'est la distance maximum de synchronisation pour les homologues acceptables pour la synchronisation.

Paramètres	Nom	Valeur
Numéro de version	NTP.VERSION	3
Port NTP	NTP.PORT	123
Strate maximum	NTP.MAXSTRATUM	15
Age d'horloge maximum	NTP.MAXAGE	86 400 S
Biais maximum	NTP.MAXSKEW	1 s
Distance maximum	NTP.MAXDISTANCE	1 s
Intervalle de consultation minimum	NTP.MINPOLL	6 (64 s)
Intervalle de consultation maximum	NTP.MAXPOLL	10 (1024 s)
Minimum d'horloges choisies	NTP.MINCLOCK	1
Maximum d'horloges choisies	NTP.MAXCLOCK	10
Dispersion minimum	NTP.MINDISPERSE	0,1 s
Dispersion maximum	NTP.MAXDISPERSE	16 s
Taille de registre d'atteignabilité	NTP.WINDOW	8
Taille de filtre	NTP.SHIFT	8
Pondération de filtre	NTP.FILTER	0,5
Pondération choisie	NTP.SELECT	0,75

Tableau 4. Paramètres

Intervalle minimum de consultation (NTP.MINPOLL) : C'est l'intervalle minimum de consultation permis par tout homologue dans le système Internet, en secondes, données en puissance de deux.

Intervalle maximum de consultation (NTP.MAXPOLL) : C'est l'intervalle maximum de consultation permis par tout homologue dans le système Internet, en secondes, données en puissance de deux.

Minimum d'horloges choisies (NTP.MINCLOCK) : Lorsque l'algorithme de choix suggéré à la Section 4 est utilisé, c'est le nombre minimum d'homologues acceptables pour la synchronisation.

Maximum d'horloges choisies (NTP.MAXCLOCK) : Lorsque l'algorithme de choix suggéré à la Section 4 est utilisé, c'est le nombre maximum d'homologues considérés pour le choix.

Dispersion minimum (NTP.MINDISPERSE) : Lorsque l'algorithme de filtre suggéré à la Section 4 est utilisé, c'est l'incrément de dispersion minimum pour chaque niveau de strate, en secondes.

Dispersion Maximum (NTP.MAXDISPERSE) : Lorsque l'algorithme de filtre suggéré à la Section 4 est utilisé, c'est la dispersion d'homologue maximum et la dispersion supposée pour les données manquantes, en secondes.

Taille du registre d'atteignabilité (NTP.WINDOW) : C'est la taille du registre d'atteignabilité (peer.reach), en bits.

Taille de filtre (NTP.SHIFT) : Lorsque l'algorithme de filtre suggéré à la Section 4 est utilisé, c'est la taille du registre à décalage de filtre d'horloge (peer.filter), en étapes.

Pondération de filtre (NTP.FILTER) : Lorsque l'algorithme de filtre suggéré à la Section 4 est utilisé, c'est la pondération utilisée pour calculer la dispersion du filtre.

Pondération du choix (NTP.SELECT) : Lorsque l'algorithme de choix suggéré à la Section 4 est utilisé, c'est la pondération utilisée pour calculer la dispersion du choix.

3.3 Modes de fonctionnement

Excepté en mode diffusion, une association NTP est formée lorsque deux homologues échangent des messages et qu'un des deux ou les deux créent et maintiennent une instance de la machine de protocole, appelée une association. L'association peut fonctionner dans un des cinq modes indiqués par la variable mode d'hôte (peer.mode) : actif symétrique, passif symétrique, client, serveur et diffusion, qui sont définis comme suit :

Actif symétrique (1) : Un hôte fonctionnant dans ce mode envoie des messages périodiques sans considérer l'état d'atteignabilité ou la strate de son homologue. En fonctionnant dans ce mode, l'hôte annonce sa volonté de se synchroniser avec et d'être synchronisé par l'homologue.

Passif symétrique (2) : Ce type d'association est ordinairement créé à l'arrivée d'un message provenant d'un homologue qui fonctionne dans le mode actif symétrique et ne persiste qu'aussi longtemps que l'homologue est atteignable et fonctionne à un niveau de strate inférieur ou égal à celui de l'hôte ; autrement, l'association est dissoute. Cependant,

l'association persistera jusqu'à ce qu'au moins un message ait été envoyé en réponse. En fonctionnant dans ce mode, l'hôte annonce sa volonté de se synchroniser avec et d'être synchronisé par l'homologue.

Client (3) : Un hôte fonctionnant dans ce mode envoie des messages périodiques sans considération de l'état d'atteignabilité ou de la strate de son homologue. En fonctionnant dans ce mode, l'hôte, qui est généralement une station de travail de LAN, annonce sa volonté d'être synchronisé par l'homologue mais pas de le synchroniser.

Serveur (4) : Ce type d'association est ordinairement créée à l'arrivée d'un message de demande client et n'existe qu'afin de répondre à cette demande, après quoi l'association est dissoute. En fonctionnant dans ce mode, l'hôte, habituellement un serveur horaire de LAN, annonce sa volonté de synchroniser l'homologue, mais pas d'être synchronisé par lui.

Diffusion (5) : Un hôte fonctionnant dans ce mode envoie des messages périodiques sans considération de l'état d'atteignabilité ou de la strate des homologues. En fonctionnant dans ce mode, l'hôte, généralement un serveur horaire de LAN fonctionnant sur un support de diffusion à grande vitesse annonce sa volonté de synchroniser tous les homologues, mais pas d'être synchronisé par l'un d'eux.

Un hôte fonctionnant en mode client envoie occasionnellement un message NTP à un hôte fonctionnant en mode serveur, peut-être juste après réinitialisation et à des intervalles périodiques ensuite. Le serveur répond simplement en échangeant les adresses et les ports, en fournissant les informations demandées et en retournant le message au client. Les serveurs n'ont besoin de retenir aucune information d'état entre les demandes des clients, alors que les clients sont libres de gérer les intervalles entre les envois de messages NTP pour s'adapter aux conditions locales. Dans ces modes, la machine de protocole décrite dans le présent document peut être considérablement simplifiée en un simple mécanisme distant de procédure d'appel sans perte significative d'exactitude ou robustesse, spécialement pour fonctionner sur des LAN à grande vitesse.

Dans les modes symétriques, la distinction client/serveur disparaît (presque). Le mode passif symétrique est destiné à servir aux serveurs horaires fonctionnant près des nœuds racines (strate la plus basse) du sous réseau de synchronisation et avec un assez grand nombre d'homologues, sur une base intermittente. Dans ce mode, l'identité de l'homologue n'a pas besoin d'être connue à l'avance, car l'association avec sa variable d'état n'est créée que lorsque arrive un message NTP. De plus, la mémorisation d'état peut être réutilisée lorsque l'homologue devient inatteignable ou fonctionne dans un niveau de strate supérieur et devient donc inéligible comme source de synchronisation.

Le mode symétrique actif est destiné à être utilisé par les serveurs horaires qui fonctionnent près des nœuds d'extrémité (strate la plus élevée) du sous réseau de synchronisation. Un service d'heure fiable peur habituellement être maintenu avec deux homologues au niveau de strate inférieur suivant et un homologue au même niveau de strate, de sorte que le taux de consultations en cours n'est habituellement pas significatif, même lorsque la connectivité est perdue et que des messages d'erreur sont retournés pour chaque consultation.

Normalement, un homologue fonctionne dans un mode actif (modes actif symétrique, client ou diffusion) comme configuré par un fichier de démarrage, alors que l'autre fonctionne en mode passif (modes passif symétrique ou serveur), souvent sans configuration préalable. Cependant, les deux homologues peuvent être configurés pour fonctionner dans le mode actif symétrique. Une condition d'erreur survient lorsque deux homologues fonctionnent dans le même mode, mais pas le mode actif symétrique. Dans de tels cas, chaque homologue va ignorer les messages provenant de l'autre, de sorte que les associations antérieures, s'il en est, seront démobilisées du fait de l'échec d'atteignabilité.

Le mode diffusion est destiné au fonctionnement de LAN à haute vitesse avec de nombreuses stations de travail et où les plus grandes exactitudes ne sont pas nécessaires. Dans le scénario normal, un ou plusieurs serveurs horaires du LAN envoient des diffusions périodiques aux stations de travail, qui déterminent alors l'heure sur la base d'un délai de latence pré configuré de l'ordre de quelques millisecondes. Comme dans les modes client/serveur, la machine de protocole peut être considérablement simplifiée dans ce mode ; cependant, une forme modifiée de l'algorithme de choix d'horloge peut se révéler utile dans les cas où plusieurs serveurs horaires sont utilisés pour une fiabilité améliorée.

3.4 Traitement d'événement

Les événements significatifs intéressants dans NTP surviennent à expiration d'un temporisateur homologue (peer.timer), dont l'un est dédié à chaque homologue avec une association active, et à l'arrivée d'un message NTP provenant des divers homologues. Un événement peut aussi survenir par suite d'une commande d'opérateur ou de la détection d'une faute du système, telle qu'une défaillance de source de référence primaire. La présente section décrit les procédures invoquées lorsque survient un de ces événements.

3.4.1 Conventions de notation

Les algorithmes de filtrage et de choix NTP agissent à partir d'un ensemble de variables pour le décalage d'horloge (θ, Θ), le délai d'aller-retour (δ, Δ) et la dispersion (ε, E). Lorsque il est nécessaire de distinguer entre elles, on se sert des lettres grecques minuscules pour les variables relatives à un homologue, alors que les lettres grecques majuscules sont utilisées pour les variables relatives à la ou les sources de référence primaires, c'est-à-dire, via l'homologue à la racine du sous réseau de synchronisation. Les indices seront utilisés pour identifier les homologues particuliers lorsque le contexte n'est pas clair. Les algorithmes se fondent sur une quantité appelée la distance de synchronisation (λ, Λ), qui est calculée à partir du délai d'aller-retour et de la dispersion comme décrit ci-dessous.

Comme décrit à l'Appendice H, la dispersion d'homologue ε inclut des contributions dues à l'erreur de mesure $\rho = 1 << \text{sys.precision}$, à l'accumulation d'erreur de biais φτ, où $\varphi = \frac{NTP.MAXSKEW}{NTP.MAXAGE}$ est le taux de biais maximum et $\tau = \text{sys.clock} - \text{peer.update}$ est l'intervalle depuis la dernière mise à jour, et la dispersion de filtre (d'échantillon) ε_σ calculée par l'algorithme de filtre d'horloge. La dispersion de racine E inclut des contributions dues à la dispersion des homologues choisis ε et l'accumulation d'erreur de biais φτ, conjointement avec la dispersion de racine pour l'homologue lui-même. La dispersion du système inclut la dispersion choisie (d'échantillon) εξ calculée par algorithme de choix d'horloge et le décalage d'horloge initial absolu |Θ| fourni par l'algorithme d'horloge locale. ε et E sont tous deux des quantités dynamiques, car ils dépendent du temps écoulé τ depuis la dernière mise à jour, ainsi que des dispersions d'échantillons calculées par les algorithmes.

Chaque fois que les variables d'homologue pertinentes sont mises à jour, toutes les dispersions associées à cet homologue sont mises à jour pour refléter l'accumulation d'erreur de biais. Les calculs peuvent être résumés comme suit :

$$\Theta \equiv \text{peer.offset} ,$$
$$\delta \equiv \text{peer.delay} ,$$
$$\varepsilon \equiv \text{peer.dispersion} = \rho + \varphi\tau + \varepsilon\sigma ,$$
$$\lambda \equiv \varepsilon + \frac{|\delta|}{2} ,$$

où τ est l'intervalle depuis que l'horodatage originel (d'après lequel θ et δ ont été déterminés) a été transmis à l'heure présente et εσ est la dispersion de filtre (voir ci-dessous la procédure de filtre d'horloge). Les variables relatives à la racine du sous réseau de synchronisation via l'homologue *i* sont déterminées comme suit :

$$\Theta_i \equiv \theta_i ,$$
$$\Delta_i \equiv \text{peer.rootdelay} + \delta_i ,$$
$$E_i \equiv \text{peer.rootdispersion} + \varepsilon_i + \varphi\tau_i ,$$
$$\Lambda_i \equiv E_i + \frac{|\Delta_i|}{2} ,$$

où toutes les variables sont comprises comme appartenant au $i^{ème}$ homologue. Finalement, en supposant que le $i^{ème}$ homologue est choisi pour la synchronisation, les variables du système sont déterminées comme suit :

$$\Theta = \text{décalage final combiné},$$
$$\Delta = \Delta_i ,$$
$$E = E_i + \varepsilon\xi + |\Theta| ,$$
$$\Lambda = \Lambda_i ,$$

où εξ est la dispersion choisie (voir ci-dessous la procédure de choix d'horloge).

Le pseudo-code informel qui effectue ces calculs est présenté ci-dessous. Noter que le pseudo-code n'est représenté dans aucun langage particulier, bien qu'il présente de nombreuses similitudes avec le langage C. Des détails spécifiques sur les algorithmes importants sont illustrés dans les routines du langage C à l'Appendice I.

3.4.2 Procédure d'émission

La procédure d'émission est exécutée lorsque le temporisateur homologue diminue jusqu'à zéro pour tous les modes excepté le mode client avec un serveur de diffusion, et le mode serveur dans tous les cas. En mode client avec un serveur de diffusion, les messages ne sont jamais envoyés. En mode serveur, les messages ne sont envoyés qu'en réponse aux messages reçus. Cette procédure est aussi appelée par la procédure de réception lorsqu'un message NTP arrive et qu'il n'en résulte pas une association persistante.

début de procédure d'émission

Ce qui suit initialise la mémoire tampon de paquet et copie les variables du paquet. La valeur *biais* est nécessaire pour prendre en compte l'erreur de biais accumulée sur l'intervalle depuis l'établissement de l'horloge locale.

```
pkt.peeraddr    ← peer.hostaddr ;        /* copie les variables de système et d'homologue */
pkt.peerport    ← peer.hostport ;
pkt.hostaddr    ← peer.peeraddr ;
pkt.hostport    ← peer.peerport ;
pkt.leap        ← sys.leap ;
pkt.version     ← NTP.VERSION ;
pkt.mode        ← peer.mode ;
pkt.stratum     ← sys.stratum ;
pkt.poll        ← peer.hostpoll ;
pkt.precision   ← sys.precision ;
pkt.rootdelay   ← sys.rootdelay ;
si (sys.leap = 11₂ ou (sys.clock - sys.reftime) > NTP.MAXAGE)
        biais ← NTP.MAXSKEW>;
autrement
        biais ← φ(sys.clock ← sys.reftime) ;
pkt.rootdispersion ← sys.rootdispersion + (1 << sys.precision) + biais ;
pkt.refid       ← sys.refid ;
pkt.reftime     ← sys.reftime ;
```

L'horodatage d'émission pkt.xmt sera utilisé plus tard afin de validaer la réponse ; et donc, les mises en œuvre doivent sauvegarder la valeur exacte transmise. De plus, l'ordre de copie des horodatages devrait être conçu de telle sorte que le temps de formatage et de copie des données ne dégrade pas l'exactitude.

```
pkt.org         ← peer.org ;                    /* copie des horodatages */
pkt.rec         ← peer.re ;
pkt.xmt         ← sys.clock ;
peer.xmt        ← pkt.xmt ;
```

L'invitation au chiffrement n'est mise en œuvre que si l'authentification l'est aussi. Si l'authentification est activée, le délai de chiffrement de l'authentifiant peut dégrader l'exactitude. Donc, les mises en œuvre devraient inclure une variable d'état de système (qui n'est mentionnée nulle part ailleurs dans la présente spécification) qui contienne un décalage calculé pour correspondre au délai de chiffrement attendu et corriger l'horodatage transmis comme obtenu de l'horloge locale.

```
#ifdef (authentification mise en œuvre)          /* voir l'Appendice C */
        invoquer encrypt;
        #endef
envoyer le paquet;
```

Le registre d'atteignabilité glisse d'une position vers la gauche, le bit libéré étant remplacé par zéro. Si tous les bits de ce registre sont à zéro, la procédure de libération est invoquée pour purger le filtre d'horloge et faire un nouveau choix de source de synchronisation, si nécessaire. Si l'association n'a pas été configurée par la procédure d'initialisation, l'association est dissoute.

```
peer.reach ← peer.reach << 1 ;    /* mettre à jour l'atteignabilité */
si (peer.reach = 0> et peer.config = 0) début
        dissoudre l'association;
        exit;
        endif
```

Si des données valides ont été déplacées dans le registre de filtre au moins une fois durant les deux intervalles de consultation précédents (les bits de plus faible poids de peer.reach sont mis à un), le compteur de données valides est incrémenté. L'intervalle de consultation est incrémenté après huit intervalles valides. Autrement, le compteur de données valides et l'intervalle de consultation sont tous deux décrémentés et la procédure de filtre d'horloge est invoquée avec des valeurs de zéro pour le décalage et le délai et de NTP.MAXDISPERSE pour la dispersion. La procédure de choix d'horloge est invoquée pour choisir à nouveau la source de synchronisation, si nécessaire.

```
si ( peer.reach & 6 = 0>)                    /* essai de deux bits de plus faible poids (décalés) */
        si (peer.valid < NTP.SHIFT>)         /* données valides reçues */
            peer.valid ← peer.valid + 1 ;
            autrement peer.hostpoll ← peer.hostpoll + 1 ;
autrement début
        peer.valid ← peer.valid - 1 ;        /* rien entendu */
        peer.hostpoll← peer.hostpoll – 1 ) ;
        invoquer clock-filter(0, 0, NTP.MAXDISPERSE);
        invoquer clock-select;               /* choisir la source d'horloge*/
        endif
invoquer la mise à jour de consultation;
fin de la procédure d'émission ;
```

3.4.3 Procédure de réception

La procédure de réception est exécutée à l'arrivée d'un message NTP. Elle valide le message, interprète les divers modes et invoque les autres procédures pour filtrer les données et choisir la source de synchronisation. Si le numéro de version dans le paquet ne correspond pas à la version en cours, le message peut être éliminé, cependant, des exceptions peuvent être recommandées au cas par cas lorsque la version est changée. Si les messages de contrôle NTP décrits dans l'Appendice B sont mis en œuvre et si le mode paquet est 6 (contrôle), la procédure de message de contrôle est invoquée. Les adresses et port de source et de destination Internet dans les en-têtes IP et UDP sont confrontés à l'homologue correct. S'il n'y a pas de correspondance, il est créé une nouvelle instance de machine de protocole et l'association est mobilisée.

```
début de procédure de réception
        si (pkt.version ≠ NTP.VERSION>) exit;
        #ifdef (messages de contrôle mis en œuvre)
            si (pkt.mode = 6) invoquer le message de contrôle;
            #endef
        pour (toutes associations)           /* le contrôle d'accès vient ici */
            correspondance d'adresses et ports avec les associations;
        si (pas de correspondance d'association)
            invoquer procédure d'instance de réception;   /* créer une association */
```

L'invitation à déchiffrer n'est mise en œuvre que si l'authentification est mise en oeuvre.

```
        #ifdef (authentification mise en œuvre)          /* voir l'Appendice C */
            invoquer déchiffrer;
            #endef
```

Si le mode paquet est différent de zéro, cela devient la valeur de *mode* utilisé dans l'étape suivante ; autrement, l'homologue est une vieille version de NTP et *mode* est déterminé à partir des numéros de port comme décrit au paragraphe 3.3.

```
        si (pkt.mode = 0)                    /* pour compatibilité avec les vieilles versions */
            mode ← (voir au paragraphe 3.3) ;
        autrement
            mode ← pkt.mode ;
```

Le Tableau 5 montre pour chaque combinaison de peer.mode et *mode* les étiquettes de cas résultantes.

```
        cas (mode, peer.hostmode)            /* voir au Tableau 5 */
```

S'il y a *erreur*, le paquet est tout simplement ignoré et l'association est dissoute, si elle n'avait été configurée précédemment.

```
erreur :  si (peer.config = 0) dissoudre l'association ;        /* il n'y a pas de mal */
          coupure ;
```

peer.mode → *mode* ↓	sym act 1	sym pas 2	client 3	serveur 4	diffusion 5
sym active	recv	pkt	recv[2]	xmit[2]	xmit[1, 2]
sym passive	recv	erreur	recv[2]	erreur	erreur
client	xmit[2]	xmit[2]	erreur	Xmit	xmit[1]

serveur	recv[2]	erreur	recv	erreur	erreur
diffusion	recv[1, 2]	erreur	recv[1]	erreur	erreur

Notes :
1 Un serveur de diffusion répond directement au client avec pkt.org et pkt.rec contenant les valeurs correctes. D'autres fois, le serveur diffuse seulement l'heure locale avec pkt.org et pkt.rec mis à zéro.
2 Normalement, ces combinaisons de modes ne seraient pas utilisées, cependant, dans les limites de la spécification, elle donneraient l'heure correcte.

Tableau 5. Modes et actions

Si *recv*, le paquet est traité et l'association est marquée comme atteignable si les essais cinq à huit (en-tête valide) énumérés dans la procédure de paquet réussissent. Si, en plus, les essais un à quatre réussissent (données valides), la procédure de mise à jour d'horloge est invoquée pour mettre à jour l'horloge locale. Autrement, si l'association n'était pas configurée précédemment, elle est dissoute.

```
recv :      invoquer packet;                /* traiter le paquet */
            si (en tête valide) début       /* si l'en-tête est valide, mettre à jour l'horloge locale */
                peer.reach ← peer.reach | 1 ;
                si (données valides) invoquer mise à jour d'horloge ;
                endif
            autrement
                si (peer.config = 0) dissoudre l'association;
            rupture;
```

Si *xmit* le paquet est traité et une réponse immédiate est envoyée. L'association est alors dissoute si non configurée précédemment.

```
xmit :      invoquer packet;                /* traiter le paquet */
            peer.hostpoll ← peer.peerpoll ; /* envoi de réponse immédiate */
            invoquer mise à jour de consultation;
            invoquer émission;
            si (peer.config = 0) dissoudre l'association;
            rupture ;
```

Si *pkt* le paquet est traité et l'association marquée atteignable si les essais cinq à huit (en-tête valide) énumérés dans la procédure de paquet réussissent. Si de plus, les essais un à quatre réussissent (données valides), la procédure de mise à jour d'horloge est invoquée pour mettre à jour l'horloge locale. Autrement, si l'association n'était pas configurée précédemment, une réponse immédiate est envoyée et l'association est dissoute.

```
pkt :       invoquer packet;                /* traiter le paquet */
            si (en-tête valide) début       /* si l'en-tête est valide, mettre à jour l'horloge locale */
                peer.reach ← peer.reach | 1 ;
                si (données valides) invoquer la mise à jour d'horloge;
                endif
            autrement si (peer.config = 0) début
                peer.hostpoll ← peer.peerpoll ;          /* envoi de réponse immédiate */
                invoquer mise à jour de consultation ;
                invoquer émission ;
                dissoudre l'association;
                endif
            endcase
        fin de procédure de réception;
```

3.4.4 Procédure de paquet

La procédure de paquet vérifie la validité du message, calcule les échantillons de délai/décalage et invoque les autres procédures pour filtrer les données et choisir la source de synchronisation. L'essai n° 1 exige que l'horodatage d'émission ne corresponde pas au dernier horodatage reçu du même homologue, autrement, le message pourrait être un vieux duplicata. L'essai n° 2 exige que l'horodatage originel corresponde au dernier envoyé par le même homologue ; autrement, le message pourrait être hors d'usage, bogué ou pire. En cas de mode diffusion (5) le délai d'aller-retour apparent sera de zéro et la pleine exactitude de l'opération de transfert de l'heure peut n'être pas réalisable. Cependant,

l'exactitude réalisée peut être adéquate pour la plupart des objets. La procédure de mise à jour de consultation est invoquée avec l'argument peer.hostpoll (peer.peerpoll peut avoir changé).

```
début de procédure paquet
        peer.rec ← sys.clock;                       /* capturer l'horodatage reçu */
        si ( pkt.mode ≠ 5) début
                test1 ← (pkt.xmt ≠ peer.org}) ;     /* test 1 */
                test2 ← (pkt.org = peer.xmt) ;      /* test 2 */
                endif
        autrement début
                pkt.org ← peer.rec ;            /* fusion des horodatages manquants */
                pkt.rec ← pkt.xmt ;
                test1 ← vrai ;                  /* essais factices */
                test2 ← vrai ;
                endif
        peer.org ← pkt.xmt ;                    /* mettre à jour l'horodatage originel */
        peer.peerpoll ← pkt.poll ;              /* régler l'intervalle de consultation */
        invoquer la mise à jour de consultation (peer.hostpoll);
```

L'essai n° 3 exige que l'horodatage originel et l'horodatage reçu soient tous deux différents de zéro. Si l'un des deux horodatages est à zéro, l'association n'a pas été synchronisée ou a perdu l'atteignabilité dans une des directions ou les deux.

```
test3 ← (pkt.org ≠ 0> et pkt.rec ≠ 0) ;          /* test 3 */
```

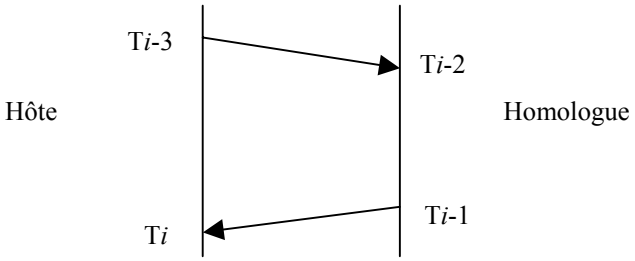

Figure 2. Calcul du délai et du décalage

Le délai d'aller-retour et le décalage d'horloge par rapport à l'homologue sont calculés comme suit. Numéroter les heures d'envoi et de réception des messages NTP comme indiqué à la Figure 2 et soit *i* u entier pair. Alors T_{i-3}, T_{i-2}, T_{i-1} et T_i sont les contenus respectifs des variables pkt.org, pkt.rec, pkt.xmt et peer.rec. Le décalage d'horloge θ, le délai d'aller retour δ et la dispersion ε de l'hôte par rapport à l'homologue sont :

$$\delta = (T_i - T_{i-3}) - (T_{i-1} - T_{i-2}) ,$$

$$\theta = \frac{(T_{i-2} - T_{i-3}) + (T_{i-1} - T_i)}{2} ,$$

$$\varepsilon = (1 \ll \text{sys.precision}) + \varphi\,(T_i - T_{i-3}) \gg ,$$

où, comme précédemment, $\varphi = \frac{NTP.MAXSKEW}{NTP.MAXAGE}$. La quantité ε représente l'erreur maximum ou la dispersion due à l'erreur de mesure à l'hôte et l'accumulation de biais d'erreur locale sur l'intervalle depuis que le dernier message a été transmis à l'homologue. Ensuite, la dispersion va être mise à jour par la procédure de filtre d'horloge.

La méthode ci-dessus revient à un système d'heure réversible à échantillonnement continu, qui est utilisé dans certains réseaux téléphoniques numériques [BEL86]. Parmi les avantages figure le fait que l'ordre et l'heure des messages sont sans importance et qu'une livraison fiable n'est pas nécessaire. Visiblement, l'exactitude réalisable dépend des propriétés statistiques des chemins d'entrée et de sortie des données. Une analyse plus poussée et des résultats expérimentaux portant sur cette question figurent dans [MIL90] et à l'Appendice H.

L'essai n° 4 exige que le délai calculé soit dans des limites "raisonnables" :

```
test4 ← (|δ| < NTP.MAXDISPERSE et ε < NTP.MAXDISPERSE) ;          /* test 4 */
```

L'essai n° 5 n'est mis en œuvre que si le mécanisme d'authentification décrit à l'Appendice C est mis en œuvre. Il exige que l'authentification soit explicitement désactivée ou que l'authentifiant soit présent et correct comme déterminé par la procédure de déchiffrement.

```
#ifdef (authentification mise en œuvre)                    /* test 5 */
        test5 ← ((peer.config = 1 et peer.authenable = 0) ou peer.authentic = 1} ;
        #endef
```

L'essai n° 6 exige que l'horloge homologue soit synchronisée et que l'intervalle depuis que l'horloge homologue a été mise à jour pour la dernière fois soit positif et inférieur à NTP.MAXAGE. L'essai n° 7 garantit que l'hôte ne se synchronisera pas sur un homologue ayant une strate supérieure. L'essai n° 8 exige que l'en-tête contienne des valeurs "raisonnables" pour les champs pkt.rootdelay et pkt.rootdispersion.

```
test6 ← (pkt.leap ≠ 11₂ et                                 /* test 6 */
        pkt.reftime ≤ pkt.xmt < pkt.reftime + NTP.MAXAGE}>)
test7 ← roman {pkt.stratum ≤ sys.stratum} et               /* test 7 */
        pkt.stratum < NTP.MAXSTRATUM} ;
test8 ← (|pkt.rootdelay| < NTP.MAXDISPERSE} et             /* test 8 */
        pkt.rootdispersion < NTP.MAXDISPERSE) ;
```

Par rapport au traitement ultérieur, le paquet inclut des données valides (synchronisées) si les essais un à quatre réussissent *(test1 & test2 & test3 & test4 = 1)*, sans considération des essais restants. Seuls les paquets avec des données valides peuvent être utilisés pour calculer les valeurs de décalage, de délai et de dispersion. Le paquet inclut un en-tête valide si les essais n° 5 à 8 réussissent *(test5 & test6 & test7 & test8 = 1)*, sans considération des essais restants. Seuls les paquets avec des en-tête valides peuvent être utilisés pour déterminer si un homologue peut être choisi pour la synchronisation. Noter que *test1* et *test2* ne sont pas utilisés en mode diffusion (forcé à vrai), car les horodatages d'origine et de réception sont indéfinis.

La procédure de filtre d'horloge est invoquée pour produire le délai (peer.delay), le décalage (peer.offset) et la dispersion (peer.dispersion) pour l'homologue. La spécification de l'algorithme de filtre d'horloge ne fait pas partie intégrante de la spécification NTP, car il peut y avoir d'autres algorithmes qui fonctionnent bien, en pratique. Cependant, un algorithme qui fonctionne bien dans l'environnement de l'Internet est décrit à la Section 4 et son utilisation est recommandée.

```
si (en-tête non valide) exit ;
peer.leap           ← pkt.leap>;                    /* copier les variables de paquet */
peer.stratum        ← pkt.stratum>;
peer.precision      ← pkt.precision>;
peer.rootdelay      ← pkt.rootdelay>;
peer.rootdispersion ← pkt.rootdispersion>;
peer.refid          ← pkt.refid>;
peer.reftime        ← pkt.reftime>;
si (données valides) invoquer filtre d'horloge (θ, δ ε);    /* traiter les échantillons */
fin de procédure de paquet;
```

3.4.5 Procédure de mise à jour d'horloge

La procédure de mise à jour d'horloge est invoquée à partir de la procédure de réception lorsque des données valides de décalage d'horloge, de délai et de dispersion ont été déterminées par la procédure de filtre d'horloge pour l'homologue en cours. Le résultat des procédures de choix d'horloge et de combinaison d'horloge est la correction d'horloge finale Θ, qui est utilisée par la procédure d'horloge locale pour mettre à jour l'horloge locale. Si aucun candidat ne survit à ces procédures, la procédure de mise à jour d'horloge se termine sans rien faire d'autre.

```
début de procédure de mise à jour d'horloge
        invoquer choix d'horloge            /* choisir la source d'horloge */
        si (sys.peer ≠ peer) exit ;
```

Il peut arriver que l'horloge locale soit remise à zéro, plutôt que réorientée vers sa valeur finale. Dans ce cas, la procédure de libération est invoquée pour que chaque homologue purge le filtre d'horloge, rétablisse l'intervalle de consultation et recommence le choix de la source de synchronisation, si nécessaire. Noter que la procédure d'horloge locale établit les sauts de bits sys.leap à "non synchronisé" 11_2 dans ce cas, de sorte qu'aucun autre homologue ne tentera de se synchroniser à l'hôte jusqu'à ce que l'hôte choisisse à nouveau un homologue pour la synchronisation.

La procédure de distance calcule le délai de racine Δ, la dispersion de racine E et la distance de synchronisation de racine Λ via l'homologue jusqu'à la racine du sous réseau de synchronisation. L'hôte ne va pas se synchroniser avec l'homologue choisi si la distance est supérieure à NTP.MAXDISTANCE. La raison pour le blocage minimum à NTP.MINDISPERSE est de décourager les va et vient d'acheminement de sous réseau qui peuvent arriver avec les algorithmes Bellman-Ford et de petits délais d'aller-retour.

```
Λ andistance(peer) ;                          /* mettre à jour les variables de système */
si (Λ ≥ NTP.MAXDISTANCE>) exit ;
sys.leap ← peer.leap ;
sys.stratum ← peer.stratum + 1 ;
sys.refid ← peer.peeraddr ;
invoquer l'horloge locale;
si (réinitialisation d'horloge locale) début      /* si réinitialisation, éliminer les variables d'état */
        sys.leap ← 11₂ ;
        pour (tous homologues) invoquer libérer ;
        endif
autrement début
        sys.peer ← peer ;                         /* si non, régler l'horloge locale */
        sys.rootdelay ← Δ ;
        sys.rootdispersion ← E + max (εξ + |Θ|, NTP.MINDISPERSE) ;
        endif
sys.reftime ← sys.clock ;
fin de procédure de mise à jour d'horloge;
```

Dans certaines configurations de système, une source d'informations horaires précise est disponible sous la forme d'un train d'impulsions horaires espacées à une seconde d'intervalle. Habituellement, c'est en plus une source d'informations de code horaire, telle que horloge radio ou même NTP lui-même, pour dénombrer les secondes, les minutes, les heures et les jours. Dans ces configurations, les variables de système sont établies pour se référer à la source d'où sont déduites les impulsions. Pour ces configurations qui prennent en charge une source de référence primaire, telles qu'une horloge radio ou une horloge atomique calibrée, la strate est réglée à un pour autant que ce soit la source de synchronisation réelle et que la procédure d'horloge primaire soit utilisée ou non.

La spécification des algorithmes de choix d'horloge et d'horloge locale ne fait pas partie intégrante de la spécification NTP, car il peut y avoir d'autres algorithmes qui procurent des performances équivalentes. Cependant, un algorithme de choix d'horloge qui se trouve bien fonctionner dans l'environnement de l'Internet est décrit à la Section 4, et un algorithme d'horloge locale est décrit à la Section 5 et leur utilisation est recommandée. L'algorithme de choix d'horloge décrit à la Section 4 prend habituellement l'homologue à la strate la plus basse et la distance de synchronisation minimum parmi toutes celles disponibles, à moins que cet homologue apparaisse être un faux tic-tac. Le résultat est que les algorithmes travaillent tous à construire un arbre d'expansion à pondération minimale par rapport aux serveurs horaires de référence primaire et donc un sous réseau de synchronisation hiérarchisé maître-esclave.

3.4.6 Procédure d'horloge primaire

Lorsqu'une source de référence primaire telle qu'une horloge radio est connectée à l'hôte, il est pratique d'incorporer ses informations dans la base de données comme si l'horloge était représentée comme un homologue ordinaire. Dans la procédure d'horloge primaire, l'horloge est consultée une fois par minute environ et le code horaire retourné est utilisé pour produire une nouvelle mise à jour pour l'horloge locale. Lorsque peer.timer descend jusqu'à zéro pour un homologue d'horloge primaire, la procédure d'émission n'est pas invoquée ; l'horloge radio est plutôt consultée, habituellement en utilisant une chaîne ASCII spécifiée à cet effet. Lorsqu'un code horaire valide est reçu de l'horloge radio, il est converti en format d'horodatage NTP et les variables d'homologue sont mises à jour. La valeur de peer.leap est réglée en fonction de l'état du bit d'avertissement de saut dans le code horaire, s'il est disponible, ou manuellement par l'opérateur. La valeur pour peer.peeraddr, qui va devenir la valeur de sys.refid lorsque la procédure de mise à jour d'horloge est invoquée, est réglée à une chaîne ASCII qui décrit le type d'horloge (voir l'Appendice A).

```
début de procédure de mise à jour d'horloge primaire
        peer.leap ← de radio ou d'opérateur ;                  /* copier les variables */
        peer.peeraddr ← identifiant ASCII ;
        peer.rec ← horodatage radio ;
        peer.reach ← 1 ;
        invoquer le filtre d'horloge(sys.clock - peer.rec, 0, 1 << peer.precision) ;   /* traiter les échantillons */
        invoquer la mise à jour d'horloge ;                    /* mettre à jour l'horloge locale */
        fin de procédure de mise à jour d'horloge primaire;
```

3.4.7 Procédures d'initialisation

Les procédures d'initialisation servent à établir et initialiser le système, ses homologues et associations.

3.4.7.1 Procédure d'initialisation

La procédure d'initialisation est invoquée au réamorçage ou au redémarrage du démon NTP. L'horloge locale est vraisemblablement indéfinie au réamorçage ; cependant, dans certains équipements, une estimation est disponible à partir de l'environnement de réamorçage, comme une horloge/calendrier sur batterie de secours. La variable de précision est déterminée par l'architecture intrinsèque de l'horloge matérielle locale. Les variables d'authentification ne sont utilisées que si le mécanisme d'authentification décrit à l'Appendice C est mis en œuvre. Les valeurs de ces variables sont déterminées en utilisant des procédures qui sortent du domaine d'application de NTP.

```
début de procédure d'initialisation
        #ifdef (authentification mise en œuvre)          / * voir l'Appendice C */
                sys.keys ← as required ;
                #endef ;
        sys.leap ← 11₂ ;                                 /* copier les variables */
        sys.stratum ← 0 (indéfini) ;
        sys.precision ← précision d'hôte ;
        sys.rootdelay ← 0 (indéfini) ;
        sys.rootdispersion ← 0 (indéfini) ;
        sys.refid ← 0 (indéfini) ;
        sys.reftime ← 0 (indéfini) ;
        sys.clock ← référence externe ;
        sys.peer ← NULL ;
        sys.poll ← NTP.MINPOLL ;
        pour (tout homologue configuré)                  /* créer les associations configurées */
                invoquer procédure d'initialisation-instanciation ;
        fin de la procédure d'initialisation ;
```

3.4.7.2 Procédure d'initialisation-instanciation

Cette procédure spécifique de mise en œuvre est invoquée à partir de la procédure d'initialisation pour définir une association. Les adresses et modes des homologues sont déterminés en utilisant les informations lues durant la procédure de réamorçage ou par suite de commandes de l'opérateur. Les variables d'authentification ne sont utilisées que si le mécanisme d'authentification décrit à l'Appendice C est mis en œuvre. Les valeurs de ces variables sont déterminées en utilisant des procédures qui sortent du domaine d'application de NTP. Avec les bits d'authentification mis comme suggéré, seuls les homologues correctement authentifiés peuvent devenir des sources de synchronisation.

```
début de procédure d'initialisation-instanciation
        peer.config ← 1 ;
        #ifdef (authentification mise en œuvre)          /* voir l'Appendice C */
                peer.authenable ← 1 (suggéré) ;
                peer.authentic ← 0 ;
                peer.hostkeyid ← comme requis ;
                peer.peerkeyid ← 0 ;
                #endef ;
        peer.peeraddr ← adresse IP d'homologue ;         /* copier les variables */
        peer.peerport ← NTP.PORT ;
        peer.hostaddr ← adresse IP d'hôte ;
        peer.hostport ←  NTP.PORT ;
        peer.mode ← mode hôte ;
        peer.peerpoll ← 0 (indéfini) ;
        peer.timer ← 0 ;
        peer.delay ← 0 (indéfini) ;
        peer.offset ← 0 (indéfini) ;
        invoquer libération                              /* initialiser l'association */
        fin de procédure d'initialisation-instanciation;
```

3.4.7.3 Procédure de réception-instanciation

La procédure de réception–instanciation est invoquée à partir de la procédure de réception lorsqu'un nouvel homologue est découvert. Elle initialise les variables d'homologue et mobilise l'association. Si le message provient d'un homologue qui fonctionne en mode client (3), le mode hôte est réglé à mode serveur (4) ; autrement, il est réglé à mode passif symétrique (2). Les variables d'authentification ne sont utilisées que si le mécanisme d'authentification décrit dans l'Appendice C est mis en œuvre. S'il est mis en œuvre, seuls les homologues non configurés correctement authentifiés peuvent devenir la source de synchronisation.

```
début de procédure de réception-instanciation
    #ifdef (authentification mise en œuvre)          /* voir l'Appendice C */
        peer.authenable ← 0 ;
        peer.authentic ← 0 ;
        peer.hostkeyid ← comme requis ;
        peer.peerkeyid ← 0 ;
        #endef
    peer.config ← 0 ;                                /* copier les variables */
    peer.peeraddr ← pkt.peeraddr>;
    peer.peerport ← pkt.peerport ;
    peer.hostaddr ← pkt.hostaddr ;
    peer.hostport ← pkt.hostport ;
    si (pkt.mode = 3)                                /* déterminer le mode */
        peer.mode ← 4 ;
        autrement
        peer.mode ← 2 ;
    peer.peerpoll ← 0 (indéfini) ;
    peer.timer ← 0 ;
    peer.delay ← 0 (indéfini) ;
    peer.offset ← 0 (indéfini) ;
    invoquer libération;                             /* initialiser l'association */
    fin de procédure de réception-instanciation;
```

3.4.7.4 Procédure d'instanciation d'horloge primaire

Cette procédure est invoquée à partir de la procédure d'initialisation dans le but d'établir les variables d'état pour l'horloge primaire. Les valeurs pour peer.precision sont déterminées à partir de la spécification d'horloge radio et d'interface matérielle. La valeur pour peer.rootdispersion est nominalement dix fois l'erreur inhérente maximum de l'horloge radio ; par exemple, 10 µs pour une horloge atomique calibrée, 10 ms pour une horloge radio WWVB ou GOES et 100 ms pour une horloge radio WWV moins exacte.

```
début de procédure d'instanciation d'horloge
    peer.config ← 1 ;                                /* copier les variables */
    peer.peeraddr ← 0 indéfini ;
    peer.peerport ← 0 (non utilisé) ;
    peer.hostaddr ← 0 (non utilisé) ;
    peer.hostport ← 0 (non utilisé) ;
    peer.leap ← 11₂ ;
    peer.mode ← 0 (non utilisé) ;
    peer.stratum ← 0 ;
    peer.peerpoll ← 0 (indéfini) ;
    peer.precision ← précision d'horloge ;
    peer.rootdelay ← 0 ;
    peer.rootdispersion ← dispersion d'horloge ;
    peer.refid ← 0 (non utilisé) ;
    peer.reftime ← 0 (indéfini) ;
    peer.timer ← 0 ;
    peer.delay ← 0 (indéfini) ;
    peer.offset ← 0 (indéfini) ;
    invoquer libération;                             /* initialiser l'association */
    fin de procédure d'instanciation d'horloge;
```

Dans certaines configurations qui impliquent une horloge atomique calibrée ou un récepteur LORAN-C, la source de référence primaire peut ne fournir qu'une impulsion de secondes, mais n'avoir pas de code de temps complet à partir duquel puisse être déduit le dénombrement des secondes, etc. Dans ces configurations, le dénombrement des secondes

peut être déduit d'autres sources, telles que des horloges radio ou même d'autres homologues NTP. Dans ces configurations, les variables d'horloge primaire devraient refléter la source de référence primaire, et non la source de dénombrement des secondes ; cependant, si la source de dénombrement des secondes est défaillante ou si son mauvais fonctionnement est connu, les mises à jour provenant de la source de référence primaire devraient être supprimées comme si elles avaient échoué.

3.4.8 Procédure de libération

La procédure de libération est invoquée lorsque survient quelque événement qui a pour résultat un changement significatif dans l'état d'atteignabilité ou une interruption potentielle de l'horloge locale.

```
début de procédure de libération
        peer.org ← 0 (indéfini) ;                   /* marquer les horodatages comme indéfinis */
        peer.rec ← 0 (indéfini) ;
        peer.xmt ← 0 (indéfini) ;
        peer.reach ← 0 ;                            /* remettre à zéro les variables d'état */
        peer.filter ← [0, ,0, NTP.MAXDISPERSE] ;    /* toutes étapes */
        peer.valid ← 0 ;
        peer.dispersion ← NTP.MAXDISPERSE ;
        peer.hostpoll ← NTP.MINPOLL ;               /* remettre à zéro l'intervalle de consultation */
        invoquer la mise à jour de consultation ;
        invoquer le choix d'horloge ;               /* choix de la source d'horloge */
        fin de la procédure de libération ;
```

3.4.9 Procédure de mise à jour de consultation

La procédure de mise à jour de consultation est invoquée lorsque survient un événement significatif qui peut avoir pour résultat un changement de l'intervalle de consultation ou de temporisateur d'homologue. Elle vérifie les valeurs de l'intervalle de consultation de l'hôte (peer.hostpoll) et l'intervalle de consultation de l'homologue (peer.peerpoll) et fait rentrer chacun dans la gamme valide. Si l'homologue est choisi pour la synchronisation, la valeur est saisie comme fonction de la conformité calculée (voir à la Section 5).

```
début de la procédure de mise à jour de consultation
        temp ← peer.hostpoll ;              /* déterminer l'intervalle de mise à jour de consultation */
        si (homologue = sys.peer )
                temp ← min (temp, sys.poll, NTP.MAXPOLL)} ;
        autrement
                temp ← min (temp, NTP.MAXPOLL) ;
        peer.hostpoll ← max (temp, NTP.MINPOLL) ;
        temp ← 1 << min (peer.hostpoll, max (peer.peerpoll, NTP.MINPOLL)) ;
```

Si l'intervalle de consultation reste inchangé et si le temporisateur d'homologue est zéro, le temporisateur est simplement remis à zéro. Si l'intervalle de consultation est changé et si la nouvelle valeur du temporisateur est supérieure à la valeur présente, aucune action supplémentaire n'est nécessaire, le temporisateur homologue doit être réduit. Lorsque le temporisateur homologue doit être réduit, il est important de décourager les tendances à synchroniser les transmissions entre les homologues. Une précaution de prudence est de rendre aléatoire la première transmission après la réduction du temporisateur, par exemple, à l'aide de la technique furtive illustrée.

```
        si (peer.timer = 0)                 /* remettre le temporisateur homologue à zéro */
                peer.timer ← temp ;
        autrement si (peer.timer > temp)
                peer.timer ← (sys.clock & (temp -  1)) + 1 ;
        fin de la procédure de mise à jour de consultation;
```

3.5 Procédure de distance de synchronisation

La procédure de distance calcule la distance de synchronisation à partir des variables d'homologue pour l'homologue *peer*.

```
début de la procédure de distance (homologue) ;
        Δ ← peer.rootdelay + |peer.delay| ;
        E ← peer.rootdispersion + peer.dispersion + φ (sys.clock - peer.update) ;
```

$$\Lambda \leftarrow E + \frac{|\Delta|}{2} ;$$

fin de la procédure de distance;

Noter que, alors que Δ peut être négatif dans certains cas, E et Λ sont tous deux toujours positifs.

3.6 Question de contrôle d'accès

La conception de NTP est telle que la modification accidentelle ou malveillante (altération) ou la destruction (mélange) de données à un serveur horaire ne devrait en général pas avoir pour résultat d'erreurs de conservation de l'heure ailleurs que dans le sous réseau de synchronisation. Cependant, le succès de cette approche dépend de la redondance des serveurs horaires et de la diversité des chemins de réseau, conjointement avec l'hypothèse que l'altération ou le mélange ne vont pas survenir sur de nombreux serveurs horaires partout sur le sous réseau de synchronisation au même moment. En principe, la vulnérabilité du sous réseau peut être encadrée par le choix de serveurs horaires connus pour être de confiance et en permettant seulement à ces serveurs horaires de devenir source de synchronisation. Les procédures d'authentification décrites dans l'Appendice C représentent un des mécanismes de sa mise en application ; cependant, les algorithmes de chiffrement peuvent être intenses en CPU et peuvent sérieusement dégrader l'exactitude, à moins que des précautions telles que celles mentionnées dans la description de la procédure d'émission ne soient prises.

Lorsqu'elles ne sont pas une caractéristique exigée de NTP lui-même, certaines mises en œuvre peuvent inclure un dispositif de contrôle d'accès qui empêche l'accès non autorisé et contrôle quels homologues sont admis à mettre à jour l'horloge locale. Dans ce but, il est utile de distinguer trois catégories d'accès : ceux qui sont pré autorisés comme de confiance, ceux pré autorisés comme amicaux et tous les autres accès (non pré autorisés). Vraisemblablement, la pré autorisation est accomplie par des entrées dans le fichier de configuration ou une sorte de système de ticket de gestion comme avec Kerberos [STE88]. Dans ce modèle, seuls les accès de confiance peuvent avoir pour résultat que l'homologue devienne la source de synchronisation. Alors que les accès amicaux ne peuvent pas avoir pour résultat que l'homologue devienne la source de synchronisation, les messages NTP et les horodatages sont retournés comme spécifié.

Il ne semble pas utile de maintenir une horloge secrète, comme il résulterait d'une interdiction des accès qui ne seraient pas pré autorisés, sauf si l'intention est de cacher l'existence du serveur horaire lui-même. Les hôtes Internet de bon comportement sont supposés retourner un message d'erreur de service ICMP indisponible si un service n'est pas mis en œuvre ou si les ressources ne sont pas disponibles ; cependant, dans le cas de NTP, les ressources requises sont minimales, et il y a donc peu d'intérêt à interdire les demandes destinées seulement à lire l'horloge. Un mécanisme simple mais efficace de contrôle d'accès est de considérer toutes les associations préconfigurées dans un mode symétrique ou un mode client (modes 1, 2 et 3) comme de confiance et toutes les autres associations, préconfigurées ou non, comme amicales.

Si un modèle de confiance plus complet est nécessaire, sa conception peut se fonder sur une liste de contrôle d'accès dont chaque entrée consiste en une adresse Internet de 32 bits, un gabarit de 32 bits et trois bits de mode. Si le ET logique de l'adresse de source (pkt.peeraddr) et le gabarit d'une entrée sont en accord avec l'adresse correspondante dans l'entrée et que le mode (pkt.mode) satisfait au mode dans l'entrée, l'accès est permis ; autrement, un message d'erreur ICMP est retourné au demandeur. Par le choix approprié d'un gabarit, il est possible d'interdire les requêtes par mode d'adresse individuelle, de sous réseau particulier ou des adresses de réseau, ou de n'avoir pas de restrictions du tout. La liste de contrôle d'accès servirait alors de filtre pour contrôler quels homologues peuvent créer des associations.

4. Algorithmes de filtrage et de choix

Un facteur des plus importants qui affecte l'exactitude et la fiabilité de la distribution de l'heure est le complexe d'algorithmes utilisés pour réduire l'effet des erreurs statistiques et des faux tic-tac dû à la défaillance de divers composants de sous réseau, de sources de référence ou de supports de propagation. Les algorithmes suggérés dans la présente section ont été développés et affinés sur plusieurs années de fonctionnement dans l'Internet sous des topologies des vitesses et des régimes de trafic très variés. Alors qu'on estime que ces algorithmes sont les meilleurs parmi ceux disponibles à l'heure actuelle, ils ne font pas partie intégrante de la spécification NTP, car d'autres algorithmes avec des performances similaires ou supérieures peuvent être développés à l'avenir.

Cependant, il est important d'observer que ce ne sont pas les serveurs ou les clients d'un sous réseau de synchronisation NTP qui doivent en tout temps mettre en œuvre ces algorithmes. Par exemple, de simples stations de travail peuvent se passer de l'un d'eux ou des deux au nom de la simplicité si les exigences d'exactitude et de fiabilité le justifient. A tout

le moins, on s'attend à ce qu'un serveur NTP qui fournit la synchronisation à une communauté d'une certaine taille, comme un campus d'université ou un laboratoire de recherche, mette en œuvre ces algorithmes ou d'autres ayant des fonctionnalités équivalentes. Une discussion complète des principes de conception et des performances est donnée dans [MIL91a].

Pour que les algorithmes NTP de filtre et de sélection fonctionnent de façon effective, il est utile d'avoir une mesure de la variance des échantillons récents enregistrés pour chaque homologue. La mesure adoptée se fonde sur des différences de premier ordre, qu'il est facile de calculer et qui sont pertinentes pour l'objet recherché. Il y a deux mesures, une appelée la *dispersion de filtre* ε_σ et l'autre la *dispersion choisie* ε_ξ. Toutes deux sont calculées comme la somme pondérée des décalages d'horloge dans une liste temporaire triée par distance de synchronisation. Si θ_i ($0 \le i < n$) est le décalage de la i^{ème} entrée, la différence d'échantillon ε_{ij} de la i^{ème} entrée par rapport à la j^{ème} entrée est alors définie $\varepsilon_{ij} = |\theta_i - \theta_j|$. La dispersion par rapport à la j^{ème} entrée est définie ε_j et calculée comme la somme pondérée de

$$\varepsilon_j = \sum_{i=0}^{n-1} \varepsilon_{ij} w^{i+j} ,$$

où w est un facteur de pondération choisi pour contrôler l'influence de la distance de synchronisation dans le budget de dispersion. Dans les algorithmes NTP, w est choisi inférieur à ½ : w = NTP.FILTER pour la dispersion de filtre et à w = NTP.SELECT pour la dispersion choisie. La dispersion (absolue) ε_σ et ε_ξ telle qu'utilisée dans les algorithmes NTP est définie par rapport à la 0ème entrée ε_0.

Deux procédures sont décrites ci-après, la procédure de filtre d'horloge, qui sert à choisir les meilleurs échantillons de décalage d'une horloge donnée, et la procédure de choix d'horloge, qui sert à choisir la meilleure horloge parmi un ensemble hiérarchisé d'horloges.

4.1 Procédure de filtre d'horloge

La procédure de filtre d'horloge est exécutée dès l'arrivée d'un message NTP ou autre événement qui a pour résultat de nouveaux échantillons de données. Elle s'articule sur la forme (θ, δ, ε), où θ est un échantillon de mesure de décalage d'horloge et δ et ε sont le délai d'aller retour et la dispersion qui y sont associés. Elle détermine le décalage de filtre d'horloge (peer.offset), le délai d'aller-retour (peer.delay) et la dispersion (peer.dispersion). Elle met aussi à jour la dispersion des échantillons déjà enregistrés et sauvegarde l'heure en cours (peer.update).

La base de la procédure de filtre d'horloge est le registre à décalage de filtre (peer.filter), qui consiste en étapes NTP.SHIFT, chaque étape contenant un triplet [θ_i, δ_i, ε_i], avec des indices numérotés à partir de zéro sur la gauche. Le filtre est initialisé avec la valeur [0, 0, NTP.MAXDISPERSE] dans toutes les étapes par la procédure de libération. De nouveaux échantillons de données sont glissés par la gauche dans le filtre, ce qui cause les premiers NULL, puis les vieux échantillons sont évacués par la droite. La procédure de paquet fournit des échantillons de la forme (θ, δ, ε) lorsqu'arrivent les nouvelles mises à jour, alors que la procédure de transmission fournit des échantillons de la forme [0, 0, NTP.MAXDISPERSE] lorsque deux intervalles de consultation s'écoulent sans nouvelle mise à jour. Alors que les mêmes symboles (θ, δ, ε) sont utilisés ici pour les arguments, le contenu de filtre d'horloge et les variables d'homologue, leur signification se précisera selon le contexte. Le pseudo code suivant décrit cette procédure.

début de procédure de filtre d'horloge (θ, δ, ε)

La dispersion ε_i pour tous les échantillons valides dans le registre de filtre doit être mise à jour pour tenir compte de l'accumulation de l'erreur de biais depuis la dernière mise à jour. Ces échantillons sont aussi insérés dans une liste temporaire avec le format d'entrée [distance,index]. Les échantillons dans le registre glissent à droite à chaque étape, avec élimination de l'échantillon qui déborde et le nouvel échantillon inséré à l'étape la plus à gauche. La liste temporaire est alors triée par distance croissante. Si aucun échantillon ne reste dans la liste, la procédure se termine sans mise à jour des variables d'homologue.

pour (i de NTP.SIZE - 1 à 1) **début** /* mettre à jour la dispersion */
[θ_i , δ_i , ε_i] ← [θ_{i-1}, δ_{i-1}, ε_{i-1}] ; /* glissement d'étape à droite*/
$\varepsilon_i = \varepsilon_i + \varphi\tau$;
ajouter [$\lambda_i \equiv \varepsilon_i + \frac{|\delta i|}{2}, i$] à la liste temporaire;
endfor;
[θ_0, δ_0, ε_0] ← [θ, δ, ε] ; /* insérer le nouvel échantillon */

ajouter [$\lambda \equiv \varepsilon + \frac{|\delta|}{2}$, 0] à la liste temporaire;
peer.update ← sys.clock ; /* remettre la base de temps à zéro */
trier la liste temporaire par [*distance* ||*index*] croissante;

où [*distance* ||*index*] représente l'enchaînement des champs *distance* et *indice* et *distance* est le champ de plus fort poids. La dispersion de filtre ε_σ est calculée et incluse dans la dispersion d'homologue. Noter que la liste temporaire est déjà triée dans ce but.

$\varepsilon_\sigma \leftarrow 0$;
pour (*i* de NTP.SHIFT - 1 à 0) /* calculer la dispersion de filtre */
si (peer.dispersion$_{index[i]} \geq$ NTP.MAXDISPERSE **ou**
$|\theta_i - \theta_0| >$ NTP.MAXDISPERSE)
$\varepsilon_\sigma \leftarrow (\varepsilon_\sigma +$ NTP.MAXDISPERSE) × NTP.FILTER ;
autrement
$\varepsilon\sigma \leftarrow (\varepsilon_\sigma + \theta_i - \theta_0 |) \times$ NTP.FILTER ;

Le décalage d'homologue θ_0, le délai δ_0 et la dispersion ε_0 sont choisis comme les valeurs correspondant à l'échantillon de distance minimum ; en d'autres termes, l'échantillon correspondant à la première entrée sur la liste temporaire, ici représenté comme le $0^{ème}$ indice.

peer.offset ← θ_0 ; /* mettre les variables d'homologue à jour */
peer.delay ← δ_0 ;
peer.dispersion ← min ($\varepsilon_0 + \varepsilon_\sigma$, NTP.MAXDISPERSE) ;
fin de procédure de filtre d'horloge

Les variables peer.offset et peer.delay représentent le décalage d'horloge et le délai d'aller-retour de l'horloge locale par rapport à l'horloge homologue. Toutes deux sont des quantités de précision qui peuvent habituellement être moyennées sur de longs intervalles afin d'améliorer l'exactitude et la stabilité sans accumulation de biais (voir l'Appendice H). La variable peer.dispersion représente l'erreur maximum due à l'erreur de mesure, à l'accumulation d'erreur de biais et à la variance d'échantillon. Ces trois variables sont toutes utilisées dans les procédures de choix d'horloge et de combinaison d'horloge pour choisir la ou les horloges homologues utilisées pour la synchronisation et pour maximiser l'exactitude et la stabilité des indications.

4.2 Procédure de choix d'horloge

La procédure de choix d'horloge utilise les variables d'homologue Θ, Δ, E et τ et elle est invoquée lorsque ces variables changent ou lorsque l'état d'atteignabilité change. Elle consiste en deux algorithmes, l'algorithme d'intersection et l'algorithme de mise en grappe. L'algorithme d'intersection construit une liste de candidats homologues éligibles pour devenir source de synchronisation, calcule un intervalle de confiance pour chacun est élimine les faux tic-tac en utilisant une technique adaptée de Marzullo et Owicki [MAR85]. L'algorithme de mise en grappe trie la liste des candidats survivants dans l'ordre des strates et des distances de synchronisation et renouvelle l'élimination des divergents sur la base de la dispersion choisie jusqu'à ce que ne soient plus laissés que les survivants les plus exacts, précis et stables. Un bit est mis pour chaque survivant pour indiquer le résultat du processus de sélection. La variable système sys.peer est mise comme pointeur sur le survivant le plus vraisemblable, s'il en est un, ou sur la valeur NULL s'il n'y en a pas.

4.2.1 Algorithme d'intersection

début de la procédure de choix d'horloge

Chaque homologue est examiné à son tour et ajouté à une liste de points d'extrémité s'il réussit à passer plusieurs contrôles de santé destinés à éviter les circuits en boucle et l'utilisation de données exceptionnellement bruyantes. Si aucun homologue ne survit aux contrôles de santé, la procédure se termine sans trouver de source de synchronisation. Pour chaque *m* survivant sont ajoutées trois entrées de la forme [*endpoint*, *type*] à la liste de points d'extrémité : [Θ – Λ, – 1], [Θ, 0] et [Θ + Λ, 1]. Il y aura 3 *m* entrées sur la liste, qui est alors triée par points d'extrémité croissants.

m ← 0 ;
pour (chaque *homologue*) /* invoquant tous les homologues */
si (peer.reach != 0 **et** peer.dispersion < NTP.MAXDISPERSE **et**
pas (peer.stratum >> 1 et peer.refid = peer.hostaddr)) **début**

```
                Λ an distance(homologue) ;                    /* faire une entrée de liste */
                ajouter [Θ – Λ, – 1] à la liste de point d'extrémité;
                ajouter [Θ, 0] à la liste de point d'extrémité;
                ajouter [Θ + Λ, 1] à la liste de point d'extrémité;
                m ← m + 1 ;
                endif
        endfor
    si (m = 0) début                                          /* échappement si pas de candidat */
        sys.peer ← NULL ;
        sys.stratum ← 0 (indéfini) ;
        exit ;
        endif
    trier la liste des points d'extrémité par point d'extrémité||type croissant ;
```

L'algorithme suivant est adapté de DTS [DEC89] et est conçu pour produire la plus grande intersection unique contenant seulement des vraies chimères. L'algorithme commence par initialiser une valeur *f* et des compteurs *i* et *c* à zéro. Puis, en commençant pas le point d'extrémité le plus bas de la liste des points d'extrémité triés, pour chaque entrée [*endpoint*, *type*] la valeur du *type* est soustraite du compteur *i*, qui est le nombre des intersections. Si le *type* est zéro, incrémenter la value de *c*, qui est le nombre de faux tic-tac (voir l'Appendice H). Si $i \geq m - f$ pour une entrée, le point d'extrémité de cette entrée devient le point d'extrémité le plus bas de l'intersection ; autrement, *f* est augmenté de un et la procédure ci-dessus se répète. Sans remettre *f* ou *c* à zéro, une procédure similaire est utilisée pour trouver le point d'extrémité le plus haut, sauf que la valeur du type est ajoutée au compteur. Si après que les deux points d'extrémité ont été déterminés $c \leq f$, la procédure continue en ayant trouvé $m - f$ vraies chimères ; autrement, *f* est augmenté de un et toute la procédure est répétée.

```
    pour (f de 0 à f ≥ m/2 ) début                            /* invoquant toutes les vraies chimères */
        c ← 0 ;
        i ← 0 ;
        pour (chaque [endpoint, type] depuis le plus bas) début    /* trouver le point le plus bas */
            i ← i - type ;
            bas ← point d'extrémité ;
            si (i ≥ m - f) interruption;
            si (type = 0) c ← c + 1 ;
            endfor;
        i ← 0 ;
        pour (chaque [point d'extrémité, type] depuis le plus haut) début    /* trouver le point le plus haut */
            i ← i + type ;
            haut ← point d'extrémité ;
            si (i ≥ m - f) interruption;
            si (type = 0) c ← c + 1 ;
            endfor ;
        si (c ≤ f) interruption;                              /* continuer jusqu'à trouver les faux tic-tac */
        endfor ;
    si (bas > haut) début                                     /* quitter si aucune intersection n'est trouvée */
        sys.peer ← NULL ;
        exit;
        endif;
```

Noter que le traitement ne continue au-delà de ce point que s'il y a plus de $\frac{m}{2}$ intersections. Cependant, il est possible, mais pas très vraisemblable, qu'il y ait moins de $\frac{m}{2}$ vraies chimères restantes dans l'intersection.

4.2.2 Algorithme de mise en grappe

Dans l'algorithme DTS originel, la procédure de choix d'horloge se termine à ce moment avec l'heure correcte supposée établie au milieu de l'intersection calculée [*bas*, *haut*]. Cependant, cela peut conduire à de considérables pertes d'exactitude et de stabilité, dans la mesure où les statistiques d'homologue individuel sont perdues. Donc, dans NTP, les candidats qui ont survécu aux étapes précédentes sont retravaillés. La liste des candidats est reconstruite avec des entrées de la forme [*distance*, *index*], où la distance est calculée (échelonnée) à partir de la strate d'homologue et de

la distance de synchronisation Λ. Le facteur d'échelonnement procure un mécanisme pour pondérer la combinaison de la strate et de la distance. Ordinairement, la strate va dominer, à moins qu'un ou plusieurs des survivants aient une distance exceptionnellement longue. La liste est alors triée par distance croissante.

```
m ← 0 ;
pour (chaque homologue) début                          /* invocation de tous les homologues */
        si (bas ≤ θ ≤ haut) début
                Λ ← distance (homologue) ;                      /* faire une entrée de liste */
                dist ← peer.stratum × NTP.MAXDISPERSE + Λ
                ajouter [dist , homologue] à la liste de candidats;
                m ← m + 1 ;
                endif ;
        endfor ;
trier la liste de candidat par distance croissante;
```

Les étapes suivantes sont conçues pour éliminer les divergents qui affichent des dispersions significatives par rapport aux autres membres de la liste des candidats tout en minimisant la dérive, spécialement pour les LAN à grande vitesse avec de nombreux serveurs horaires. La dérive cause des redondances réseau inutiles, car l'intervalle de consultation est calé à sys.poll puisque chaque nouvel homologue est choisi pour la synchronisation et ne s'accroît que lentement lorsque l'homologue n'est plus sélectionné. C'est l'expérience pratique qui a montré que le nombre de candidats survivants à ce point peut devenir assez important et peut résulter en un nombre significatif de cycles de traitement sans améliorer matériellement la stabilité et l'exactitude. En conséquence, la liste de candidats est tronquée à NTP.MAXCLOCK entrées.

Noter que $\varepsilon_{\xi i}$ est la dispersion choisie (d'échantillons) par rapport au $i^{ème}$ homologue représenté sur la liste des candidats, qui peut être calculé de façon similaire à la dispersion de filtre décrite précédemment. Le E_j est la dispersion du $j^{ème}$ homologue représenté sur la liste et inclut des composants dus à l'erreur de mesure, à l'accumulation d'erreur de biais et à la dispersion de filtre. Si le maximum $\varepsilon_{\xi i}$ est supérieur au minimum E_j et que le nombre de survivants est supérieur à NTP.MINCLOCK, le $i^{ème}$ homologue est éliminé de la liste et la procédure se répète. Si la source de synchronisation en cours est un des survivants et qu'il n'y a pas d'autre survivant de strate inférieure, la procédure se termine alors sans aucune autre action. Autrement, la source de synchronisation est réglée sur le premier survivant de la liste des candidats. Dans ce qui suit, *i*, *j*, *k*, *l* sont des indices d'homologues, avec *k* comme indice de la source de synchronisation en cours (NULL si aucune) et *l* l'indice du premier survivant sur la liste des candidats.

```
while début
        pour (chaque survivant [distance, index]) début          /* calculer les dispersions */
                trouver l'indice i pour max ε_ξi ;
                trouver l'indice j pour min E_j ;
                endfor
        si (ε_ξi ≤ E_j ou m ≤ NTP.MINCLOCK) interrompre ;
        peer.survivor [i] ← 0 ;                          /* écarter le i^ème homologue */
        si (i = k) sys.peer ← NULL ;
        supprimer le i^ème homologue de la liste des candidats;
        m ← m - 1 ;
        endwhile
si (peer.survivor [k] = 0> ou peer.stratum [k] > peer.stratum [l]) début
        sys.peer ← l ;                                   /* nouvelle source d'horloge */
        invoquer la mise à jour de consultation;
        endif
fin de la procédure de choix d'horloge;
```

L'algorithme est conçu pour favoriser les homologues qui se trouvent près de la tête de la liste des candidats, qui ont la plus faible strate et distance et peuvent vraisemblablement fournir l'heure la plus exacte et la plus stable. Avec un choix approprié de facteur de pondération *v* (aussi appelé NTP.SELECT), les entrées seront débarrassées de la queue de la liste, sauf si quelques déviants marquent une divergence significative par rapport aux entrées restantes, auquel cas les déviants sont éliminés les premiers. Les conditions de terminaison sont conçues pour éviter les passages inutiles entre sources de synchronisation lorsque cela n'est pas justifié statistiquement, tout en maintenant un préjugé favorable à l'égard des homologues de basse strate et faible distance.

5. Horloges locales

Afin de mettre en œuvre une horloge locale précise et exacte, l'hôte doit être équipé d'une horloge matérielle consistant en un oscillateur et une interface capables de la précision et de la stabilité requises. Une horloge logique est alors construite en utilisant ces composants plus les composants logiciels qui règlent le temps apparent et la fréquence en réponse aux mises à jour périodiques calculées par NTP ou autre protocole de synchronisation de l'heure tels que Hellospeak [MIL83b] ou Unix 4.3bsd TSP [GUS85a]. La présente section décrit le modèle d'horloge locale Fuzzball et sa mise en œuvre, qui incluent des dispositions pour le réglage précis de l'heure et de la fréquence et peuvent maintenir l'heure dans les 15 ns et la fréquence dans les 0,3 ms par jour. Le modèle convient pour une utilisation à la fois avec des oscillateurs à quartz compensés et non compensés et peut être adapté à des oscillateurs à fréquence électrique. Un résumé des caractéristiques de ces types d'oscillateurs et d'autres types figure à l'Appendice E, et une analyse mathématique complète du modèle d'horloge locale NTP figure à l'Appendice G.

Il est important de noter que la mise en œuvre particulière décrite est une des nombreuses possibles qui fournissent des fonctionnalités équivalentes. Cependant, il est également important de noter que le modèle d'horloge décrit à l'Appendice G est qui sert de base à la mise en œuvre implique une sorte particulière de boucle de rétrocontrôle qui est potentiellement instable si les règles de conception sont violées. Le modèle et les paramètres décrits dans l'Appendice G sont conçus pour fournir une heure exacte et stable dans des conditions de fonctionnement typiques utilisant un matériel conventionnel et faire face à des perturbations de la connectivité du matériel ou du réseau. Les paramètres ont été élaborés pour un fonctionnement fiable dans un sous réseau hiérarchique multi niveau où une instabilité de fonctionnement à un niveau peut perturber de nombreux autres niveaux.

5.1 Mise en œuvre Fuzzball

L'horloge locale Fuzzball consiste en une collection de registres matériels et logiciels, conjointement à un ensemble d'algorithmes, qui mettent en œuvre une horloge logique qui fonctionne comme un oscillateur discipliné et se synchronise à une source externe. Une description de ses composants et de sa façon de fonctionner suit. Noter que toute l'arithmétique est en entiers compléments à deux et toutes les touches "<<" et ">>" sont arithmétiques (rempli de signes pour décalages à droite et rempli de zéros pour décalages à gauche). Noter aussi que $x << n$ est équivalent à $x >> -n$.

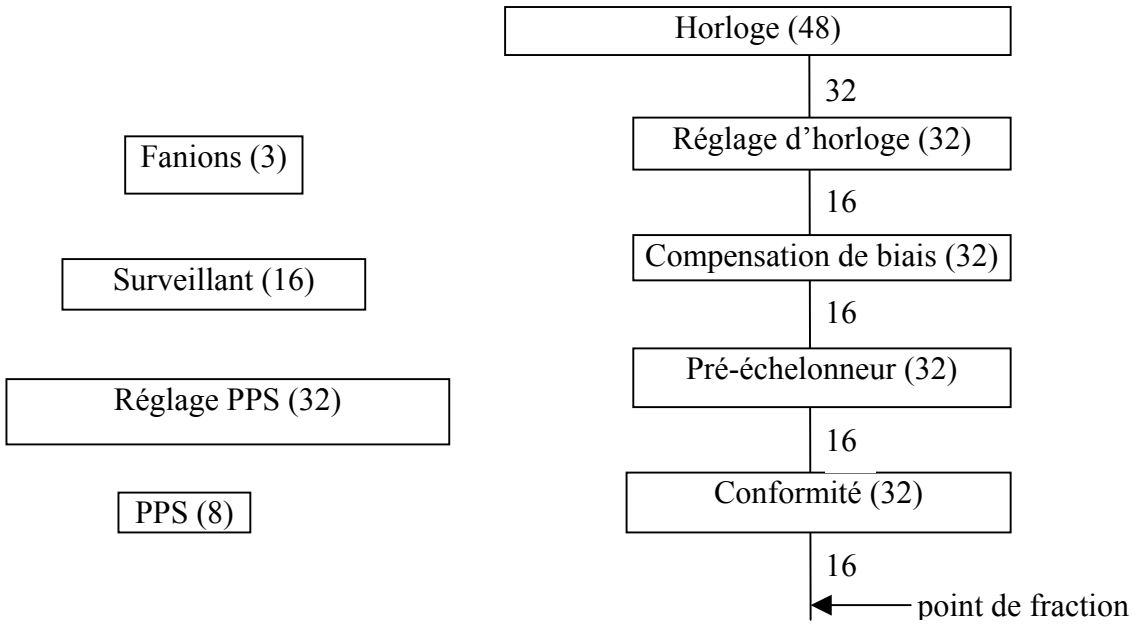

Figure 3. Registres d'horloge

Les principaux composants d'horloge locale sont montrés à la Figure 3, dans laquelle les points de fraction indiqués se rapportent à des millisecondes entières. Le registre d'horloge à 48 bits et le pré-échelonnement à 32 bits fonctionnent comme un oscillateur discipliné qui incrémente en millisecondes par rapport à minuit au point de fraction. Le registre de réglage d'horloge de 32 bits sert à régler la phase d'oscillateur en pas graduels pour éviter les discontinuités dans l'échelle de temps indiquée. Son contenu est désigné par x dans ce qui suit. Le registre de compensation de biais de 32 bits sert à réduire la fréquence de l'oscillateur en ajoutant de petits incréments de phase à des intervalles de réglage périodiques et il peut compenser des erreurs de fréquence jusqu'à 0,1°% ou ± 100 ppm. Son contenu est désigné par y dans ce qui suit. Le compteur de surveillance de 16 bits et le registre de conformité de 32 bits sont utilisés pour

déterminer la validité, ainsi que pour établir la bande passante PLL et l'intervalle de consultation (voir l'Appendice G). Le contenu du registre de conformité est désigné par *z* dans ce qui suit. Le registre de réglage PPS de 32 bits sert à tenir un réglage d'heure de précision lorsqu'une source de 1 – pps impulsions est disponible, alors que le compteur PPS à 8 bits sert à vérifier la présence de ces impulsions. Le registre de fanions de deux bits contient les deux bits de saut décrits par ailleurs (à saut).

Tous les registres excepté le registre Pré échelonnement sont normalement mis en œuvre en mémoire. Dans les conceptions normales d'interface d'horloge tels que DEC KWV11-C, le registre de pré-échelonnement est mis en œuvre comme un compteur de 16 bits en mémoire tampon piloté par un oscillateur à quartz à un multiple de 1000 Hz. Un débordement de compteur est signalé par une interruption, qui résulte en un incrément du registre d'horloge au bit correspondant au débordement. L'heure du jour est déterminée par la lecture du registre de pré-échelonnement, ce qui ne dérange pas le processus de comptage, et par l'ajout de sa valeur à celle du registre d'horloge avec les points de fraction alignés comme indiqué et avec les bits de plus faible poids non mis en œuvre réglés à zéro. Dans d'autres conceptions d'interface, tels que le mécanisme de ligne d'événement LSI-11, chaque tic de l'horloge est signalé par une interruption à des intervalles de 16-2/3 ms ou 20 ms, selon l'interface et la fréquence du réseau d'alimentation électrique. Lorsque cela se produit, on ajoute l'incrément approprié en fractions de millisecondes au registre d'horloge.

Les divers paramètres utilisés sont résumés au Tableau 6, dans lequel certains paramètres ont été rééchelonnés à partir de ceux donnés dans l'Appendice G du fait que les unités y sont en millisecondes. Lorsque le système est initialisé, tous les registres et compteurs sont libérés et les bits de saut sont réglés à 11_2 (non synchronisé). Aux intervalles de réglage de CLOCK.ADJ secondes, CLOCK.ADJ est soustrait du compteur PPS, mais seulement si le contenu précédent du compteur PPS est supérieur à zéro. Aussi, CLOCK.ADJ est ajouté au compteur de surveillance, mais ce dernier est contraint de ne pas dépasser NTP.MAXAGE divisé par CLOCK.ADJ (un jour plein). De plus, si le compteur de surveillance atteint cette valeur, les bits de saut sont mis à 11_2 (non synchronisé).

Paramètre	Nom	Cristal	Secteur
Intervalle de réglage	CLOCK.ADJ	4 s	1 s
Temporisation PPS	CLOCK.PPS	60 s	60 s
Temporisation d'étape	CLOCK.MINSTEP	900 s	900 s
Ouverture maximale	CLOCK.MAX	± 128 ms	± 512 ms
Pondération de fréquence	CLOCK.FREQ	16	16
Pondération de phase	CLOCK.PHASE	8	9
Pondération de conformité	CLOCK.WEIGHT	13	13
Maximum de conformité	CLOCK.COMP	4	4
Multiplicateur de conformité	CLOCK.MULT	4	4

Tableau 6. Paramètres d'horloge

Dans certaines configurations de système, une source précise d'informations horaires est disponible sous la forme d'un train d'impulsions horaires espacées d'intervalles d'une seconde. Habituellement, cela vient en plus d'une source d'informations de code horaire, telle qu'une horloge radio ou même NTP, pour dénombrer les secondes, minutes, heures et jours. Dans une interface d'horloge typique telle que DEC KWV11-C, une entrée spéciale est fournie pour déclancher une interruption à la réception de chaque impulsion. Lorsque cela arrive, le compteur PPS est réglé à CLOCK.PPS et le décalage horaire en cours est déterminé de la façon habituelle. Puis le registre de réglage PPS est réglé au décalage horaire échelonné en millisecondes. Finalement, si le registre de réglage PPS est supérieur ou égal à 500, 1000 est soustrait de son contenu. Comme décrit plus loin, le registre de réglage PPS et les compteurs PPS peuvent être utilisé conjointement avec un code horaire ordinaire pour produire une horloge locale extrêmement exacte.

5.2 Réglages de phase graduels

Laissée non corrigée, l'horloge locale tourne avec le décalage et la fréquence résultant de sa dernière mise à jour. Une mise à jour est produite par un événement qui résulte en un choix d'horloge valide. Il consiste en un entier signé *(algébrique)* de 48 bits en millisecondes entières et fractions, avec le point de fraction à la gauche du 32. Si la magnitude est supérieure à l'ouverture maximum CLOCK.MAX, une étape de réglage est nécessaire, auquel cas on procède comme décrit plus loin. Autrement, on effectue une phase de réglage graduel. Normalement, la mise à jour est calculée par les algorithmes NTP décrits plus haut ; cependant, si le compteur PPS est supérieur à zéro, la valeur du registre de réglage PPS est utilisée à la place. Soit *u* une quantité de 32 bits avec les bits 0 à 31 réglés comme les bits 16 à 47 de la mise à jour. Si certains des bits de plus faible poids de la mise à jour ne sont pas mis en œuvre, ils sont réglés comme la valeur du bit de signe. Ces opérations déplacent le point de fraction de *u* à la gauche du bit 16 et minimisent les effets de la troncature et des erreurs d'arrondi. Soit *b* le nombre de zéros en tête de la valeur absolue du registre de conformité et soit *c* le nombre de zéros en tête du compteur de surveillance, qui sont tous deux facilement calculés par des boucles compactes. Mettons alors *b* à $b = b - 16 + \text{CLOCK.COMP}$ et fixons qu'il ne soit pas inférieur à zéro. Cela

représente le logarithme de la constante de boucle de temps. Puis réglons c à $c = 10 - c$ et fixons qu'il ne soit pas supérieur à NTP.MAXPOLL – NTP.MINPOLL. Cela représente le logarithme de l'intervalle d'intégration depuis la dernière mise à jour. Les valeurs fixées assurent un fonctionnement stable dans les conditions normales rencontrées dans l'Internet. Puis calculons les nouvelles valeurs pour les registres de réglage d'horloge et de compensation de biais

$$x = u >> b ,$$
$$y = y + (u >> (b + b – c)) .$$

Finalement, calculons la moyenne exponentielle

$$z = z + (u << (b + CLOCK.MULT) – z) >> CLOCK.WEIGHT ,$$

où le glissement à gauche réaligne le point de fraction pour une plus grande précision et faciliter le calcul.

A chaque intervalle de réglage, la correction d'horloge finale consistant en deux composants est déterminée. Le premier composant (la phase) comporte la quantité x >> CLOCK.PHASE , qui est alors soustraite du précédent contenu du registre de réglage d'horloge pour former le nouveau contenu de ce registre. Le second composant (fréquence) comporte la quantité y >> CLOCK.FREQ.

La somme des composants phase et fréquence est la correction finale d'horloge, qui est alors ajoutée au registre d'horloge. Finalement, le compteur de surveillance est mis à zéro. L'opération continue de cette façon jusqu'à ce qu'une nouvelle correction soit introduite.

La valeur de b calculée ci-dessus peut être utilisée pour mettre à jour la variable de système d'intervalle de consultation (sys.poll). Cela fonctionne comme un paramètre adaptif fournissant une caractéristique très précieuse qui réduit la redondance de consultation, en particulier si l'algorithme de combinaison d'horloge décrit à l'Appendice F est utilisé :

$$\text{sys.poll} \leftarrow b + \text{NTP.MINPOLL}$$

Dans des conditions où le bruit de mise à jour est élevé ou si la fréquence de l'oscillateur matériel change relativement rapidement du fait de conditions environnementales, la magnitude de la conformité s'accroît. Avec les paramètres spécifiés, cela cause l'augmentation de la largeur de bande de boucle (réciproque de la constante de temps) et la diminution de l'intervalle de consultation, éventuellement jusqu'à NTP.MINPOLL secondes. Lorsque le bruit est faible et que l'oscillateur matériel est très stable, la conformité décroît, ce qui cause la décroissance de la largeur de bande de boucle et l'accroissement de l'intervalle de consultation, éventuellement jusqu'à NTP.MAXPOLL secondes.

Les paramètres du Tableau 6 ont été choisis de telle sorte que, dans de bonnes conditions avec des mises à jour de l'ordre de quelques millisecondes, une précision d'une milliseconde par jour (environ 0,1 ppm ou 10^{-8}), puisse être réalisée. Il faut veiller dans la mise en œuvre à s'assurer de la monotonie du registre d'horloge et à préserver la plus grande précision tout en minimisant la propagation des erreurs d'arrondi. Comme toutes les opérations de multiplication/division (excepté celles impliquées avec les impulsions 1-pps) calculées en temps réel peuvent être approximées par des opérations de glissement de bit, il n'est pas nécessaire de mettre en œuvre une pleine capacité de multiplication/division dans le matériel ou le logiciel.

Dans les diverses mises en œuvre de NTP pour de nombreux systèmes fondés sur Unix, il a été couramment expérimenté que le seul plus important facteur affectant la stabilité de l'horloge locale est la correspondance des coefficients de phase et de fréquence avec la mise en œuvre particulière du noyau. Il est vital que ces coefficients soient élaborés conformément aux valeurs du modèle, car autrement le PLL peut échouer à retracer les variations normales de l'oscillateur et peut même devenir instable.

5.3 Ajustements de phase d'allure

Lorsque la magnitude d'une correction excède l'ouverture maximum CLOCK.MAX, il existe une possibilité que l'horloge soit tellement éloignée de la synchronisation avec la source de référence que la meilleure action soit un remplacement immédiat et complet du contenu de registre d'horloge, plutôt que des ajustements graduels comme décrit ci-dessus. Cependant, dans les cas où la variance des échantillons est extrêmement élevée, il est prudent de ne pas croire à un changement d'allure, à moins qu'un intervalle significatif ne se soit écoulé depuis le dernier réglage graduel. Donc si un changement d'allure est indiqué et si le compteur de surveillance est inférieur à la valeur pré configurée CLOCK.MINSTEP, la mise à jour est ignorée et la procédure d'horloge locale se termine. Ces sauvegardes sont particulièrement utiles dans les configurations de systèmes qui utilisent une horloge atomique calibrée ou un récepteur LORAN-C conjointement à une source séparée d'informations de dénombrement de secondes, tels qu'une horloge radio ou un homologue NTP.

Si un changement d'allure est indiqué, la mise à jour est ajoutée directement au registre d'horloge et le registre de réglage d'horloge et le compteur de surveillance sont tous deux mis à zéro, mais les autres registres sont laissés sans changement. Comme un changement d'allure invalide les données présentes dans les filtres d'horloge, les bits de saut

sont mis à 11_2 (non synchronisé) et, comme décrit ailleurs, la procédure de libération est invoquée pour purger les filtres d'horloge et les variables d'état pour tous les homologues. En pratique, la nécessité d'effectuer un changement d'allure est rare et survient généralement lorsque l'hôte local ou une source de référence est réamorcée, par exemple. Ceci est heureux car les changements d'allure peuvent avoir pour résultat que l'horloge locale semble marcher à reculons, ainsi que des délais et des mesures de décalage incorrects du mécanisme de synchronisation lui-même.

Une expérience considérable de l'environnement de l'Internet suggère que les valeurs de CLOCK.MAX figurant au Tableau 6 sont appropriées. En pratique, ces valeurs ne sont dépassées par une seule source de serveur horaire que dans des conditions de plus extrême encombrement ou lorsque plusieurs défaillances de nœuds ou liaisons sont survenues. Le cas le plus commun où le maximum est dépassé est lorsque la source de serveur horaire est changée et que l'heure indiquée par les nouvelle et ancienne sources dépassent le maximum du fait d'erreurs systématiques dans la source de référence primaire ou de grandes différences dans le délais d'acheminement. Il est recommandé que les mises en œuvre incluent des dispositions pour tailler CLOCK.MAX sur mesure pour les situations spécifiques. La quantité dont CLOCK.MAX peut être augmenté sans violer l'exigence de monotonie dépend de l'incrément du registre d'horloge. Pour un incrément de 10 ms, comme utilisé dans de nombreuses stations de travail, la valeur indiquée au Tableau 6 peut être augmentée d'un facteur cinq.

5.4 Questions de mise en œuvre

Le modèle de robustesse NTP de base est qu'un hôte n'a pas d'autre moyen de vérifier l'heure que NTP lui-même. Dans certains équipements, une horloge/calendrier à batterie de secours est disponible pour une vérification de bonne santé. Si un tel dispositif est disponible, il ne devrait être utilisé que pour confirmer la bonne santé du système de conservation de l'heure, et non comme source de l'heure du système. Avec l'hypothèse commune (pas toujours justifiée) que l'horloge/calendrier est plus fiable mais moins exacte que le sous réseau de synchronisation NTP, l'approche recommandée à l'initialisation est de régler le registre d'horloge comme déterminé à partir de l'horloge/calendrier et les autres registres, compteurs et fanions comme décrit ci-dessus. Pour les mises à jour ultérieures, si le décalage de temps est plus grand qu'un paramètre de configuration (par exemple, 1000 secondes), la mise à jour doit alors être écartée et une condition d'erreur rapportée. Certaines mises en œuvre enregistrent périodiquement le contenu du registre de compensation de biais dans une mémoire stable comme un fichier système ou NVRAM et restituent cette valeur à l'initialisation. Cela peut significativement réduire le temps pour converger vers la stabilité nominale et le régime d'exactitude.

La conversion du format NTP vers les formats communs de date et d'heure utilisés par les programmes d'application est simplifiée si le format interne d'horloge locale utilise des variables de date et heure séparées. La variable d'heure est conçue pour tourner sur 24 heures, prendre ou rendre une seconde sautée comme déterminé par les bits d'indicateur de saut, avec ses débordements (ou manques) qui incrémentent (décrémentent) la variable de date. Les variables de date et heure indiquent alors le nombre de jours et de secondes depuis la référence horaire précédente, mais non corrigée pour les secondes sautées à venir.

Le jour avant l'insertion d'une seconde sautée, les bits de saut (sys.leap) sont réglés sur les serveurs primaires, vraisemblablement manuellement. Ensuite, ces bits apparaissent à l'hôte local et sont passés à la procédure d'horloge locale. Cela cause l'augmentation ou la diminution d'une seconde selon le cas approprié du module de la variable de temps, qui est la longueur du jour en cours. Les bits de saut sont remis à zéro immédiatement après l'insertion. On trouvera des éléments de discussion supplémentaires sur cette question à l'Appendice E.

Le manque d'un mécanisme complet pour gérer les bits de saut dans les serveurs primaires est présentement un problème fâcheux qu'il conviendrait qu'un mécanisme de gestion de réseau complet à développer prenne en charge. D'un point de vue pratique et tant que des dispositions spécifiques n'auront pas été prises à cet effet, les fabricants d'horloges radio n'ont pas de dispositifs pour les secondes sautées, ni automatiques ni manuels. Et donc, lorsque survient réellement une seconde sautée, l'horloge radio doit se resynchroniser au code horaire diffusé, ce qui peut prendre un temps variant de quelques minutes à quelques heures. Sauf dispositions spéciales, un serveur primaire peut sauter à la nouvelle échelle horaire, pour être ramené à l'échelle horaire précédente lorsque il se synchronise à nouveau avec la radio. Ultérieurement, le serveur sera ramené en avant à nouveau lorsque la radio elle-même se sera resynchronisée avec le code horaire diffusé.

Ce problème ne peut être évité de façon fiable en utilisant quelque algorithme de sélection que ce soit, car il existera toujours un intervalle d'au moins quelques minutes et peut-être de plusieurs heures lorsque certaines horloges radio, ou toutes, seront désynchronisées d'avec le code horaire diffusé et c'est seulement après que la majorité d'entre elles se seront resynchronisées que le sous réseau trouvera le calme. Le délai CLOCK.MINSTEP est conçu pour tenir compte de ce problème en forçant un intervalle minimum depuis le dernier réglage graduel effectué avant d'autoriser un changement d'allure. Donc, jusqu'à ce que la radio se resynchronise, on continuera avec la vieille échelle de temps, qui est divergente d'une seconde d'avec l'horloge locale après le saut et en-dehors de l'ouverture maximum CLOCK.MAX permise pour les réglages de phase graduels. Lorsque la radio se resynchronise finalement, elle va presque certainement

venir dans l'ouverture et redevenir une source de référence. Et donc, même dans le cas peu vraisemblable où l'horloge locale fait le saut de façon incorrecte, le serveur ne mettra pas plus de CLOCK.MINSTEP secondes avant de se resynchroniser.

6. Remerciements

De nombreuses personnes ont contribué au contenu de ce document, qui a fait l'objet de débats minutieux par messagerie électronique et a été débogué en utilisant deux différents prototypes de mise en œuvre pour le système d'exploitation Unix 4.3bsd, l'un écrit par Louis Mamakos et Michael Petry de l'Université du Maryland et l'autre par Dennis Ferguson de l'Université de Toronto. Une autre mise en œuvre pour le système d'exploitation Fuzzball [MIL88b] a été écrite par l'auteur. De nombreux individus, trop nombreux à mentionner, ont méticuleusement testé les versions prototypes de beta-test et ont éliminé les bogues sans pitié, aussi bien dans le code que dans la spécification. Particulièrement utiles ont été les commentaires de Dennis Ferguson et de Bill Sommerfeld, ainsi que les discussions avec Joe Comuzzi et quelques autres de Digital Equipment Corporation.

7. Références

[ABA89] Abate, et al. AT&T's new approach to the synchronization of telecommunication networks. IEEE Communications Magazine (avril 1989), 35-45.

[ALL74a] Allan, D.W., J.H. Shoaf and D. Halford. Statistics of time and frequency data analysis. In: Blair, B.E. (Ed.). Time and Frequency Theory and Fundamentals. National Bureau of Standards Monograph 140, U.S. Department of Commerce, 1974, 151-204.

[ALL74b] Allan, D.W., J.E. Gray and H.E. Machlan. The National Bureau of Standards atomic time scale: generation, stability, accuracy and accessibility. In: Blair, B.E. (Ed.). Time and Frequency Theory and Fundamentals. National Bureau of Standards Monograph 140, U.S. Department of Commerce, 1974, 205-231.

[BEL86] Bell Communications Research. Digital Synchronization Network Plan. Technical Advisory TA-NPL-000436, 1er novembre 1986.

[BER87] Bertsekas, D., and R. Gallager. Data Networks. Prentice-Hall, Englewood Cliffs, NJ, 1987.

[BLA74] Blair, B.E. Time and frequency dissemination: an overview of principles and techniques. In: Blair, B.E. (Ed.). Time and Frequency Theory and Fundamentals. National Bureau of Standards Monograph 140, U.S. Department of Commerce, 1974, 233-314.

[BRA80] Braun, W.B. Short term frequency effects in networks of coupled oscillators. IEEE Trans. Communications COM-28, 8 (août 1980), 1269- 1275.

[COL88] Cole, R., and C. Foxcroft. An experiment in clock synchronisation. Computer Journal 31, 6 (1988), 496-502.

[DAR81a] Defense Advanced Research Projects Agency. Internet Protocol. DARPA Network Working Group Report RFC-791, USC Information Sciences Institute, septembre 1981.

[DAR81b] Defense Advanced Research Projects Agency. Internet Control Message Protocol. DARPA Network Working Group Report RFC-792, USC Information Sciences Institute, septembre 1981.

[DEC89] Digital Time Service Functional Specification Version T.1.0.5. Digital Equipment Corporation, 1989.

[DER90] Dershowitz, N., and E.M. Reingold. Calendrical Calculations. Software Practice and Experience 20, 9 (septembre 1990), 899-928.

[FRA82] Frank, R.L. History of LORAN-C. Navigation 29, 1 (printemps 1982).

[GUS84] Gusella, R., and S. Zatti. TEMPO - A network time controller for a distributed Berkeley UNIX system. IEEE Distributed Processing Technical Committee Newsletter 6, NoSI-2 (juin 1984), 7-15. Also in: Proc. Summer USENIX Conference (June 1984, Salt Lake City).

[GUS85a] Gusella, R., and S. Zatti. The Berkeley UNIX 4.3BSD time synchronization protocol: protocol specification. Technical Report UCB/CSD 85/250, University of California, Berkeley, juin 1985.

[GUS85b] Gusella, R., and S. Zatti. An election algorithm for a distributed clock synchronization program. Technical Report UCB/CSD 86/275, University of California, Berkeley, décembre 1985.

[HAL84] Halpern, J.Y., B. Simons, R. Strong and D. Dolly. Fault-tolerant clock synchronization. Proc. Third Annual ACM Symposium on Principles of Distributed Computing (août 1984), 89-102.

[JOR85] Jordan, E.C. (Ed). Reference Data for Engineers, Seventh Edition. H.W. Sams & Co., New York, 1985.

[KOP87] Kopetz, H., and W. Ochsenreiter. Clock synchronization in distributed real-time systems. IEEE Trans. Computers C-36, 8 (août 1987), 933-939.

[LAM78] Lamport, L., Time, clocks and the ordering of events in a distributed system. Comm. ACM 21, 7 (juillet 1978), 558-565.

[LAM85] Lamport, L., and P.M. Melliar-Smith. Synchronizing clocks in the presence of faults. J. ACM 32, 1 (janvier 1985), 52-78.

[LIN80] Lindsay, W.C., and A.V. Kantak. Network synchronization of random signals. IEEE Trans. Communications COM-28, 8 (août 1980), 1260-1266.

[LUN84] Lundelius, J., and N.A. Lynch. A new fault-tolerant algorithm for clock synchronization. Proc. Third Annual ACM Symposium on Principles of Distributed Computing (août 1984), 75-88.

[MAR85] Marzullo, K., and S. Owicki. Maintaining the time in a distributed system. ACM Operating Systems Review 19, 3 (juillet 1985), 44-54.

[MIL81a] Mills, D.L. Time Synchronization in DCNET Hosts. DARPA Internet Project Report IEN-173, COMSAT Laboratories, février 1981.

[MIL81b] Mills, D.L. DCNET Internet Clock Service. DARPA Network Working Group Report RFC-778, COMSAT Laboratories, avril 1981.

[MIL83a] Mills, D.L. Internet Delay Experiments. DARPA Network Working Group Report RFC-889, M/A-COM Linkabit, décembre 1983.

[MIL83b] Mills, D.L. DCN local-network protocols. DARPA Network Working Group Report RFC-891, M/A-COM Linkabit, décembre 1983.

[MIL85a] Mills, D.L. Algorithms for synchronizing network clocks. DARPA Network Working Group Report RFC-956, M/A-COM Linkabit, septembre 1985.

[MIL85b] Mills, D.L. Experiments in network clock synchronization. DARPA Network Working Group Report RFC-957, M/A-COM Linkabit, septembre 1985.

[MIL85c] Mills, D.L. Network Time Protocol (NTP). DARPA Network Working Group Report RFC-958, M/A-COM Linkabit, septembre 1985.

[MIL88a] Mills, D.L. Network Time Protocol (version 1) - specification and implementation. DARPA Network Working Group Report RFC-1059, University of Delaware, juillet 1988.

[MIL88b] Mills, D.L. The Fuzzball. Proc. ACM SIGCOMM 88 Symposium (Palo Alto, CA, août 1988), 115-122.

[MIL89] Mills, D.L. Network Time Protocol (version 2) - specification and implementation. DARPA Network Working Group Report RFC-1119, University of Delaware, septembre 1989.

[MIL90] Mills, D.L. Measured performance of the Network Time Protocol in the Internet system. ACM Computer Communication Review 20, 1 (janvier 1990), 65-75.

[MIL91a] Mills, D.L. Internet time synchronization: the Network Time Protocol. IEEE Trans. Communications 39, 10 (octobre 1991), 1482-1493.

[MIL91b] Mills, D.L. On the chronology and metrology of computer network timescales and their application to the Network Time Protocol. ACM Computer Communications Review 21, 5 (octobre 1991), 8-17.

[MIT80] Mitra, D. Network synchronization: analysis of a hybrid of master-slave and mutual synchronization. IEEE Trans. Communications COM-28, 8 (août 1980), 1245-1259.

[NBS77] Data Encryption Standard. Federal Information Processing Standards Publication 46. National Bureau of Standards, U.S. Department of Commerce, 1977.

[NBS79] Time and Frequency Dissemination Services. NBS Special Publication 432, U.S. Dep. of Commerce, 1979.

[NBS80] DES Modes of Operation. Federal Information Processing Standards Publication 81. National Bureau of Standards, U.S. Department of Commerce, décembre 1980.

[POS80] Postel, J. User Datagram Protocol. DARPA Network Working Group Report RFC-768, USC Information Sciences Institute, août 1980.

[POS83a] Postel, J. Daytime protocol. DARPA Network Working Group Report RFC-867, USC Information Sciences Institute, mai 1983.

[POS83b] Postel, J. Time protocol. DARPA Network Working Group Report RFC-868, USC Information Sciences Institute, mai 1983.

[RIC88] Rickert, N.W. Non Byzantine clock synchronization - a programming experiment. ACM Operating Systems Review 22, 1 (janvier 1988), 73-78.

[SCH86] Schneider, F.B. A paradigm for reliable clock synchronization. Department of Computer Science Technical Report TR 86-735, Cornell University, février 1986.

[SMI86] Smith, J. Modern Communications Circuits. McGraw-Hill, New York, NY, 1986.

[SRI87] Srikanth, T.K., and S. Toueg. Optimal clock synchronization. J. ACM 34, 3 (juillet 1987), 626-645.

[STE88] Steiner, J.G., C. Neuman, and J.I. Schiller. Kerberos: an authentication service for open network systems. Proc. Winter USENIX Conference (février 1988).

[SU81] Su, Z. A specification of the Internet protocol (IP) timestamp option. DARPA Network Working Group Report RFC-781. SRI International, mai 1981.

[TRI86] Tripathi, S.K., and S.H. Chang. ETempo: a clock synchronization algorithm for hierarchical LANs - implementation and measurements. Systems Research Center TR-86-48, University of Maryland, 1986.

[VAN84] Van Dierendonck, A.J., and W.C. Melton. Applications of time transfer using NAVSTAR GPS. In: Global Positioning System, Papers Published in Navigation, Vol. II, Institute of Navigation, Washington, DC, 1984.

[VAS78] Vass, E.R. OMEGA navigation system: present status and plans 1977-1980. Navigation 25, 1 (printemps 1978).

Note du traducteur : on trouvera des éléments intéressants en français sur les sites
http://alexandre.alapetite.net/iup-gmi/ntp
http://taxilevesinet.free.fr/GPS.htm
le premier étant un résumé de NTP, et le second un exposé sur le système horaire du GPS. CBI.

Appendice A Format de données NTP - version 3

Le format de la zone de données de message NTP, qui suit immédiatement l'en-tête UDP, est indiqué à la Figure 4. Ci-après figure une description de ses champs.

Indicateur de saut (LI) : C'est un code de deux bits qui avertit de l'arrivée d'un saut de seconde à insérer/supprimer dans la dernière minute du jour en cours, avec le bit 0 et le bit 1, respectivement codés comme suit :

00	pas d'avertissement
01	la dernière minute a 61 secondes
10	la dernière minute a 59 secondes
11	condition d'alarme (horloge non synchronisée)

Numéro de version (VN) : Entier de trois bits qui indique le numéro de version NTP, actuellement trois (3).

Mode : Entier de trois bits indiquant le mode, avec des valeurs définies comme suit :

0	réservé
1	actif symétrique
2	passif symétrique
3	client
4	serveur
5	diffusion
6	réservé pourle message de contrôle NTP (voir l'Appendice B)
7	réservé pour utilisation privée

Strate : Entier de huit bits qui indique le niveau de strate de l'horloge locale, avec des valeurs définies comme suit :

0	non spécifié
1	référence primaire (par exemple, horloge radio)
2-255	référence secondaire (via NTP)

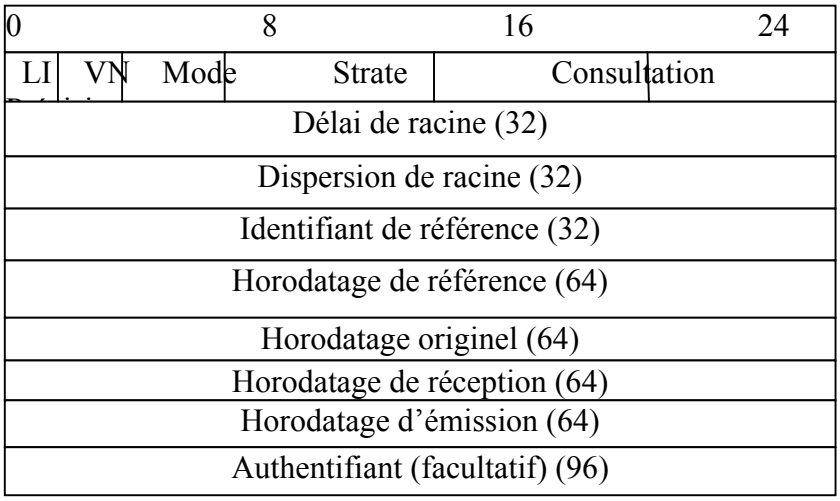

Figure 4. En-tête de message NTP

Les valeurs qui peuvent apparaître dans ce champ vont de zéro à NTP.INFIN inclus.

Intervalle de consultation : Entier signé de huit bits qui indique l'intervalle maximum entre des messages successifs, en secondes jusqu'à la plus proche puissance de deux. Les valeurs qui peuvent apparaître dans ce champ vont de NTP.MINPOLL à NTP.MAXPOLL inclus.

Précision : Entier signé de huit bits indiquant la précisison de l'horloge locale, en secondes jusqu'à la plus proche puissance de deux.

Délai de racine : Nombre algébrique de 32 bits à virgule fixe qui indique le délai total d'aller-retour jusqu'à la source de référence primaire, en secondes avec le point de fraction entre les bits 15 et 16. Noter que cette variable peut prendre des valeurs positives et négatives, selon la précision et le biais de l'horloge.

Dispersion de racine : Nombre algébrique de 32 bits à virgule fixe qui indique l'erreur maximum par rapport à la source de référence primaire, en secondes avec le point de fraction entre les bits 15 et 16. Seules les valeurs positives supérieures à zéro sont possibles.

Identifiant d'horloge de référence : Code de 32 bits qui identifie l'horloge de référence particulière. Dans le cas de la strate 0 (non spécifiée) ou de la strate 1 (référence primaire), c'est une chaîne ASCII de quatre octets, justifiée à gauche, bourrée de zéros. Bien que ne figurant pas au titre de la spécification NTP, on suggère les identifiants ASCII suivants :

Strate	**Code**	**Signification**
0	DCN	Protocole d'acheminement DCN
0	NIST	Modem public NIST
0	TSP	Protocole horaire TSP
0	DTS	Service de l'heure numérique
1	ATOM	Horloge atomique (calibrée)
1	VLF	Radio VLF (OMEGA, etc.)
1	callsign	Radio générique
1	LORC	Radionavigation LORAN-C
1	GOES	Satellite d'environnement UHF GOES
1	GPS	Positionnement par satellite UHF GPS

Dans le cas de la strate 2 et au dessus (référence secondaire) c'est l'adresse Internet à quatre octets de l'hôte de référence primaire.

Horodatage de référence : C'est l'heure locale à laquelle l'horloge locale a été réglée ou corrigée, en format d'horodatage à 64 bits.

Horodatage originel : C'est l'heure locale à laquelle la demande est partie de chez l'hôte client pour l'hôte de service, en format d'horodatage à 64 bits.

Horodatage de réception : C'est l'heure locale à laquelle la demande est arrivée à l'hôte de service, en format d'horodatage à 64 bits.

Horodatage d'émission : C'est l'heure locale à laquelle la réponse est partie de l'hôte de service pour l'hôte client, en format d'horodatage à 64 bits.

Authentifiant (facultatif) : Lorsque le mécanisme d'authentification NTP est mis en œuvre, il contient les informations sur l'authentifiant défini à l'Appendice C.

Appendice B Messages de contrôle NTP

Dans un environnement de gestion de réseau complet, on suppose que des facilités sont disponibles pour effectuer les fonctions de routine de contrôle et surveillance NTP, telles que de régler les bits d'indicateur de saut chez les serveurs primaires, de régler les divers paramètres de système et de surveiller les opérations régulières. Normalement, ces fonctions peuvent être mises en œuvre en utilisant un protocole de gestion de réseau tel que SNMP et des extensions convenables à la base de données MIB. Cependant, dans les cas où de telles facilités ne sont pas disponibles, ces fonctions peuvent être mises en œuvre en utilisant les messages de commande spéciaux de NTP décrits ci-après. Ces messages sont seulement destinés à être utilisés dans les systèmes où aucune autre facilité de gestion n'est disponible ou appropriée, comme dans les périphériques de bus à fonctions dédiées. La prise en charge de ces messages n'est pas exigée pour la conformité à la présente spécification.

Le message de commande NTP a la valeur 6 spécifiée dans le champ de mode du premier octet de l'en-tête NTP et est formaté comme indiqué ci-dessous. Le format du champ données est spécifique de chaque commande ou réponse ; cependant, dans la plupart des cas, le format est conçu pour être construit et vu par l'homme et est donc codé en formes libres ASCII. Cela facilite la spécification et la mise en œuvre d'outils de gestion simples en l'absence de facilités de gestion de réseau évoluées. Comme dans les messages NTP ordinaires, le champ authentifiant suit le champ des données. Si l'authentifiant est utilisé, le champ des données est bourré de zéros jusqu'à une limite de 32 bits, mais les

bits de bourrage ne sont pas considérés comme faisant partie du champ de données et ne sont pas inclus dans le compte du champ.

Les hôtes IP ne sont pas obligés de constituer des datagrammes de plus de 576 octets ; cependant, certaines commandes ou réponses peuvent impliquer plus de données que n'en contient un seul datagramme. Par conséquent, un dispositif simple de réassemblage est inclus, par lequel chaque octet des données du message est numéroté à partir de zéro. Au fur et à mesure que chaque fragment est transmis, le numéro de son premier octet est inséré dans le champ décalage et le nombre des octets est inséré dans le champ compte. Le bit données à venir (M) est mis dans tous les fragments sauf le dernier.

La plupart des fonctions de commande impliquent l'envoi d'une commande et la réception d'une réponse, qui peut impliquer plusieurs fragments. L'expéditeur choisit un numéro de séquence distinct, différent de zéro, et met le champ d'état et les bits R et E à zéro. Le répondant interprête le opcode et les informations supplémentaires dans le champ données, met à jour le champ état, met le bit R à un et retourne les mots de 32 bits de l'en-tête avec les informations supplémentaires dans le champ données. Dans le cas de format ou contenu de message invalide, le répondant insère un code dans le champ état, met les bits R et E à un et facultativement insère un message de diagnostic dans le champ données.

Certaines commandes lisent ou écrivent des variables système et des variables d'homologue pour une association identifiée dans la commande. D'autres lisent ou écrivent des variables associées à une horloge radio ou autre appareil directement connecté à une source d'informations de synchronisation primaire. Un identifiant d'association de 16 bits est utilisé pour identifier le type de variable et d'association. Les variables de système sont indiquées par l'identifiant zéro. Comme chaque association est constituée, un identifiant unique, différent de zéro est créé pour elle. Ces identifiants sont utilisés de façon cyclique, de sorte que la probabilité d'utiliser un vieil identifiant qui corresponde à une association nouvellement créée soit faible. Une entité de gestion peut demander une liste des identifiants en cours et les utiliser ensuite pour lire et écrire les variables de chaque association. Une tentative d'utilisation d'un identifiant périmé a pour résultat une réponse d'exception, après quoi la liste peut être demandée à nouveau.

Certains événements d'exception, tels que lorsque un homologue devient atteignable ou inatteignable, surviennent spontanément et ne sont pas nécessairement associés à une commande. Une mise en œuvre peut choisir de sauvegarder les informations d'événement pour une restitution ultérieure, ou pour envoyer une réponse asynchrone (appelée un piège *[trap]*) ou les deux. En cas de piège, l'adresse IP et le numéro de port sont déterminés par une commande antérieure et le champ de séquence est réglé comme décrit plus loin. Les informations sur l'état en cours et de résumé pour le dernier événement d'exception sont retournées dans les réponses normales. Les bits dans le champ d'état indiquent si une exception est survenue depuis la dernière réponse et si plus d'une exception est survenue.

Les commandes ne sont pas nécessairement envoyées par un homologue NTP, aussi les procédures ordinaires de contrôle d'accès peuvent ne pas s'appliquer ; cependant le mécanisme facultatif de gabarit/correspondance suggéré plus haut dans le présent document fournit la capacité de contrôler l'accès par le numéro de mode, de sorte que cela peut être utilisé pour limiter l'accès pour les messages de commande (mode 6) à des gammes d'adresses choisies.

B.1 Format de message de commande

Le format de l'en-tête de message de commande NTP, qui suit immédiatement l'en-tête UDP, est montré à la Figure 5 . Une description de ses champs suit. Les positions binaires marquées zéro sont réservées et devraient toujours être transmises comme zéro.

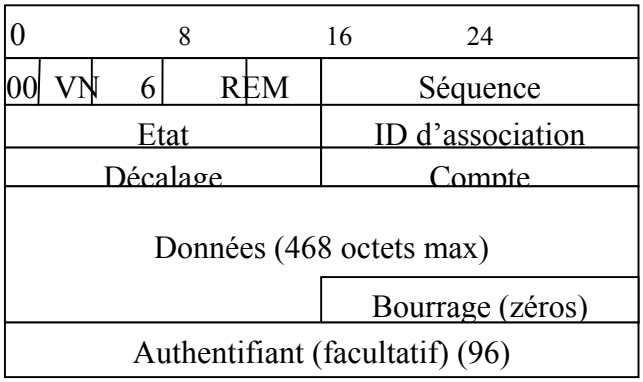

Figure 5. En-tête de message de contrôle NTP

Numéro de version (VN) : Entier de trois bits qui indique le numéro de version NTP, actuellement trois (3).

Mode : Entier de trois bits qui indique le mode. Il doit avoir la valeur 6, qui indique un message de commande NTP.

Bit de réponse (R) : Mis à zéro pour la commande, à un pour les réponses.

Bit d'erreur (E) : Mis à zéro pour une réponse normale, à un pour une réponse d'erreur.

Plus de bits (M) : Mis à zéro pour le dernier fragment, à un pour tous les autres.

Code de fonctionnement (Op) : Entier de cinq bits qui spécifie la fonction de commande. Les valeurs actuellement définies incluent les suivantes :

0	réservé
1	lire la commande/réponse d'état
2	lire la commande/réponse de variables
3	écrire la commande/réponse de variables
4	lire la commande/réponse de variables d'horloge
5	écrire la commande/réponse de variables d'horloge
6	régler la commande/réponse d'adresse/port de piège
7	réponse de piège
8 à 31	réservé

Séquence : Entier de 16 bits qui indique le numéro de séquence de la commande ou réponse.

Etat : Code de 16 bits qui indique l'état en cours du système, homologue ou horloge, avec des valeurs codées comme décrit dans les paragraphes suivants.

ID d'association : Entier de 16 bits qui identifie une association valide.

Décalage : Entier de 16 bits qui indique le décalage, en octets, du premier octet dans la zone de données.

Compte : Entier de 16 bits qui indique la longueur du champ de données, en octets.

Données : Il contient les données du message de commande ou de réponse. Le nombre maximum d'octets de données est de 468.

Authentifiant (facultatif) : Lorsque le mécanisme d'authentification NTP est mis en œuvre, cela contient les informations d'authentifiant définies à l'Appendice C.

B.2 Avis d'état

Les avis d'état indiquent l'état présent du système, des associations et horloges. Ils sont conçus pour être interprétés par les programmes de surveillance de réseau et sont dans un des quatre formats de 16 bits montrés à la Figure 6 et décrits dans la présente section. Les avis d'état de système et d'homologue sont associés aux réponses pour toutes les commandes excepté les variable de lecture d'horloge, les variables d'ériture d'horloge et les commandes d'établissement de piège d'adresse/port. L'identifiant d'association zéro spécifie l'avis d'état de système, tandis que un identifiant différent de zéro spécifie une association d'homologue particulière. L'avis d'état retourné en réponse à des variables de lecture d'horloge et à des variables d'écriture d'horloge indiquent l'état d'un matériel d'horloge et d'un logiciel de décodage. Un avis spécial d'état d'erreur est utilisé pour apporter des champs de commande malformés ou des valeurs invalides.

B.2.1 Avis d'état de système

L'avis d'état de système apparaît dans le champ état de la réponse à une commande d'état de lecture ou de variables de lecture avec un identifiant d'association zéro. Le format de l'avis d'état de système est le suivant :

Indicateur de saut (LI) : Avertissement codé de deux bits d'une seconde sautée à venir à insérer/supprimer dans la dernière minute du jour en cours, avec le bit 0 et le bit 1, codés respectivement comme suit :

00	pas d'avertissement
01	la dernière minute a 61 secondes
10	la dernière minute a 59 secondes
11	condition d'alarme (horloge non synchronisée)

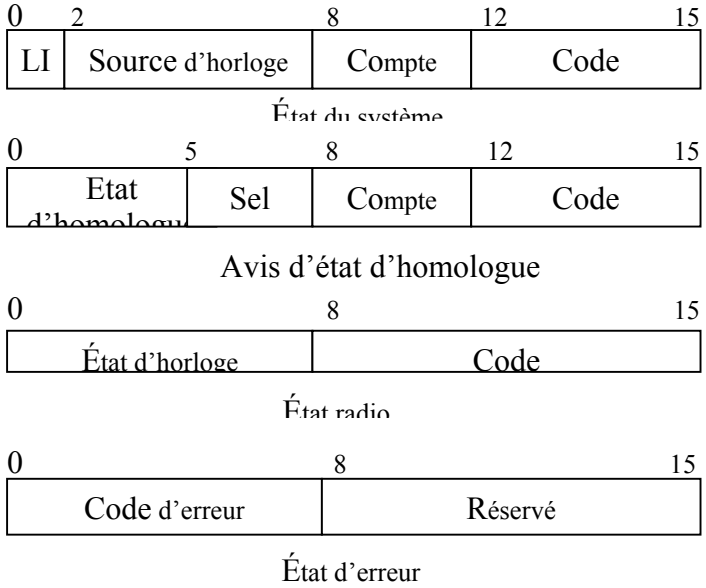

Figure 6. Formats d'avis d'état

Source d'horloge : Entier de six bits qui indique la source de synchronisation en cours, avec ses valeurs codées comme suit :

0	non spécifié ou inconnu
1	horloge atomique calibrée (par exemple, HP 5061)
2	radio VLF (bande 4) ou LF (bande 5) radio (par exemple, OMEGA,, WWVB)
3	radio HF (bande 7) (par exemple, CHU,, MSF,, WWV/H)
4	satellite UHF (bande 9) (par exemple, GOES,, GPS)
5	réseau local (par exemple, DCN,, TSP,, DTS)
6	UDP/NTP
7	UDP/TIME
8	eyeball-and-wristwatch
9	modem téléphonique (par exemple, NIST)
10 à 63	Réservé

Compteur d'événement système : Entier de quatre bits qui indique le nombre d'événements d'exception de système qui sont survenus depuis la dernière fois que l'avis d'état de système a été retourné dans une réponse ou inclus dans un message piège. Le compteur est remis à zéro lorsqu'il est retourné dans le champ état d'une réponse et est gelé lorsqu'il atteint la valeur 15.

Code d'événement de système : Entier de quatre bits qui identifie le dernier événement d'exception de système, avec de nouvelles valeurs qui se substituent aux précédentes, et codées comme suit :

0	non spécifié
1	redémarrage du système
2	faute de système ou de matériel
3	avis de nouvel état du système (bits de saur ou changement de synchronisation)
4	nouvelle source ou strate de synchronisation système (changement de sys.peer ou sys.stratum)
5	réinitialisation d'horloge système (la correction de décalage dépasse CLOCK.MAX)
6	heure ou date de système invalide (voir la spécification NTP)
7	exception d'heure système (voir l'avis d'état d'horloge système)
8 à 15	réservé

B.2.2 Avis d'état d'homologue

Un avis d'état d'homologue est retourné dans le champ état d'une réponse à une commande état de lecture, variables de lecture ou variables d'écriture et apparaît aussi dans la liste des identifiants d'association et des avis d'état retournée par une commande d'état de lecture avec un identifiant d'association zéro. Le format d'un avis d'état d'homologue est le suivant :

Etat d'homologue : Code de cinq bits qui indique l'état de l'homologue déterminé par la procédure de paquet, avec les bits alloués comme suit :

0	configuré (peer.config)
1	authentification activée (peer.authenable)
2	authentification okay (peer.authentic)
3	atteignabilité okay (peer.reach ≠ 0)
4	réservé

Choix d'homologue (Sel) : Entier de trois bits qui indique l'état de l'homologue déterminé par la procédure de choix d'horloge, avec ses valeurs codées comme suit :

0	rejeté
1	vérifications de bonne santé réussies (essais 1 à 8 du paragraphe 3.4.3)
2	vérifications de correction réussies (algorithme d'intersection du paragraphe 4.2.1)
3	vérifications de candidat réussies (si la vérification de limite est mise en œuvre)
4	vérifications des points aberrants réussies (algorithme de mise en grappe du paragraphe 4.2.2)
5	source de synchronisation en cours ; distance maximum dépassée (si la vérification de limite est mise en œuvre)
6	source de synchronisation en cours ; distance maximum okay
7	réservé

Compteur d'événements d'homologue : Entier de quatre bits qui indique le nombre d'événements exceptionnels d'homologue survenus depuis la dernière fois qu'un avis d'état d'homologue a été retourné en réponse à un message piège ou inclus dedans. Le compteur est remis à zéro lorsqu'il est retourné dans le champ état d'une réponse et gelé lorsqu'il atteint la valeur 15.

Code d'événement d'homologue : Entier de quatre bits qui identifie le dernier événement exceptionnel d'homologue, avec les nouveles valeurs qui se substituent aux valeurs précédentes, et codé comme suit :

0	non spécifié
1	erreur IP d'homologue
2	échec d'authentification d'homologue (le bit peer.authentic qui était à un est à zéro)
3	homologue inatteignable (peer.reach était zéro et maintenant différent de zéro)
4	homologue atteignable (peer.reach était à zéro et maintenant différent de zéro)
5	exception d'horloge d'homologue (voir l'avis d'état d'horloge homologue)
6 à 15	réservé

B.2.3 Avis d'état d'horloge

Une horloge de référence peut être rattachée de deux façons à un hôte de service NTP, comme un appareil dédié géré par le système d'exploitation et comme homologue synthétique géré par NTP. Comme dans la commande d'état de lecture, l'identifiant d'association sert à identifier qui est qui, avec zéro pour l'horloge système et une valeur différente de zéro pour une horloge homologue. Une seule horloge système est prise en charge par le protocole, bien que de nombreuses horloges homologues puissent être prises en charge. Un avis d'état d'horloge système ou homologue apparaît dans le champ état de la réponse à une commande de variables de lecture d'horloge ou de variables d'écriture d'horloge. Cet avis peut être considéré comme une extension de l'avis d'état de système ou l'avis d'état d'homologue selon le cas. Le format de l'avis d'état d'horloge est le suivant :

Etat d'horloge : Entier de huit bits qui indique l'état en cours de l'horloge, avec les valeurs codées comme suit :

0	horloge fonctionnant dans les valeurs nominales
1	fin de temporisation de réponse
2	mauvais format de réponse
3	faute de matériel ou de logiciel

4	défaillance de propagation
5	mauvais format ou valeur de date
6	mauvais format ou valeur d'heure
7 à 255	réservé

Code d'événement d'horloge : Entier de huit bits qui identifie le dernier événement exceptionnel d'horloge, les nouvelles valeurs se substituant aux valeurs précédentes. Lorsque survient un changement de toute valeur différente de zéro dans le champ d'état radio, le champ d'état radio est copié sur le champ de code d'événement d'horloge et un événement exceptionnel d'horloge système ou homologue est déclaré selon le cas.

B.2.4 Avis d'erreur d'état

Un avis d'erreur d'état est retourné dans le champ état d'une réponse d'erreur en résultat d'un format ou contenu de message invalide. Sa présence est indiquée lorsque le bit E (erreur) est mis avec le bit réponse (R) dans la réponse. Il consiste en un entier de huit bits codé comme suit :

0	non spécifié
1	échec d'authentification
2	longueur ou format de message invalide
3	opcode invalide
4	identifiant d'association invalide
5	nom de variable inconnu
6	valeur de variable invalide
7	administrativement interdit
8-255	réservé

B.2 Commandes

Les commandes consistent en un en-tête et un champ de données facultatif, montré à la Figure 6. Lorsqu'il est présent, le champ données contient une liste d'identifiants ou d'allocations de la forme

<identifiant>[=<valeur>],<identifiant>[=<valeur>],...

où <identifiant> est le nom ASCII d'une variable de système ou d'homologue spécifié au Tableau 2 ou au Tableau 3 et <valeur> est exprimé comme une constante décimale, hexadécimale ou de chaîne dans la syntaxe du langage de programmation C. Lorsqu'il n'existe aucune ambiguité, les préfixes "sys." ou "peer." indiqués au Tableau 2 ou au Tableau 4 peuvent être supprimés. Des espaces blancs (effecteurs de format ASCII non imprimés) peuvent être ajoutés pour améliorer la lisibilité pour les programmes simples de surveillance qui ne reformattent pas le champ données. Les adresses Internet sont représentées comme quatre octets de la forme [n.n.n.n], où n est en notation décimale et les crochets sont facultatifs. Les horodatages, y compris les références, qui sont à l'origine des valeurs, les reçoivent et les transmettent ainsi que les horloges logiques, sont représentés en unités de secondes et leurs fractions, de préférence en notation hexadécimale, tandis que les valeurs de délai, décalage, dispersion et distance sont représentées en unités de millisecondes et leurs fractions, de préférence en notation décimale. Toutes les autres valeurs sont représentées comme elles sont, de préférence en notation décimale.

Les mises en œuvre peuvent définir des variables autres que celles dont la liste figure au Tableau 2 ou au Tableau 3. Appelées variables extramurales, elles se distingues par l'inclusion de certains types de caractère autres qu'alphanumériques ou "." dans le nom. Pour les commandes qui retournent une liste d'allocations dans le champ données de réponse, si le champ données de commande est vide, on s'attend à ce que toutes les variables disponibles définies dans les Tableaux 3 ou 4 de la spécification NTP soient incluses dans la réponse. Pour les commandes de lecture, si le champ données de la commande n'est pas vide, une mise en œuvre peut choisir de traiter ce champ pour choisir individuellement quelles variables sont à retourner.

Les commandes sont interprétées comme suit :

Etat de lecture (1) : Le champ données de commande est vide ou contient une liste d'identifiants séparés par des virgules. La commande fonctionne dans deux sens selon la valeur de l'identifiant d'association. Si cet identifiant est différent de zéro, la réponse inclut l'identifiant d'homologue et l'avis d'état. Facultativement, le champ données de réponse peut contenir d'autres informations, telles que décrites dans la commande Variables de lecture. Si l'identifiant d'association est zéro, la réponse inclut l'identifiant de système (0) et l'avis d'état, tandis que le champ données contient une liste de paires <<identifiant d'association>> <<avis d'état>> codés en binaire, une pour chaque association actuellement définie.

Variables de lecture (2) : Le champ données de commande est vide ou contient une liste des identifiants séparés par des virgules. Si l'identifiant d'association est différent de zéro, la réponse inclut l'identifiant d'homologue demandé et l'avis d'état, tandis que le champ données contient une liste de variables et valeurs d'homologue comme décrit ci-dessus. Si l'identifiant d'association est zéro, le champ données contient une liste de variables et valeurs de système. Si un homologue a été choisi comme source de synchronisation, la réponse inclut l'identifiant d'homologue et l'avis d'état ; autrement, la réponse inclut l'identifiant de système (0) et l'avis d'état.

Variables d'écriture (3) : Le champ données de commande contient une liste des allocations, comme décrit ci-dessus. Les variables sont mises à jour comme indiqué. La réponse est comme décrit pour la commande Variables de lecture.

Variables de lecture d'horloge (4) : Le champ données de commande est vide ou contient une liste des identifiants séparés par des virgules. L'identifiant d'association choisit les variables d'horloge système ou les variables d'horloge homologue de la même façon que dans la commande Variables de lecture. La réponse inclut l'identifiant d'horloge et l'avis d'état demandés et le champ données contient une liste des variables et valeurs d'horloge, y compris le dernier message de code horaire reçu de l'horloge.

Variables d'écriture d'horloge (5) : Le champ données de commande contient une liste des allocations, comme décrit ci-dessus. Les variables d'horloge sont mises à jour comme indiqué. La réponse est comme décrit pour la commande Variables de lecture d'horloge.

Régler adresse/port de piège (6) : Les champs identifiant d'association de commande, état et données sont ignorés. L'adresse et le numéro de port pour les messages pièges suivants sont tirés de l'adresse et port de source du message de commande lui-même. Le compteur de piège initial pour les messages de réponse pièges est tiré du champ séquence de la commande. Les champs identifiant d'association de réponse, état et données ne sont pas significatifs. Les mises en œuvre devraient inclure des temporisations "sanitaires" pour empêcher les transmissions pièges si le programme de surveillance ne renouvelle pas ces informations après un long intervalle.

Réponse piège (7) : Ce message est envoyé lorsqu'un événement de système, d'homologue ou d'horloge exceptionnel survient. Le champ opcode est 7 et le bit R est mis. Le compteur de piège est incrémenté de un pour chaque piège envoyé et le champ séquence est réglé à cette valeur. Le message piège est envoyé en utilisant les champs adresse et port IP établis par la commande adresse/port piège. Pour un piège système l'identifiant d'association est mis à zéro et le champ état contient l'avis d'état système. Pour un piège homologue, le champ d'identifiant d'association est réglé sur cet homologue et le champ état contient l'avis d'état d'homologue. Des informations codées en ASCII facultatives peuvent être incluses dans le champ données.

Appendice C Questions d'authentification

Les exigences de robustesse de NTP sont similaires à celles des autres protocoles distribués entre homologues multiples utilisés pour l'acheminement réseau, la gestion et l'accès fichier. Cela inclut la protection contre les mises en œuvre fautives, le fonctionnement inapproprié et de possibles attaques malveillantes de répétition avec ou sans modification des données. Ces exigences sont particulièrement contraignantes avec les protocoles distribués dans la mesure où les dommages dus aux défaillances peuvent se propager rapidement tout au long du réseau, dévastant les archives, les acheminements et les systèmes de surveillance et même détruisant des portions majeures du réseau à la façon du classique ver de l'Internet.

Les mécanismes de contrôle d'accès suggérés dans la spécification NTP répondent à ces exigences en limitant l'accès aux homologues de confiance. Les diverses vérification de bonne santé résistent à la plupart des attaques de répétition et d'usurpation en détruisant les vieux duplicata et en utilisant les horodatages originels comme bourrage à usage unique,car il n'est pas vraisemblable que même un homologue synchronisé puisse prédire les futurs horodatages avec la précision requise sur la seule base des observations du passé. De plus, l'environnement du protocole résiste aux attaques de brouillage en employant des serveurs horaires redondants et des acheminements réseau diversifiés. La résistance aux perturbations stochastiques, réelles ou fabriquées, sont minimisées par une conception soignée des algorithmes de filtrage et de sélection.

Cependant, il est possible qu'un intrus déterminé puisse perturber les opérations de conservation de l'heure entre homologues par de subtiles modifications des données de message NTP, telles que la falsification des champs d'en-tête ou certains horodatages. Dans les cas où est nécessaire une protection même contre ces types d'attaques, il faut alors un mécanisme d'authentification de message spécifiquement élaboré fondé sur des techniques cryptographiques. Les mécanismes typiques impliquent l'utilisation de certificats cryptographiques, du support d'algorithmes et de clés, ainsi que de bases de données de support sécurisées et de protocoles de gestion de clés. Les efforts de recherche en cours

dans ce domaine portent sur le développement d'une méthode normalisée qui puisse être utilisée avec de nombreux protocoles, y compris NTP. Cependant, bien qu'il puisse arriver qu'une méthode d'authentification universelle, largement applicable soit adoptée par la communauté de l'Internet et remplace effectivement le mécanisme décrit ici, il n'apparaît pas que des normes et mises en œuvre spécifiques doivent être mises en place pendant la durée de vie de la présente version de NTP.

Le mécanisme d'authentification de NTP décrit ici est destiné à une utilisation provisoire jusqu'à ce que des normes et mises en œuvre spécifiques fonctionnant au niveau réseau ou au niveau transport soient disponibles. La prise en charge de ce mécanisme n'est pas exigée pour la conformité à la spécification NTP elle-même. Le mécanisme, qui fonctionne au niveau application, est conçu pour protéger contre les modifications non autorisées de flux de messages et les déguisements de source en s'assurant que des chemins inviolés et authentifiés existent entre un serveur de confiance, de strate un, dans un sous réseau de synchronisation particulier et tous les autres serveurs de ce sous réseau. Il emploie une somme de contrôle cryptée, calculée par l'expéditeur et vérifiée par le récepteur, conjointement avec un ensemble d'algorithmes pré distribués, de certificats et de clés cryptographiques indexés par un identifiant de clé inclus dans le message. Cependant, il n'y a pas de dispositions dans NTP lui-même pour distribuer ou maintenir les certificats, algorithmes ou clés. Ces quantités peuvent occasionnellement être changées, ce qui peut avoir pour résultat des informations de clés incohérentes alors que le changement de clés est en cours. La nature de NTP lui-même est assez tolérante à de telles perturbations, et aucune disposition particulière n'est incluse pour y faire face.

Le but du mécanisme d'authentification est de fournir un cadre qui puisse être utilisé conjointement avec des combinaisons de mode choisies pour construire des plans spécifiques pour la gestion des communautés de gardiens de l'heure et mettre en œuvre les politiques nécessaires. Il peut être activé de façon sélective ou désactivé homologue par homologue. Il n'y a pas de plan spécifique proposé pour gérer l'utilisation de tels schémas, bien que plusieurs possibilités soient immédiatement évidentes. Dans un scénario, un groupe de serveurs horaires homologues entre eux utilisent des modes symétriques et partagent une clé secrète, disons la clé 1, tandis qu'un autre groupe de serveurs homologues entre eux utilisent des modes symétriques et partagent une autre clé secrète, disons la clé 2. Maintenant, supposons qu'une politique décide que des serveurs choisis dans le groupe 1 peuvent fournir la synchronisation au groupe 2, mais pas l'inverse. Les serveurs choisis dans le groupe 1 reçoivent la clé 2, mais fonctionnent seulement en mode serveur, et donc ne peuvent accepter la synchronisation provenant du groupe 2 ; cependant, le groupe 2 a un accès authentifié aux serveurs du groupe 1. De nombreux autres scénarios sont possibles avec des combinaisons appropriées de modes et de clés.

Les paragraphes qui suivent spécifient un format de paquet et une procédure de somme de contrôle cryptée appropriée pour NTP. Les informations cryptographiques sont portées par un authentifiant qui suit les champs d'en-tête NTP (non modifiés). La procédure de somme de contrôle cryptée utilise la norme de chiffrement de données (DES, *Data Encryption Standard*) [NBS77] ; cependant, seul l'algorithme de chiffrement DES est utilisé et l'algorithme de déchiffrement n'est pas nécessaire. Ce dispositif est spécialement ciblé sur la sensibilité des gouvernements à l'exportation des technologies de cryptage, car l'algorithme de déchiffrement DES n'a pas besoin d'être inclus dans les distributions de logiciels NTP et ne peut donc être extrait et utilisé dans d'autres applications pour éviter de révéler les données de message.

C.1 Mécanisme NTP d'authentification

A sa création et aussi peut-être à d'autres moments, chaque association reçoit des variables qui identifient l'autorité de certificat, l'algorithme de chiffrement, la clé cryptographique et d'autres données possibles. Les procédures spécifiques pour allouer et initier ces variables sont en dehors du domaine d'application de la présente spécification, comme le sont les association des identifiants et des clés et la gestion et la distribution des clés elles-mêmes. Par exemple et pour la cohérence avec les conventions des spécifications NTP, un ensemble de variables appropriées d'homologue et de paquet pourrait comprendre celles qui suivent :

Bit d'authentification activé (peer.authenable) : Bit qui indique que l'association va fonctionner en mode authentifié. Pour les homologues configurés, ce bit est déterminé à partir de l'environnement de démarrage. Pour les homologues non configurés, ce bit est réglé à un si un message arrivant inclut l'authentifiant et il est réglé à zéro autrement.

Bit authentifié (peer.authentic) : Bit indiquant que le dernier message reçu de l'homologue a été correctement authentifié.

Identifiant de clé (peer.hostkeyid, peer.peerkeyid, pkt.keyid) : Entier qui identifie la clé cryptographique utilisée pour générer le code d'authentification de message. La variable de système peer.hostkeyid sert pour les associations actives. La variable peer.peerkeyid est initialisée à zéro (non spécifié) lorsque l'association est mobilisée. Pour les besoins de l'authentification une valeur non allouée est interprétée comme zéro (non spécifié).

Clés cryptographiques (sys.key) : Ensemble de clés DES de 64 bits. Chaque clé est construite comme dans les distributions Unix Berkeley, qui consistent en huit octets, où les sept bits de plus faible poids de chaque octet correspondent aux bits 1 à 7 de DES et le bit de plus fort poids correspond au bit 8 de parité impaire de DES. Par convention, la clé non spécifiée 0 (zéro), consistant en huit octets zéro de parité impaire, sert pour les essais et est supposée connue dans toute la communauté NTP. Les clés restantes sont distribuées par des méthodes qui sortent du domaine d'application de NTP.

Somme de contrôle cryptée (pkt.check) : Somme de contrôle cryptée calculée par la procédure de chiffrement.

Le champ d'authentifiant consiste en deux sous champs, l'un étant la variable pkt.keyid et l'autre étant la variable pkt.check calculée par la procédure de chiffrement, qui est invoquée par la procédure d'émission décrite dans la spécification NTP, et par la procédure de déchiffrement, qui est invoquée par la procédure de réception décrite dans la spécification NTP. Sa présence est révélée par le fait que la longueur totale du datagramme, conformément à l'en-tête UDP, est plus longue que celle du message NTP, qui inclut l'en-tête plus le champ données, s'il est présent. Pour les besoins de l'authentification, le message NTP est bourré de zéros si nécessaire pour une limite de 64 bits, bien que les bits de bourrage ne soient pas considérés comme faisant partie du message NTP lui-même. Le format d'authentifiant montré à la Figure 7 a 96 bits, y compris un identifiant de clé de 32 bits et une somme de contrôle cryptée de 64 bits, et il est aligné sur une frontière de 32 bits pour l'efficacité du calcul. Des informations supplémentaires sont nécessaires dans certaines mises en œuvre, comme l'autorité de certificat et l'algorithme de chiffrement, et elles peuvent être insérées entre le message NTP (bourré) et l'identifiant de clé, pour autant que les conditions de verrouillage soient satisfaites. Comme l'authentifiant lui-même, ces informations ne sont pas incluses dans la somme de contrôle cryptée. L'utilisation de ces données sort du domaine d'application de la présente spécification. Ces conventions pourront être modifiées à l'avenir par suite de l'influence d'autres normes.

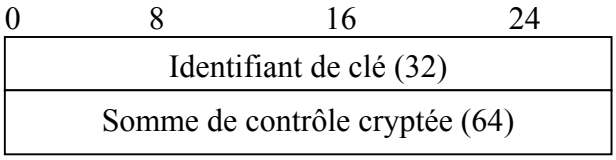

Figure 7. Format d'authentifiant

C.2 Procédures NTP d'authentification

Lorsque l'authentification est mise en œuvre, deux procédures supplémentaires s'ajoutent à celles décrites dans la spécification NTP. Une de celles-ci (encrypt) construit la somme de contrôle cryptée dans les messages émis, tandis que l'autre (decrypt) vérifie cette quantité dans les messages reçus. La procédures utilise une variante de la méthode de chaînage des blocs de chiffrement décrite dans [NBS80] telle qu'elle est appliquée dans DES. En principe, la procédure est indépendante de DES et exige seulement que l'algorithme de chiffrement fonctionne sur des blocs de 64 bits. Alors que le mécanisme d'authentification NTP spécifie l'utilisation de DES, d'autres algorithmes pourraient être utilisés selon une convention préalable.

C.2.1 Procédure de chiffrement

Pour les messages NTP ordinaires, la procédure de chiffrement fonctionne comme suit. Si l'authentification n'est pas activée, la procédure est simplement abandonnée. Si l'association est active (modes 1, 3, 5), la clé est déterminée à partir de l'identifiant de clé système. Si l'association est passive (modes 2, 4) la clé est déterminée à partir de l'identifiant de clé homologue, si le bit authentifié est mis, ou autrement comme la clé par défaut (zéro). Ces conventions permettent une protection supplémentaire contre les attaques de répétition et les erreurs de frappe, ainsi que des facilites pour les essais et pour la migration vers de nouvelles versions. La somme de contrôle cryptée est calculée en utilisant l'en-tête NTP de 64 bits et les avis de données, mais pas les bits de l'authentifiant ou de bourrage.

```
début de procédure de chiffrement
    si (peer.authenable = 0) exit ;                          /* ne rien faire si pas activé */
    si (peer.hostmode = 1 ou peer.hostmode = 3 ou peer.hostmode = 5)
        keyid ← peer.hostkeyid ;                             /* les modes actifs utilisent la clé système */
    autrement
        si (peer.authentic =1)                               /* les modes passifs utilisent la clé d'homologue */
            keyid ← peer.peerkeyid ;
        autrement
            keyid ← 0 ;                                      /* non authentifié utilise la clé 0 */
```

```
    temp ← 0 ;                                          /* calculer la somme de contrôle cryptée */
    pour (chaque en-tête de 64 bits et mot de données) début
        temp ← temp xor mot ;
        temp ← DES (temp, keyid) ;
        endfor;
    pkt.keyid ← keyid ;                                 /* insérer les variables de paquet */
    pkt.check ← temp ;
    fin de procédure de chiffrement;
```

C.2.2 Procédure de déchiffrement

Pour les messages ordinaires la procédure de déchiffrement fonctionne comme suit. Si l'homologue n'est pas configuré, la portion de données du message est inspectée pour déterminer si les champs d'authentifiant sont présents. Si ils le sont, l'authentification est activée ; autrement, elle est désactivée. Si l'authentification est activée et que les champs d'authentifiant sont présents et si la somme de contrôle cryptée correspond, le bit d'authentification est mis à un ; autrement, il est mis à zéro.

```
début de la procédure de déchiffrement
    peer.authentic ← 0 ;
    si (peer.config = 0)                                /* si non configuré, activé selon le paquet */
        si (authentifiant présent)
            peer.authenable ← 1 ;
        autrement
            peer.authenable ← 0 ;
    si (peer.authenable = 0 ou authentifiant non présent)) exit;
    peer.peerkeyid ← pkt.keyid} ;                       /* utilise la clé d'homologue */
    temp ← 0 ;                                          /* calcule la somme de contrôle cryptée */
    pour (chaque en-tête de 64 bits et mot de données) début
        temp ← temp xor mot ;
        temp ← DES (temp, peer.peerkeyid) ;
        endfor;
    si (temp == pkt.check)  peer.authentic ← 1 ;        /* déclare le résultat */
    fin de la procédure de déchiffrement;
```

C.2.3 Procédures de message de contrôle

Dans l'anticipation que les fonctions fournies par les messages de contrôle NTP seront finalement subsumées par une fonction complète de gestion du réseau, les variables d'homologue ne sont pas utilisées pour l'authentification de message de contrôle. Si un message de commande NTP est reçu avec une champ authentifiant, la somme de contrôle cryptée est calculée comme dans la procédure de déchiffrement et le message de réponse inclut le champ authentifiant tel que calculé par la procédure de chiffrement. Si l'authentifiant reçu est correct, la clé pour la réponse est la même que dans la commande ; autrement, la clé par défaut (zéro) est utilisée. Les commandes qui causent un changement de la base de données d'homologue, telles que les commandes de variables d'écriture et d'établissement d'adresse/port piège, doivent être correctement authentifiées ; cependant, les commandes restantes ne sont normalement pas authentifiées afin de minimiser la charge du chiffrement.

Appendice D. Différences avec les versions précédentes

Le protocole NTP d'origine, appelé plus tard NTP Version 0, était décrit dans la RFC-958 [MIL85c]. Ensuite, la Version 0 a été supplantée par la Version 1 (RFC-1059 [MIL88a]), et la Version 2 (RFC-1119 [MIL89]. La description de la Version 2 était séparée en deux documents, la RFC-1119 définissant l'architecture et spécifient le protocole et les algorithmes, et un autre [MIL90b] décrivant le modèle de service, les analyses algorithmiques et réalisant des expériences. Dans les versions précédentes, ces deux objectifs étaient combinés dans un seul document. Bien que l'architecture supposée dans la Version 3 soit identique à celle de la Version 2, les protocoles et algorithmes diffèrent de façon mineure. Les différences entre NTP Version 3 et les versions précédentes sont décrites dans le présent Appendice. Du fait de bogues connues dans les très vieilles mises en œuvre, la poursuite de la prise en charge de mises en œuvre de Version 0 est déconseillée. Il est recommandé que les nouvelles mises en œuvre suivent les lignes directrices ci-dessous lorsqu'elles inter fonctionnent avec de plus anciennes mises en œuvre.

La Version 3 ne change le protocole d'aucune façon significative ni ne rend obsolète des versions précédentes ou des mises en œuvre existantes. La principale motivation de la nouvelle version est de raffiner l'analyse et les modèles de

mise en œuvre pour les nouvelles applications à des vitesse de réseau bien plus élevées que le régime du gigabit par seconde et de fournir la stabilité, l'exactitude et la précision améliorées nécessaires à de telles vitesses. En particulier, les erreurs de source d'heure et de fréquence ont été examinées avec rigueur et des limites d'erreur ont été établies afin d'améliorer les performances, fournir un modèle d'affirmation d'exactitude et indiquer la qualité de la conservation de l'heure à l'utilisateur. La Version 3 incorpore aussi deux nouvelles caractéristiques facultatives, (1) un algorithme pour combiner les décalages d'un certain nombre de serveurs horaires homologues afin d'améliorer l'exactitude et (2) d'améliorer les algorithmes d'horloge locale qui permettent d'accroître de façon substantielle les intervalles de consultation sur tous les chemins de synchronisation afin de réduire la charge du réseau. Ci-après figure un résumé des précédentes versions du protocole ainsi que les détails des changements de la Version 3.

1. La Version 1 ne prend en charge que les modes symétrique actif et symétrique passif, qui sont déterminés en inspectant les champs numéro de port de l'en-tête de paquet UDP. Le mode d'homologue peut être déterminé explicitement d'après la variable de mode de paquet (pkt.mode) si il est différent de zéro, et implicitement d'après les variables de port de source (pkt.peerport) et de port de destination (pkt.hostport) si c'est zéro. Pour le cas où pkt.mode est zéro, le mode est déterminé comme suit :

pkt.peerport	**pkt.hostport**	**Mode**
NTP.PORT	NTP.PORT	symétrique actif
NTP.PORT	pas NTP.PORT	serveur
pas NTP.PORT	NTP.PORT	client
pas NTP.PORT	pas NTP.PORT	pas possible

Noter qu'il n'est pas possible dans ce cas de distinguer entre les modes symétrique actif et symétrique passif. L'utilisation des variables pkt.mode et NTP.PORT de cette façon n'est pas recommandée et pourrait n'être plus prise en charge dans les futures versions du protocole. Le trois bits de plus faible poids du premier octet, spécifiés comme zéro dans la Version 1, sont utilisés pour le champ mode dans la Version 2. Les mises en œuvre de Version-2 et de Version-3 qui inter fonctionnent avec des mises en œuvre de Version-1 devraient fonctionner seulement en mode passif et utiliser la valeur un dans le champ numéro de version (pkt.version) et zéro dans le champ mode (pkt.mode) dans les messages émis.

2. La Version 1 ne prend pas en charge le message de contrôle NTP décrit dans l'Appendice B. Certaines vieilles versions du démon NTP Unix *ntpd* utilisent les bits de plus fort poids du champ strate (pkt.stratum) pour les besoins du contrôle et de la surveillance. Alors que ces bits ne sont jamais établis durant le fonctionnement normal des Version-1, Version-2 ou Version-3, de nouvelles mises en œuvre peuvent utiliser le mode 6 NTP réservé décrit dans l'Appendice B et/ou le mode réservé privé 7 pour des besoins particuliers, tels que le contrôle et la surveillance à distance, et dans de tels cas le format du paquet suivant le premier octet peut être arbitraire. Alors qu'il n'y a pas de garantie que différentes mises en œuvre puissent interopérer en utilisant le mode réservé privé 7, il est recommandé que le format ASCII ordinaire soit utilisé chaque fois que possible.

3. La Version 1 ne prend pas en charge l'authentification. Les identifiants de clé, les clés cryptographiques et les procédures décrites dans l'Appendice C sont nouveaux dans la Version 2 et sont continués dans la Version 3, avec les variables, procédures et champs d'authentifiant correspondants. Dans le message NTP décrit à l'appendice A et le message de contrôle NTP décrit à l'Appendice B le format et le contenu des champs d'en-tête sont indépendants du mécanisme d'authentification et l'authentifiant lui-même suit les champs d'en-tête, aussi les précédentes versions ignoreront l'authentifiant.

4. Dans la Version 1 le champ dispersion totale (pkt.rootdispersion) de l'en-tête NTP était appelé le taux de dérive estimé, mais n'était pas utilisé dans le protocole ou dans les procédures de conservation de l'heure. Les mises en œuvre du protocole de Version-1 règlent normalement ce champ à la valeur courante du registre de compensation de biais, qui est une quantité algébrique. Dans une mise en œuvre de Version 2 des valeurs apparement grandes dans ce champ peuvent affecter l'ordre considéré dans la procédure de choix d'horloge. Les mises en œuvre de Version-2 et Version-3 interopérant avec des mises en œuvre plus anciennes devraient supposer que ce champ est à zéro, sans tenir compte de son contenu réel.

5. La Version 2 et la Version 3 incorporent plusieurs vérification de bonne santé destinées à éviter des interruptions dues à des informations d'horodatage non synchronisées, dupliquées ou boguées. Les vérifications dans la Version 3 sont spécifiquement destinées à détecter les paquets perdus ou dupliqués et résister aux horodatages invalides. Les bits indicateurs de saut sont mis pour montrer aux strates non synchronisées si des mises à jour ne sont pas reçues d'une source de référence depuis une durée considérable ou si la source de référence n'a pas reçu de mises à jour de puis un temps considérable. Certaines mises en œuvre de Version-1 pourraient réclamer indéfiniment une synchronisation valide à la suite d'une perte de source de référence.

6. La procédure de choix d'horloge de la Version 2 a été considérablement précisée par suite de l'expérience accumulée par la mise en œuvre de la Version-1. Des vérifications de bonne santé sont incluses pour l'authentification, les limites de gammes et pour éviter d'utiliser des vieilles données. La liste des candidats est triée deux fois, une fois pour choisir relativement peu de candidats robustes au sein d'une population potentiellement grande d'homologues de tous horizons et ensuite pour ordonner la liste résultante selon les qualités de mesure. Comme dans la Version 1, la procédure de sélection finale élimine de façon répétitive les déviants sur la base de la dispersion pondérée.

7. La procédure d'horloge locale de la Version 2 a été considérablement améliorée par rapport à la Version 1 à la suite des analyses, simulations et expériences. Des vérifications ont été ajoutées pour avertir que l'oscillateur est resté trop longtemps sans mise à jour de la part d'une source de référence. Le registre de conformité a été ajouté pour améliorer la stabilité de fréquence de l'ordre d'une milliseconde par jour. Les divers paramètres ont été réajustés pour une stabilité optimum de boucle en utilisant des données mesurées sur des chemins typiques de l'Internet et avec une horloge locale matérielle typique. Dans la version 3 le modèle de boucle de verrouillage de phase a été affiné pour donner des caractéristiques de bande passante adaptive qui s'ajustent automatiquement pour la stabilité inhérente de l'horloge de référence et de l'horloge locale tout en fournissant la stabilité de boucle optimale dans chaque cas.

8. On a trouvé des problèmes dans les calculs de conservation de l'heure dans la Version 1 avec des LAN à grande vitesse, et ils ont été corrigés dans la Version 2. Ces problèmes étaient causés par la gigue due à de petites différences entre débits d'horloges et des précisions différentes entre les homologues. On a trouvé et corrigé des bogues sournoises dans le contrôle d'atteignabilité et de taux de consultation de la Version-1. Les variables peer.valid et sys.hold ont été ajoutées pour éviter les instabilités lorsque la source de référence change rapidement à cause de grands délais de dispersion dans des conditions d'encombrement sévère du réseau. Les bits peer.config, peer.authenable et peer.authentic ont été ajoutés pour contrôler des caractéristiques spéciales et simplifier la configuration.

9. Dans la Version 3 l'algorithme d'horloge locale a été revu en détail pour améliorer la stabilité et la précision. L'Appendice G présente un modèle mathématique détaillé et un exemple de conception qui ont été affinés à l'aide d'une analyse rétrograde et de simulations extensives utilisant les données collectées sur des chemins Internet ordinaires. La Section 5 de la RFC-1119 sur l'horloge locale NTP a été complètement réécrite pour décrire le nouvel algorithme. Comme le nouvel algorithme peut résulter en des taux de messages très en dessous de l'ancien, il est fortement recommandé qu'il soit utilisé dans les nouvelles mises en œuvre. Noter que cet algorithme ne fait pas partie intégrante de la spécification de protocole NTP elle-même et son utilisation n'affecte pas l'interopérabilité avec les versions précédentes ou les mises en œuvre existantes ; cependant, afin d'assurer la stabilité globale du sous réseau NTP dans l'Internet, il est essentiel que les caractéristiques d'horloge locale de tous les serveurs horaires NTP soient conformes aux modèles analytiques présentés précédemment et dans le présent document.

10. Dans la Version 3, un nouvel algorithme pour combiner les décalages d'un certain nombre de serveurs horaires homologues est présenté dans l'Appendice F. Cet algorithme est modélisé sur ceux qui sont utilisés par les laboratoires de normalisation nationaux pour combiner les décalages pondérés provenant d'un certain nombre d'horloges standard pour construire une échelle de temps synthétique de laboratoire plus exacte que celle de toute horloge prise séparément. Il peur être utilisé dans une mise en oeuvre NTP pour améliorer l'exactitude et la stabilité et réduire les erreurs dues à l'asymétrie de chemins dans l'Internet. Le nouvel algorithme a été simulé en utilisant des données collectées sur des chemins ordinaires de l'Internet et, conjointement avec le nouvel algorithme d'horloge locale, mis en œuvre et testé dans les serveurs horaires Fuzzball qui fonctionnent maintenant sur l'Internet. Noter que cet algorithme ne fait pas partie intégrante de la spécification du protocole NTP lui-même et que son utilisation n'affecte pas l'interopérabilité avec les précédentes versions ou les mises en œuvre existantes.

11. Plusieurs incohérences et erreurs mineures dans des versions précédentes ont été corrigées dans la Version 3. La description des procédures a été réécrite en pseudo-code augmenté par un commentaire pour sa clarté et éviter les ambiguïtés. L'Appendice I a été ajouté pour illustrer les mises en œuvre en langage C des divers algorithmes de filtrage et de sélection suggérés pour NTP. Des informations supplémentaires sont incluses dans la Section 5 et dans l'Appendice E, qui inclut le matériel didactique précédemment inclus dans la Section 2 de la RFC-1119, ainsi que beaucoup de matériel nouveau clarifiant l'interprétation des échelles de temps et les secondes de saut.

12. Des changements mineurs ont été apportés à l'algorithme d'horloge locale de la Version-3 pour éviter les problèmes observés lorsque les secondes de saut sont introduites dans l'échelle de temps UTC et aussi pour prendre en charge un oscillateur auxiliaire de précision, tels qu'une horloge au césium ou un récepteur horaire, comme base de temps de précision. De plus, des changements ont été apportés à certaines procédures décrites à la Section 3 et dans les procédures de filtre d'horloge et de choix d'horloge décrites à la Section 4. Alors que ces changements ont été faits pour corriger des bogues mineures trouvées par l'expérience et qu'ils sont recommandés pour les nouvelles mises en oeuvre, ils n'affectent l'interopérabilité avec les versions précédentes ou les mises en œuvre existantes que de façon mineure (au moins jusqu'au prochain saut de seconde).

13. Dans la Version 3, des changements ont été faits sur la façon dont sont définis, calculés et traités le délai, le décalage et la dispersion, afin de limiter de façon fiable les erreurs inhérentes dans les procédures de transfert de l'heure. En particulier, les accumulations d'erreurs ont été déplacées du calcul du délai au calcul de la dispersion et tous deux ont été inclus dans les procédures de filtre et de choix d'horloge. La procédure de choix d'horloge a été modifiée pour retirer la première des deux étapes de tri/élimination et remplacée par un algorithme d'abord proposé par Marzullo et ensuite incorporé dans le Service de l'heure numérique. Ces changements n'affectent pas significativement le fonctionnement ordinaire ni la compatibilité avec les diverses versions de NTP, mais ils donnent une base aux déclarations formelles d'exactitude telles que décrites à l'Appendice H.

Appendice E L'échelle de temps NTP et sa chronométrie

E.1 Introduction

Ci-après figure un exposé complet de la chronométrie de réseau d'ordinateurs, qui est la détermination précise de l'heure et de la fréquence d'un ordinateur par rapport aux normes internationales et la détermination de la date et de l'heure civile conventionnelle conformément au calendrier moderne. Il décrit les méthodes conventionnelles utilisées pour établir la date et l'heure civiles et les diverses échelles de temps utilisées actuellement. En particulier, il caractérise l'échelle de temps du protocole de l'heure du réseau (NTP) par rapport à l'échelle de temps du temps universel coordonné (UTC), et établit l'interprétation précise des secondes sautées de l'UTC en NTP.

Dans la discussion suivante, les termes heure, oscillateur, horloge, époque, calendrier, date et échelle horaire sont utilisés dans un sens technique. Strictement parlant, l'heure d'un événement est une abstraction qui détermine l'ordre des événements dans une trame de référence donnée. Un oscillateur est un générateur capable de fournir une fréquence précise (par rapport à la trame de référence donnée) à une tolérance spécifiée. Une horloge est un oscillateur jointe à un compteur qui enregistre le nombre (fractionnel) de cycles depuis l'initialisation avec une valeur donnée à une heure donnée. La valeur du compteur à tout moment donné est appelée son époque à cette heure. En général, les époques ne sont pas continues et dépendent de la précision du compteur.

Un calendrier est une transposition de l'époque dans une certaine trame de référence aux dates et heures utilisées dans la vie de tous les jours. Comme plusieurs calendriers sont utilisés de nos jours qui diffèrent parfois sur la date du même événement dans le passé, la chronométrie des événements passés et présents est un art pratiqué par les historiens. Un des objectifs de la présente discussion est de fournir une chronométrie standard pour la datation précise d'événements présents et futurs dans une communauté mondiale de réseautage. Synchroniser les fréquences signifie ajuster les oscillateurs qui sont dans le réseau pour battre la même fréquence ; synchroniser l'heure signifie régler les horloges de telle sorte qu'elles soient en accord à une époque particulière par rapport à l'UTC, comme fourni par les normes internationales, et synchroniser les horloges signifie les synchroniser à la fois en fréquence et en heure.

Afin de synchroniser les horloges, il doit y avoir un moyen de les comparer directement ou indirectement en heure et en fréquence. La trame de référence ultime pour notre monde est constituée par les oscillateurs cosmiques : le soleil, la lune et les autres orbiteurs galactiques. Comme les fréquences de ces oscillateurs sont relativement instables et pas connues avec exactitude, l'oscillateur standard de référence ultime a été choisi par accord international comme synthèse de nombreuses observations d'une transition atomique d'une stabilité parfaite. Les époques de chaque oscillateur céleste et terrestre définissent une échelle de temps distincte, pas toujours nécessairement continue, par rapport à l'oscillateur standard. Un autre but de cette présentation est de décrire une chronométrie standard pour rationaliser le temps d'ordinateur conventionnel et l'UTC ; en particulier, comment traiter les sauts de secondes.

E.2 Fréquence primaire et heures standard

Un standard de fréquence primaire est un oscillateur qui peut maintenir des fréquences extrémement précises par rapport à un phénomène physique, tel qu'une transition dans les états orbitaux d'un électron. Les oscillateurs atomiques actuellement disponibles sont fondés sur les transitions des atomes d'hydrogène, de césium et de rubidium. Le Tableau 7 montre les caractéristiques d'oscillateurs typiques de ces genres comparées à celles de divers types d'oscillateurs à quartz qu'on trouve dans les appareils électroniques. Pour des raisons de coût et de robustesse, les oscillateurs au césium sont utilisés dans le monde entier pour les normes de fréquences primaires nationales. D'un autre côté, les horloges locales utilisées dans les ordinateurs sont presque toujours conçues avec des oscillateurs à cristal non compensés.

Type d'oscillateur	**stabilité** (par jour)	**dérive/vieillissement** (par jour)
Maser à hydrogène	2×10^{-14}	1×10^{-12}/an
Jet de césium	3×10^{-13}	1×10^{-12}/an
Cellule à gaz de rubidium	5×10^{-13}	1×10^{-11}/mois
Cristal contrôlé par four	1×10^{-9} de 0 à 50 °C	1×10^{-10}
Cristal à compensation numérique	5×10^{-8} de 0 à 60 °C	1×10^{-9}
Cristal à température compensée	5×10^{-7} de 0 à 60 °C	1×10^{-9}
Cristal non compensé	~1×10^{-6} par °C	

Tableau 7. Caractéristiques des oscillateurs standard

Pour les trois oscillateurs atomiques figurant sur la liste du Tableau 7, la colonne dérive/vieillissement montre le décalage maximum par jour par rapport à la fréquence standard nominale due aux caractéristiques mécaniques et électriques systématiques. Dans le cas d'oscillateurs à cristal, ce décalage n'est pas constant, d'où il résulte un changement graduel de la fréquence avec le temps, appelé vieillissement. Même si un oscillateur à cristal est à température compensée par quelque moyen, il doit être périodiquement comparé à un standard primaire afin de maintenir la plus haute exactitude. Pour tous les types d'oscillateurs, la colonne stabilité indique la variation maximum en fréquence par jour due au bruit de circuit et aux facteurs d'environnement.

Comme les réseaux de téléphone du monde évoluent rapidement vers les technologies numériques, les méthodes utilisées pour la synchronisation de fréquence dans les réseaux numériques devraient être prises en considération. Un réseau d'horloges dans lesquelles chaque oscillateur est verrouillé en phase avec un seul standard de fréquence est appelé isochrone, tandis qu'un réseau dans lequel certains oscillateurs sont verrouillés en phase avec différents oscillateurs maîtres, mais avec les oscillateurs maîtres étroitement synchronisés en fréquence (pas nécessairement verrouillés en phase) sur un seul standard de fréquence est appelé plésiochrone. Dans les systèmes plésiochrones, la phase de certains oscillateurs peut glisser par rapport à d'autres et causer des erreurs de données occasionnelles dans les systèmes de transmission synchrones.

L'industrie s'est mise d'accord sur une classification des oscillateurs d'horloge en fonction de l'exactitude minimum, de la stabilité minimum et d'autres facteurs [ALL74a]. Il y a trois facteurs qui déterminent la classification : stabilité, gigue et dérapage. La stabilité se réfère à la variation de fréquence systématique dans le temps et est synonyme de vieillissement, dérive, tendance, etc. Gigue (appelée aussi gigue temporelle) se réfère à des variations en fréquence à court terme avec des composants supérieurs à 10 Hz, alors que dérapage se réfère à des variations en fréquence à long terme avec des composants inférieurs à 10 Hz. La classification détermine la strate de l'oscillateur (à ne pas confondre avec la strate NTP), les oscillateurs les plus exacts se voyant allouer la plus basse strate et les oscillateurs les moins exacts la plus haute strate :

Strate	**Exactitude minimum** (par jour)	**Stabilité minimum** (par jour)
1	1×10^{-11}	non spécifiée
2	1.6×10^{-8}	1×10^{-10}
3	4.6×10^{-6}	$3{,}7 \times 10^{-7}$
4	3.2×10^{-5}	non spécifiée

La construction, le fonctionnement et la maintenance des oscillateurs de strate un est supposée être cohérente avec les normes nationales et inclut souvent des oscillateurs au césium ou des oscillateur à cristal de précision synchronisés via LORAN-C aux standard nationaux. Les oscillateurs de strate deux présentent la stabilité requise pour les commutateurs inter centraux tels que le 4ESS d'AT&T et les systèmes d'interconnexion numériques intercentraux, alors que les oscillateurs de couche trois présentent la stabilité requise pour les commutateurs tels que le 5ESS d'AT&T et les systèmes d'interconnexion locaux. Les oscillateurs de strate quatre présentent la stabilité requise pour des équipements d'extrémité de voies numériques et des systèmes PBX.

E.3 Heure et dissémination de fréquence

Pour que l'heure atomique et l'heure civile puissent être coordonnées dans le monde entier, les administrations nationales entretiennent des normes d'heure et de fréquence principale et les coordonnent de façon coopérative en observant diverses diffusions radiophoniques et par l'utilisation occasionnelle d'horloges atomiques portables. La plupart des nations maritimes du monde effectuent une sorte de diffusion d'un service de l'heure pour les besoins du calibrage des chronographes, qui sont utilisés conjointement avec les données d'éphémérides pour déterminer les positions des navigateurs. Dans de nombreux pays le service est primitif et limité à la diffusion de tops à la seconde par les stations de communication marines à certaines heures. Par exemple, une erreur de chronographe d'une seconde représente une erreur de position longitudinale d'environ 0,23 mile nautique à l'équateur.

L'Institut national des normes et technologies des U.S.A (NIST – ex Bureau national des normes) fait fonctionner trois services radio pour la dissémination des informations d'heure et fréquence primaires. Une d'elles utilise des émissions à hautes fréquences (HF ou bande 7 du CCIR) sur les fréquences de 2,5, 5, 10, 15 et 20 MHz à partir de Fort Collins, CO (WWV), et Kauai, HI (WWVH). La propagation du signal se fait habituellement par réflexion sur les couches ionosphériques supérieures,qui varient en hauteur et en composition au fil des jours et des saisons d'où il résulte des délais de variation imprévisibles au récepteur. Le code horaire est émis sur un intervalle de 60 seconde à un débit de données de 1 bit/s en utilisant une sous-porteuse de 100 Hz sur le signal de diffusion. Les informations de code horaire incluent les informations sur l'heure UTC, mais n'incluent pas actuellement les avis sur les années ou les secondes sautées. Alors que ces émissions et celles du Canada depuis Ottawa, Ontario (CHU), et d'autres pays peuvent être reçues sur de larges zones de l'hémisphère occidental, des comparaisons de fréquence fiables ne peuvent être faites que de l'ordre de 10^{-7} et l'exactitude de l'heure est limitée à l'ordre de la milliseconde [BLA74]. Les horloges radio qui fonctionnent avec ces transmissions incluent le Traconex 1020, qui procure des exactitudes de l'ordre de dix millisecondes et vaut dans les 1 500 $ US.

Un second service géré par le NIST utilise des émissions longues fréquences (LF ou bande 5 du CCIR) sur 60 kHz à partir de Boulder, CO (WWVB), et peuvent être reçues sur les USA continental et les zones côtières adjacentes. La propagation du signal se fait via les couches ionosphériques inférieures, qui sont relativement stables et ont des variations diurnes prévisibles en hauteur. Le code horaire est transmis sur un intervalle de 60 secondes à un taux de 1 pps en utilisant des réductions périodiques de la puissance de la porteuse. Avec les techniques de réception et moyenne appropriées et les corrections des effets de propagation diurnes et saisonniers, les comparaisons de fréquence jusqu'à 10^{-11} sont possibles et on peut obtenir des exactitudes horaires approchant les 50 microsecondes [BLA74]. Certains pays d'Europe occidentale font fonctionner des services similaires qui utilisent des transmissions sur 60 kHz à partir de Rugby, U.K. (MSF), et sur 77,5 kHz à partir de Mainflingen, Allemagne (DCF77). Les informations de code horaire incluent l'heure UTC et les avertissements de saut de seconde. Les horloges radio qui fonctionnent avec ces émissions incluent le Spectracom 8170 et Kinemetrics/TrueTime 60-DC et LF-DC, qui fournissent une exactitude à la milliseconde ou moins et valent dans les 2 500 $ US. Cependant, ces récepteurs n'extraient pas l'information de l'année et du saut de seconde.

Le troisième service que fait fonctionner le NIST utilise les émissions en fréquences ultra hautes (UHF ou bande 9 du CCIR) aux environs de 468 MHz à partir des satellites environnementaux à orbite géosynchrone (GOES, *Geosynchronous Orbit Environmental Satellites*), dont trois couvrent l'hémisphère occidental. Le code horaire est entrelacé avec des messages utilisés pour interroger des capteurs distants et consiste en 60 mots de 4 bits codés en décimal binaire émis sur un intervalle de 30 secondes. Les informations de code horaire incluent les informations de l'heure UTC et les avertissement de saut de seconde. Les horloges radio qui fonctionnent avec ces émissions incluent le Kinemetrics/TrueTime 468-DC, qui donne une exactitude de 0,5 ms et vaut dans les 6 000 $ US. Cependant, ce récepteur n'extrait pas les informations de l'année et les avertissements de saut de seconde.

Le Ministère américain de la Défense développe le système de positionnement mondial (GPS, *Global Positioning System*) pour une navigation de précision dans le monde entier. Ce système fournira une couverture mondiale 24 heures sur 24 en utilisant une constellation de 24 satellites dans des orbites de 12 heures. Pour les applications de transfert de l'heure, le GPS a une exactitude potentielle de l'ordre de quelques nanosecondes ; cependant, diverses considérations de politique de défense pourraient limiter l'exactitude à la centaine de nanosecondes [VAN84]. Les informations de code horaire incluent l'heure GPS et la correction UTC ; cependant, il n'y a pas d'avertissement de saut de seconde. Les horloges radio qui fonctionnent avec ces émissions incluent le Kinemetrics/TrueTime GPS-DC, qui procure une exactitude de 200 µs et vaut dans les 12 000 $ US. Cependant, comme seulement la moitié des satellites ont été lancés, de coûteux oscillateurs à quartz ou au rubidium sont nécessaires pour préserver l'exactitude durant les pannes. De plus, comme c'est un récepteur mono canal, il doit être alimenté en coordonnées géographique à un degré de précision à partir d'une source externe avant le début du fonctionnement.

Les garde-côtes des U.S.A, en collaboration avec des agences d'autres pays, ont mis en œuvre depuis de longues années le système de radionavigation LORAN-C [FRA82]. Il fournit actuellement une exactitude de transfert de l'heure de moins d'une microseconde et pourrait obtenir 100 ns dans la zone de couverture d'ondes de sol de quelques centaines de kilomètres de l'émetteur. Au-delà de la zone d'ondes de sol, la propagation du signal se fait via les couches ionosphériques inférieures, ce qui diminue l'exactitude de l'ordre de 50 µs. Avec l'ajout récent de la chaîne médio-continentale, le déploiement des émetteurs LORAN-C donne maintenant une couverture complète des récepteurs horaires du LORAN-C des USA, qui comme le Austron 2000 sont spécialisés et extrémement coûteux (jusqu'à 20 000 $ US). Ils sont principalement utilisés pour surveiller les horloges locales au césium et ne sont pas adaptés pour un fonctionnement automatique sans surveillance. Alors que le système LORAN-C fournit une référence d'heure et de fréquence de haute exactitude dans la zone des ondes de sol, il n'y a pas de modulation du code horaire, de sorte que le récepteur doit être fourni avec l'heure UTC à quelques dixièmes de seconde près à partir d'une source externe avant le début du fonctionnement.

Le système de radionavigation OMEGA [VAS78] que fait fonctionner la marine des USA et d'autres pays consiste en huit émetteurs à très basses fréquences (VLF ou bande 4 du CCIR) qui fonctionnent à des fréquences de 10,2 à 13,1 kHz et fournissent une couverture mondiale 24 heures sur 24. Avec des techniques de réception et de moyenne appropriées et la correction des effets de propagation, les comparaisons d'exactitude de fréquence et d'heure sont dans la gamme des systèmes LF, mais avec une couverture mondiale [BLA74]. Les horloges radio qui fonctionnent avec ces transmissions incluent le Kinemetrics/TrueTime OM-DC, qui procure une exactitude de 1 ms et vaut dans les 3 500 $ US. Alors que le système OMEGA fournit une référence de fréquence d'une haute exactitude, il n'y a pas de modulation du code horaire, de sorte que le récepteur doit être alimenté en coordonnées géographiques au degré près et en temps UTC à cinq secondes près à partir d'une source externe avant le début du fonctionnement. Il y a plusieurs autres services VLF destinés principalement aux communications mondiales de données avec des caractéristiques similaires à celles d'OMEGA. Ces services peuvent être utilisés de façon similaire à OMEGA, mais cela exige des techniques spécialisées qui ne conviennent pas pour un fonctionnement automatique sans surveillance.

Noter que tous les formats de transmission utilisés par les services de diffusion radio du NIST [NBS79] et qu'aucune des horloges radio actuellement disponibles n'incluent de dispositions pour les informations sur l'année et les avertissements de saut de seconde. Ces informations doivent être déterminées à partir d'autres sources. NTP inclut des dispositions pour distribuer des avertissements avant les sauts de secondes en utilisant les bits d'indication de saut de seconde décrits dans la spécification NTP. Le protocole est conçu de telle sorte que ces bits puissent être réglés manuellement ou par un code horaire radio aux serveurs horaires primaires puis distribués automatiquement sur tout le sous réseau de synchronisation à tous les autres serveurs horaires.

E.4 Systèmes calendaires

Les systèmes calendaires utilisés dans l'ancien monde reflètent les besoins agricoles, politiques et rituels caractéristiques des sociétés dans lesquels ils se sont développés. Les observations astronomiques pour établir les solstices d'hiver et d'été étaient utilisées depuis trois à quatre mille ans. Au 14ème siècle AC les chinois Shang avaient établi l'année solaire à 365,25 jours et le mois lunaire à 29,5 jours. Le calendrier lunisolaire, dans lequel le mois rituel est basé sur la lune et l'année agricole sur le soleil a été utilisé dans tous l'ancien Moyen Orient (excepté l'Egypte) et la Grèce depuis le troisième millénaire AC. Les premiers calendriers utilisaient soit treize mois lunaires de 28 jours soit douze mois lunaires alternés de 29 et 30 jours et des moyens hasardeux pour réconcilier l'année lunaire de 354/364 jours avec une vague année solaire de 365 jours.

L'ancien calendrier lunisolaire égyptien avait douze mois lunaires de 30 jours, mais était guidé par l'apparition saisonnière de l'étoile Sirius (Sothis). Afin de réconcilier ce calendrier avec l'année solaire, un calendrier civil fut inventé en ajoutant cinq jours intercalaires pour un total de 365 jours. Cependant, il fut observé avec le temps que l'année civile était plus courte d'un quart de jour que l'année solaire réelle et effectuait donc une précession par rapport à elle sur un cycle de 1 460 ans appelé le cycle sothique. Comme les chinois Shang, les anciens égyptiens avaient donc établi l'année solaire à 365,25 jours, ou à environ 11 minutes de la valeur mesurée présente. En 432 AC, environ cent ans après que les chinois l'aient fait, l'astronome grec Méton calcula qu'il y avait 110 mois lunaires de 29 jours et 125 mois lunaires de 30 jours pour un total de 235 mois lunaires dans 6 940 jours solaires, soit juste 19 ans. Le cycle de 19 ans, appelé cycle métonique, établit le mois lunaire à 29,532 jours solaires, soit à deux minutes près la valeur mesurée présente.

Le calendrier républicain romain était fondé sur une année lunaire et en 50 AC était décalé de huit semaines avec l'année solaire. Jules César invita l'astronome d'Alexandrie Sosigenes à redéfinir le calendrier, ce qui conduisit à l'adoption en 46 AC du calendrier Julien. Ce calendrier est fondé sur une année de 365 jours avec un jour intercalaire inséré tous les quatre ans. Cependant, pour les 36 premières années un jour intercalaire fut inséré par erreur tous les trois ans au lieu de tous les quatre ans. Il en résulta 12 jours intercalaires au lieu de neuf, et une série de corrections qui ne furent pas achevées avant 8 PC.

La semaine sumérienne de sept jours fut introduite seulement au quatrième siècle de notre ère par l'empereur Constantin I. Durant l'ère romaine, un cycle censitaire de 15 ans, appelé le cycle d'indiction, fut institué pour des raisons fiscales. La séquence des noms de jours pour les occurrences consécutives d'un jour particulier de l'année ne se reproduit pas pendant 28 ans, appelé le cycle solaire. Et donc, le plus petit commun multiple du cycle de l'année solaire de 28 ans, du cycle Métonique de 19 ans, et du cycle d'indiction de 15 ans résulte en un grand super cycle de 7 980 ans appelé l'ère julienne, qui commence en 4 713 AC. Une combinaison particulière du jour de la semaine, du jour de l'année, de la phase de la Lune et d'un tour du census va recommencer en 3 268 de notre ère.

En 1545, la discordance entre l'année julienne par rapport à l'année solaire avait accumulé dix jours. En 1582, suivant les suggestions des astronomes Christopher Clavius et Luigi Lilio, le Pape Grégoire XIII publia une bulle papale qui décrétait, entre autres choses, que l'année solaire comporterait 365,2422 jours. Afin de mieux approximer la nouvelle valeur, seules les années centennaires divisibles par 400 seraient des années bissextiles, alors que les autres centennaires ne le seraient pas, ce qui fait la valeur réelle de 365,2425, ou à 26 secondes près la valeur mesurée actuelle. Depuis le

début de l'ère commune et avant 1990, il y avait 474 jours intercalairyes insérés dans le calendrier julien, mais 14 d'entre eux ont été retirés dans le calendier Grégorien. Alors que le calendrier Grégorien est utilisé dans la plupart des pays du monde de nos jours, certains pays ne l'ont pas adopté avant le début du vingtième siècle. Alors que cela reste un sujet d'étude fascinant pour les historiens de l'heure, le récit qui précède montre à l'évidence que la conjugaison des dates calendaires des événements significatifs avec les horodatages de NTP est approchée au mieux. En principe, une datation fiable de tels événements exige seulement un compte exact des jours par rapport à des jours remarquables au niveau mondial, tels que le passage d'une comète ou l'explosion d'une supernova ; cependant, seules les sociétés historiquement persistentes et politiquement stables, telles que les anciens chinois et égyptiens, et aussi les Maya classiques, possédaient les moyens et la volonté de le faire.

E.5 Le système Julien modifié

Pour mesurer l'expansion de l'univers ou la décomposition du proton, il est nécessaire d'avoir un plan de numérotation des jours standard. Par conséquent, l'Union astronomique internationale a adopté l'utilisation de la seconde standard et du calendrier julien(JDN, Julian Day Number) pour dater les événements cosmologiques et les phénomènes qui s'y rapportent. Le jour standard consiste en 86 400 secondes standard, où le temps est exprimé comme une fraction d'un jour entier et l'année standard comporte 365,25 jours standard.

Dans le schéma imaginé en 1583 par le clerc français Joseph Julius Scaliger et nommé d'après son père, Julius Caesar Scaliger, le JDN 0.0 correspond à 12 h (midi) du premier jour de l'ère Julienne, le 1er janvier 4 713 AC. Les années avant l'ère commune (AC) sont dénombrées conformément au calendrier Julien, alors que les années de l'ère commune (PC) sont dénombrées conformément au calendrier Grégorien. Comme le 1er janvier 1 PC dans le calendrier Grégorien correspond au 3 janvier 1 dans le calendrier Julien [DER90], JDN 1 721 426,0 correspond à 12 h du premier jour de l'ère commune, le 1er janvier 1 PC. La Date Julienne Modifiée (MJD), qui est parfois utilisée pour représenter des dates proches de notre ère en temps conventionnel avec moins de chiffres, est définie comme MJD = JD – 2 400 000,5. Suivant la convention selon laquelle notre siècle commence à 0 h le 1er janvier 1900, heure à laquelle l'année tropicale était déjà vieille de 12 h, cet instant choisi correspond à MJD 15 020,0. Et donc, l'échelle de temps Julienne bat avec des siècles en 365,25 jours standard (atomiques) et a été réglé à une valeur donnée à l'époque approximative d'un événement cosmique qui synchronise apparemment la communauté humaine toute entière, l'origine de l'ère commune.

E.6 Détermination de fréquence

Depuis de nombreuses années, l'utilisation la plus importante des informations d'heure et de fréquence était pour la navigation mondiale et la science de l'espace, qui dépendent des observations astronomiques du soleil, de la lune et des étoiles [JOR85]. Le temps sidéral est fondé sur le transit des étoiles à travers le méridien céleste d'un observateur. Le jour sidéral moyen est de 23 heures, 56 minutes et 4,09 secondes, mais varie d'environ ± 30 ms tout au long de l'année du fait de la dérive polaire et des variations d'orbites. L'heure éphémère est fondée sur des tableaux avec lesquels un intervalle de temps standard comme l'année tropicale - une révolution complète de la Terre autour du Soleil - peut être déterminée par des observations du Soleil, de la Lune et des planètes. En 1958, la seconde standard a été définie comme 1/31 556 925,9747 de l'année tropicale qui commençait ce siècle. Sur cette échelle, l'année tropicale est de 365,2421987 jours et le mois lunaire - une révolution complète de la Lune autour de la Terre – est de 29,53059 jours ; cependant, l'année tropicale réelle ne peut être déterminée qu'avec une exactitude d'environ 50 ms et a augmenté d'environ 5,3 ms par an.

Des trois oscillateurs célestes apparents pour les anciens marins et astronomes – la rotation de la Terre autour de son axe, la révolution de la Terre autour du Soleil et la révolution de la Lune autour de la terre – dont aucun n'a une stabilité intrinsèque, par rapport aux technologies modernes, pour servir d'oscillateur de référence standard. En 1967, la seconde standard a été redéfinie comme "9 192 631 770 périodes de la radiation correspondant à la transition entre les deux niveaux hyperfins de l'état de base de l'atome de césium 133." Depuis 1972 le standard d'heure et de fréquence du monde a été fondé sur le temps atomique international (TAI, *International Atomic Time*), qui est défini et entretenu en utilisant plusieurs oscillateurs à jet de césium à une exactitude d'une petite fraction de 10^{13}, soit mieux qu'une microseconde par jour. Noter que, alors que cela fournit une échelle de temps d'une extraordinaire précision, elle n'est pas forcément en accord avec l'heure solaire conventionnelle et peut n'être pas en fait même absolument uniforme, a moins qu'une subtile conspiration atomique puisse être montée.

E.7 Détermination de l'heure et secondes sautées

Le Bureau International des Poids et Mesures (BIPM) utilise des observations astronomiques fournies par l'observatoire naval américain et d'autres observatoires pour déterminer le temps UTC. En partant de l'heure solaire apparente moyenne telle qu'elle est observée, l'échelle de temps UT0 est déterminée en utilisant des corrections pour l'orbite et l'inclinaison de la Terre (l'équation du temps, telle qu'utilisée par les cadrans solaires), l'échelle de temps UT1 (des navigateurs) en ajoutant des corrections pour la migration des pôles et l'échelle de temps UT2 en ajoutant des

corrections pour les variations de périodicité connues. Alors que les fréquences standard sont fondées sur TAI, l'heure civile conventionnelle est fondée sur UT1, qui ralentit présentement par rapport à TAI d'une fraction de seconde par an. Lorsque la magnitude de correction approche de 0,7 seconde, un saut de seconde est inséré ou supprimé dans l'échelle horaire TAI le dernier jour de juin ou de décembre.

Date UTC	MJD	Heure NTP	Décalage
01 Jan 72	41,317	2,272,060,800	0
30 Jun 72	41,498	2,287,785,600	1
31 Dec 72	41,682	2,303,683,200	2
31 Dec 73	42,047	2,335,219,200	3
31 Dec 74	42,412	2,366,755,200	4
31 Dec 75	42,777	2,398,291,200	5
31 Dec 76	43,143	2,429,913,600	6
31 Dec 77	43,508	2,461,449,600	7
31 Dec 78	43,873	2,492,985,600	8
31 Dec 79	44,238	2,524,521,600	9
30 Jun 81	44,785	2,571,782,400	10
30 Jun 82	45,150	2,603,318,400	11
30 Jun 83	45,515	2,634,854,400	12
30 Jun 85	46,246	2,698,012,800	13
31 Dec 87	47,160	2,776,982,400	14
31 Dec 89	47,891	2,840,140,800	15
31 Dec 90	48,256	2,871,676,800	16

Table 8. Tableau des insertions de saut de secondes

Pour la plus précise coordination et l'horodatage des événements depuis 1972, il est nécessaire de savoir quant des sauts de seconde sont mis en œuvre en UTC et comment les secondes sont numérotées. Comme spécifié dans le rapport 517 du CCIR, qui est reproduit dans [BLA74], un saut de seconde est inséré à la suite de la seconde 23:59:59 du dernier jour de juin ou de décembre et devient la seconde 23:59:60 de ce jour. Un saut de seconde serait supprimé en omettant la seconde 23:59:59 d'un de ces jours, bien que cela ne soit jamais arrivé. Les sauts de secondes ont été insérés avant le 1er janvier 1991 pour les occasions énumérées au Tableau 8 (dû à la courtoisie de l'Observatoire Naval américain). Les corrections publiées par le BIPM ne comportent pas que les sauts de secondes, qui résultent en discontinuités par rapport au TAI, mais aussi les ajustements UT1 de 100 ms appelés DUT1, qui fournissent une exactitude accrue pour la navigation et les sciences spatiales.

Noter que la colonne heure NTP montre en réalité les époques qui suivent la dernière seconde du jour donné dans les colonnes date UTC et MJD (sauf pour la première ligne), qui est l'époque précise de l'insertion. La colonne décalage montre les secondes de décalage cumulées entre l'échelle de temps non coordonnée (Julienne) et l'échelle de temps UTC ; c'est à dire, le nombre de secondes à ajouter à l'horloge Julienne afin de maintenir un accord nominal avec l'horloge UTC. Finalement, noter que l'époque de l'insertion est relative à l'échelle de temps immédiatement avant cette époque ; par exemple, l'insertion de l'époque du 31 décembre 1990 est déterminée sur l'échelle de temps en effet après l'insertion du 31 décembre 1990, ce qui signifie que l'insertion réelle par rapport à l'horloge Julienne est quatorze secondes plus tard que l'heure apparente sur l'échelle du temps UTC.

L'échelle de temps UTC bat donc en secondes standard (atomiques) et a été établie à la valeur 0 h MJD 41 317,0 à l'époque déterminée par des observations astronomique comme étant à 0 h le 1er janvier 1972 selon le calendrier Grégorien,c'est-à-dire le tic inaugural de l'ère UTC. En fait, le tic inaugural qui synchronise à jamais les oscillateurs cosmiques, l'horloge Julienne, l'horloge UTC et le calendier grégorien a été déplacé d'environ dix secondes par rapport à l'horloge civile utilisée à ce moment, alors que l'horloge GPS était en avance sur l'horloge UTC de six secondes fin 1990. Par conséquent, l'horloge UTC a reculé par rapport à l'échelle de temps Julienne d'exactement une seconde à des occasions planifiées à des époques symboliques ancrées dans la mémoire institutionnelle de notre civilisation. Noter en passant que les ajustements de saut de seconde affectent le nombre de secondes par jour et par conséquent le nombre de secondes par an. Apparemment, si nous devions choisir de nous en soucier, l'horloge UTC, l'horloge Julienne et les diverses horloges cosmiques dériveront inexorablement avec le temps jusqu'à ce qu'elles soient rationalisées par quelque future bulle papale.

E.8 L'échelle de temps de NTP et l'estimation d'UTC

L'échelle de temps NTP est fondée sur l'échelle de temps UTC, mais pas nécessairement toujours en coincidence avec elle. A 0 h le 1er janvier 1972 (MJD 41 317,0), le premier tic de l'ère UTC, l'horloge NTP a été réglée à 2 272 060 800, représentant le nombre de secondes standard depuis 0 h du 1er janvier 1900 (MJD 15 020,0). L'insertion de sauts de

secondes en UTC et par conséquent dans NTP n'affecte pas l'oscillateur UTC ou NTP, mais seulement la conversion en heure civile UTC conventionnelle. Cependant, comme la seule mémoire institutionnelle disponible pour NTP est le service de diffusion du code horaire de l'UTC, l'échelle de temps NTP est recalée sur UTC à chaque fois que le code horaire est reçu. Et donc, lorsqu'un saut de seconde est inséré dans l'UTC et ensuite dans NTP, la connaissance de tous les précédents sauts de secondes est perdue.

Une autre façon de décrire ceci est de dire qu'il y a de nombreuses échelles de temps NTP comme sauts de secondes historiques. En effet, une nouvelle échelle des temps est établie après chaque saut de seconde. Et donc, les sauts de seconde précédents, pour ne pas mentionner l'origine apparente de l'échelle de temps elle-même, ramènent en arrière d'une seconde chaque fois qu'une nouvelle échelle de temps est établie. Si une horloge synchronisée à NTP en 1990 était utilisée pour établir l'époque UTC d'un événement survenu au début 1972 sans correction, l'événement apparaîtrait quinze secondes en retard par rapport à l'UTC. Cependant, les serveurs horaires primaires de NTP résolvent l'époque en utilisant le code horaire diffusé, de sorte que l'horloge NTP est réglée à la valeur diffusée sur l'échelle de temps courante. Il en résulte que pour la plus précise détermination de l'époque par rapport à l'horloge UTC historique, l'utilisateur doit soustraire de l'époque NTP apparente les décalages indiqués au Tableau 8 aux époques relatives indiquées. C'est une caractéristique de presque tous les mécanismes de distribution de l'heure de nos jours.

La chronométrie impliquée peut être illustrée à l'aide de la Figure 8, qui montre les détails de la numérotation des secondes juste avant, pendant et après la dernière insertion programmée de saut à 23:59:59 du 31 décembre 1989. Remarquer que les bits de saut NTP sont établis le jour avant l'insertion, comme indiqué par les symboles "+" sur la figure. Comme cela fait le jour plus long d'une seconde que d'habitude, le jour NTP ne fera pas la culbute avant la fin de la première occurrence de la seconde 800. Les routines de conversion de l'heure UTC doivent noter l'heure apparente et les bits de saut et traiter en conséquence les conversions d'échelle de temps. Immédiatement après l'insertion du saut les échelles de temps recommencent à battre les secondes comme si le saut ne s'était pas produit. La correspondance chronométrique entre les échelles de temps UTC et NTP continue, mais NTP a tout oublié des insertions de saut passées. Dans NTP, la détermination chronométrique des intervalles de temps UTC s'étendant sur les sauts de secondes sera donc erronnée sauf si les temps d'insertion exacts sont connus.

	UTC			NTP
	heures	secondes	kilosecondes	secondes
31 déc. 90	23:59	:59	2,871,590	,399 +
saut	23:59	:60	2,871,590	,400 +
1er jan. 91	00:00	:00	2,871,590	,400
	00:00	:01	2,871,590	,401

Figure 8. Comparaison des échelles de temps UTC et NTP au saut de seconde

Il est possible que des systèmes individuels utilisent des formats de données internes autres que le format d'horodatage NTP, qui est représenté en secondes à une précision d'environ 200 picosecondes ; cependant, un argument fort existe en faveur d'une représentation en deux parties, une partie pour les jours entiers (MJD ou un décalage fixe qui en est tiré) et l'autre pour les secondes (ou quelque autre valeur étalonnée, comme des millisecondes). Non seulement cela facilite la conversion entre NTP et l'heure civile conventionnelle, mais cela rend plus facile l'insertion des sauts de seconde. Tout ce qui est nécessaire est de changer le modulo du compteur de secondes, qui lorsqu'il est saturé incrémente le compteur de jours. Cette conception assure la continuité de l'échelle de temps, même si la synchronisation externe est perdue avant, durant ou après l'insertion d'un saut de seconde. Comme les données d'horodatage ne sont pas affectées, la synchronisation est assurée, même si les données d'horodatage sont "en vol" à cet instant et ont été générées avant ou à cet instant.

Appendice F. L'algorithme de combinaison d'horloge de NTP

F.1 Introduction

Un problème commun aux sous réseau de synchronisation est l'erreur systématique de décalage de temps qui résulte des chemins de transmission asymétriques, car les réseaux ou supports de transmission dans une direction sont substantiellement différents de ceux de l'autre. Les erreurs peuvent aller de la microseconde sur des réseaux à grande vitesse à d'importantes fractions d'une seconde sur des chemins alliant satellite et voie terrestre. Il a été prouvé expérimentallement que ces erreurs peuvent être considérablement réduites en combinant les décalages apparents d'un certain nombre de serveurs horaires pour produire un décalage courant plus exact. Ci-après figure une description de la méthode combinatoire utilisée dans la mise en œuvre de NTP pour le Fuzzball [MIL88b]. La méthode est similaire à celle utilisée par les laboratoire nationaux de normalisation pour déterminer une échelle de temps synthétique de

laboratoire à partir d'un ensemble d'horloges au césium [ALL74b]. Ces procédures sont facultatives et ne sont pas exigées d'une mise en œuvre conforme à NTP.

Dans la description suivante, la stabilité d'une horloge est la façon dont elle peut conserver une fréquence constante, l'exactitude est la façon dont sa fréquence et son heure peuvent se comparer avec la norme nationale, et la précision est avec quelle précision ces quantités peuvent être maintenues au sein d'un système de conservation de l'heure particulier. Sauf indication contraire, le décalage de deux horloges est la différence d'heure entre elles, alors que le biais est la différence de fréquence (dérivée première du décalage par rapport au temps) entre elles. Les horloges réelle affichent une certaine variation du biais (dérivée seconde du décalage par rapport au temps), qui est appelée dérive.

F.2 Détermination de l'heure et de la fréquence

La Figure 9 montre l'organisation globale du modèle de serveur horaire NTP. Les horodatages échangés avec d'autres sous réseaux homologues éventuellement nombreux sont utilisés pour déterminer les délais individuels d'aller-retour et les décalages d'horloge par rapport à chaque homologue comme décrit dans la spécification NTP. Comme indiqué sur la figure, les délais et les décalages calculés sont traités par le filtre d'horloge pour réduire le bruit horaire incident et le sous réseau le plus exact et le plus fiable déterminé par l'algorithme de choix d'horloge. Les décalages résultants de ce sous réseau sont d'abord combinés comme décrit ci-après puis traités par la boucle de verrouillage de phase (PLL, *phase-locked loop*). Dans la PLL, les effets combinés des opérations de filtrage, de sélection et de combinaison sont la production d'un terme de correction de phase. Celui-ci est traité par le filtre de boucle pour contrôler l'horloge locale, qui fonctionne comme un oscillateur à tension controllée (VCO, *voltage-controlled oscillator*). Le VCO fournit la référence de temps (phase) pour produire les horodatages utilisés dans tous les calculs.

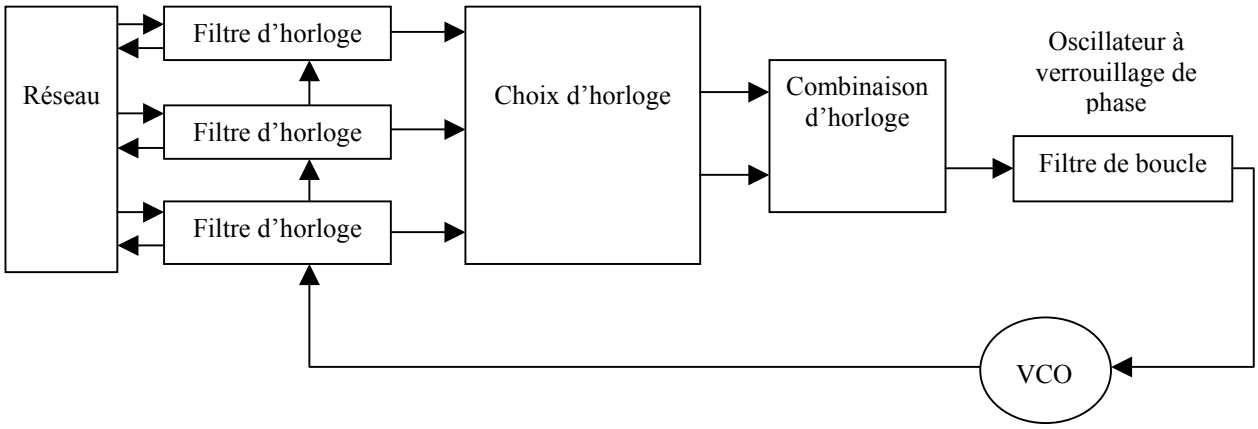

Figure 9. Protocole de l'heure duréseau

F.3 Modélisation d'horloge

La définition normalisée internationale de l'intervalle de temps est en termes de seconde standard : "la durée de 9 192 631 770 périodes de la radiation correspondant à la transition entre les deux niveaux hyperfins de l'état de base de l'atome de césium 133." Représentons par u l'unité standard d'intervalle de temps ainsi définie et par $v = \frac{1}{u}$ l'unité standard de fréquence. L'époque, notée t, est définie comme la lecture d'un compteur qui tourne à la fréquence v et commence à compter à une époque initiale convenue t_0, qui définit l'échelle de temps standard ou absolue. Pour les besoins de l'analyse suivante, l'époque de l'échelle de temps standard, ainsi que l'heure indiquée par une horloge sera considéré comme continu. En pratique, l'heure est déterminée par rapport à une horloge construite à partir d'un oscillateur atomique et d'un système de compteurs/diviseurs, qui définit une échelle de temps associée à cet oscillateur particulier. L'heure et la fréquence standard sont alors déterminées à partir d'un ensemble de tels algorithmes et échelles de temps conçus pour les combiner afin de produire une échelle de temps composite approximant l'échelle de temps standard.

Soit $T(t)$ l'heure affichée par une horloge à l'époque t par rapport à l'échelle de temps standard :

$$T(t) = \tfrac{1}{2}D(t_0)[t - t_0]^2 + R(t_0)[t - t_0] + T(t_0) + x(t) ,$$

où $D(t_0)$ est la dérive fractionnelle de fréquence par unité de temps, $R(t_0)$ est la fréquence et $T(t_0)$ est l'heure à une époque précédente t_0. Dans le modèle stationnaire usuel, ces quantités peuvent être supposées constantes ou changeant

lentement avec l'époque. La nature aléatoire de l'horloge est caractérisée par $x(t)$, qui représente le bruit aléatoire (gigue) par rapport à l'échelle de temps standard. Dans l'analyse usuelle, le terme de second ordre $D(t_0)$ est ignoré et le terme de bruit $x(t)$ modélisé comme une distribution normale avec une densité spectrale prévisible ou une fonction d'autocorrélation.

La fonction de densité de probabilité du décalage de temps p(t - $T(t)$) apparaît habituellement comme une courbe en cloche centrée quelque part près de zéro. La largeur et la forme générale de la courbe sont déterminées par $x(t)$, qui dépend de la précision de l'oscillateur et des caractéristiques de gigue, ainsi que du système de mesures et de ses voies de transmission. En commençant à l'époque t_0 le décalage est mis à zéro, à la suite de quoi la cloche glisse vers la gauche ou la droite, selon la valeur de $R(t_0)$ et accélère selon la valeur de $D(t_0)$.

F.4 Développement d'une échelle de temps composite

Considérons maintenant les décalages de temps d'un certain nombre d'horloges réelles connectées par de vrais réseaux. Un affichage des décalages de toutes les horloges par rapport à l'échelle de temps standard va apparaître comme un système de courbes en cloche avec une lente précession des unes par rapport aux autres, mais avec certaines plus loin du zéro nominal que les autres. Les cloches vont normalement être éparpillées sur l'espace de décalage, plus ou moins proches les unes des autres, certaines se recouvrant et pas d'autres. Le problème est d'estimer le véritable décalage par rapport à l'échelle de temps standard à partir d'un système de décalages collectés de façon mécanique parmi les horloges.

Une échelle de temps composite peut être déterminée à partir d'une séquence de décalages mesurés entre les n horloges d'un ensemble à des intervalles nominaux τ. Soit $R_i(t_0)$ la fréquence et $T_i(t_0)$ l'heure de la $i^{ème}$ horloge à l'époque t_0 par rapport à l'échelle de temps standard et soit "^" qui désige les estimations associées. Une estimation de T_i calculée à t_0 pour l'époque $t_0 + \tau$ est :

$$\hat{T}_i(t_0 + \tau) = \hat{R}_i(t_0)\,\tau + T_i(t_0)\,,$$

en négligeant les termes de second ordre. Considérons un ensemble de n mesures indépendantes du décalage de temps faites entre les horloges à l'époque $t_0 + \tau$ et soit $T_{ij}(t_0 + \tau)$ le décalage entre l'horloge i et l'horloge j à cette époque, défini comme

$$T_{ij}(t_0 + \tau) \equiv T_i(t_0 + \tau) - T_j(t_0 + \tau)\,.$$

Noter que $T_{ij} = - T_{ji}$ et que $Tii = 0$. Soit $wi(\tau)$ un facteur de pondération précédemment déterminé associé à la $i^{ème}$ horloge pour l'intervalle nominal τ. La base de la nouvelle estimation à l'époque t0 + τ est

$$T_j(t_0 + \tau) = \sum_{i=1}^{n} w_i(\tau)\left[\hat{T}_i(t_0 + \tau) + T_{ij}(t_0 + \tau\right]$$

C'est-à-dire que l'heure apparente indiquée par la $j^{ème}$ horloge est une moyenne pondérée de l'heure estimée de chaque horloge à l'époque $t_0 + \tau$ plus le décalage de temps mesuré entre la $j^{ème}$ horloge et cette horloge à l'époque $t_0 + \tau$.

On peut avoir une compréhension intuitive du comportement de cet algorithme avec l'aide de quelques exemples. Si $wi(\tau)$ est l'unité pour la $i^{ème}$ horloge et zéro pour toutes les autres, l'heure apparente pour chacune des autres horloges est simplement l'heure estimée $\hat{T}_i(t_0 + \tau)$. Si $w_i(\tau)$ est zéro pour la $i^{ème}$ horloge, cette horloge ne peut jamais affecter aucune autre horloge et son heure apparente est déterminée entièrement à partir des autres horloges. Si $w_i(\tau) = 1/n$ pour tout i, l'heure apparente de la $i^{ème}$ horloge est égale à la moyenne des estimations d'heure calculées à t_0 plus la moyenne des décalages de temps mesurés sur toutes les autres horloges. Finalement, dans un système avec deux horloges et $w_i(\tau) = ½$ pour chacune, et si l'heure estimée à l'époque $t_0 + \tau$ est en avance de 1 s pour une horloge et en retard de 1 s pour l'autre, l'heure apparente pour les deux horloges va coïncider avec l'échelle de temps standard.

Pour établir une base pour le prochain intervalle τ, il est nécessaire de mettre à jour la fréquence estimée $\hat{R}_i(t_0 + \tau)$ et le facteur de pondération $w_i(\tau)$. La fréquence moyenne supposée pour la $i^{ème}$ horloge durant l'intervalle précédent τ est simplement la différence entre les heures au commencement et à la fin de l'intervalle divisée par τ. Une bonne estimation pour $R_i(t_0 + \tau)$ a été trouvée comme étant la moyenne exponentielle de ces différences, qui est donnée par

$$\hat{R}_i(t_0+\tau) = \hat{R}_i(t_0) + \alpha_i\left[\hat{R}_i(t_0) - \frac{T_i(t_0+\tau) - T_i(t_0)}{\tau}\right]$$

où α_i est un facteur de pondération déterminé expérimentalement qui dépend de l'erreur de fréquence estimée sur la $i^{ème}$ horloge. Pour calculer le facteur de pondération $w_i(\tau)>$, il est nécessaire de déterminer l'erreur attendue $\varepsilon_i(\tau)$ pour chaque horloge. Dans ce qui suit, "|" indique la valeur absolue et "<>" indique la moyenne de temps infinie. En pratique, les moyennes infinies sont calculées comme moyennes de temps exponentielles. Une estimation de la magnitude de l'erreur non biaisée de la ième horloge accumulée sur l'intervalle nominal τ est

$$\varepsilon_i(\tau) = |\hat{T}_i(t_0+\tau) - T_i(t_0+\tau)| + \frac{0{,}8 < \varepsilon_e^2(\tau) >}{\sqrt{< \varepsilon_i^2(\tau) >}},$$

où $\varepsilon_i(\tau)$ et $\varepsilon_e(\tau)$ sont respectivement les erreurs accumulées de la $i^{ème}$ horloge et de l'ensemble entier des horloges. L'erreur accumulée de l'ensemble entier est

$$< \varepsilon_e^2(\tau) >= \left[\sum_{i=1}^{n} \frac{1}{< \varepsilon_i^2(\tau)}\right]^{-1}$$

Finalement, le facteur de pondération pour la $i^{ème}$ horloge est calculé comme

$$wi(\tau) = \frac{< \varepsilon_e^2(\tau) >}{< \varepsilon_i^2(\tau)}.$$

Lorsque toutes les estimations et tous les facteurs de pondération ont été mis à jour, l'origine de l'intervalle d'estimation est déplacée et la nouvelle valeur de t_0 devient la vieille valeur de $t_0 + \tau$.

Tout en entrant pas dans le détail des calculs, il est utile d'estimer l'erreur de fréquence, car les horloges de l'ensemble peuvent être localisées à une certaine distance l'une de l'autre et devenir isolées pendant un certain temps du fait d'une défaillance de réseau. L'erreur de décalage de fréquence dans R_i est équivalente à la fréquence fractionnaire y_i,

$$y_i = \frac{v_i - v_I}{v_I}$$

mesuré entre la $i^{ème}$ échelle de temps et l'échelle de temps standard I. En laissant de côté temporairement l'indice i pour la lisibilité, considérons une séquence de N échantillons de décalage de fréquence indépendants $y(j)$ (j = 1, 2, ... , N) où l'intervalle entre les échantillons est uniforme et égal à T. Soit τ l'intervalle nominal sur lequel ces échantillons sont moyennés. La variance d'Allan $\sigma_y^2(N, T, \tau)>$ [ALL74a] est définie par

$$,< \sigma_y^2(N,T,\tau) >=< \frac{1}{N-1}\left[\sum_{j=1}^{N} y(j)^2 - \frac{1}{N}\left(\sum_{j=1}^{N} y(j)\right)^2\right] >$$

Une formulation particulièrement utile est N = 2 et T = τ :

$$< \sigma_y^2(N=2,T=\tau,\tau) >\equiv \sigma_y^2(\tau) =< \frac{[y(j+1) - y(j)]^2}{2} >,$$

de sorte que

$$\sigma_y^2(\tau) = \frac{1}{2(N-1)}\sum_{j=1}^{n-1}[y(j+1) - y(j)]^2.$$

Alors que la variance d'Allan trouve une application lors de l'estimation des erreurs dans les ensembles d'horloges au césium, son application à NTP est limitée du fait de la charge de calcul et de mémorisation des données. Comme décrit

à la section suivante, il est possible d'estimer les erreurs avec un certain degré de confiance en utilisant les sous-produits normaux des algorithmes de traitement de NTP.

F.5 Application à NTP

Le modèle d'horloge de NTP est un peu moins complexe que le modèle général décrit ci-dessus. Par exemple, au niveau de développement actuel, il n'est pas nécessaire d'estimer séparément l'heure et la fréquence de toutes les horloges homologues, mais seulement la fréquence de l'horloge locale. Si la référence de conservation du temps est l'horloge locale elle-même, les décalages disponibles dans les variables d'homologue peer.offset peuvent être utilisées directement à partir des quantités T_{ij} ci-dessus. De plus, le modèle d'hologe locale NTP incorpore une boucle à verrouillage de phase de type II, qui estime elle-même de façon fiable les erreurs de fréquence et les corrige en conséquence. Et donc, l'exigence d'estimation de la fréquence est entièrement éliminée.

Il reste le problème de la détermination d'une estimation d'erreur robuste et aisément calculable ε_i. La méthode décrite ci-dessus, bien qu'analytiquement justifiée, est très difficile à mettre en œuvre. Heureusement, comme sous produit de l'algorithme de filtre d'horloge de NTP, une utile estimation d'erreur est disponible dans la forme de la dispersion. Comme décrit dans la spécification NTP, la dispersion inclut la valeur absolue de la moyenne pondérée des décalages entre l'échantillon de décalage choisi et les n - 1 autres échantillons retenus pour la sélection. L'efficacité de cette estimation a été comparée à l'estimation précédente par simulation en utilisant les données de conservation de l'heure observées et on a trouvé qu'elle donne des résultats tout à fait acceptables.

L'algorithme de combinaison d'horloge de NTP peut être mis en œuvre avec seulement des modifications mineures aux algorithmes décrits dans la spécification NTP. Bien qu'ailleurs dans la spécification NTP l'utilisation de routines de multiplication/division génériques ait été évitée avec succès, il ne semble pas qu'il y ait de moyen de les éviter dans l'algorithme de combinaison d'horloge. Cependant, pour de meilleures performances, l'algorithme d'horloge locale décrit ailleurs dans le présent document devrait lui aussi être mis en œuvre, car les algorithmes de combinaison produisent un modeste accroissement du bruit de phase que l'algorithme d'horloge locale révisé est sensé supprimer.

F.6 Procédure de combinaison d'horloges

Le résultat de la procédure NTP de choix d'horloge est un ensemble de survivants (il doit y en avoir au moins un) qui représente des vraies chimères, ou horloges correctes. Lorsque la combinaison d'horloge n'est pas mise en œuvre, un de ces homologues, choisi comme candidat le plus vraisemblable, devient la source de synchronisation et son décalage calculé devient la correction d'horloge finale. A la suite de cela, les variables système sont ajustées comme décrit dans les procédures NTP de mise à jour d'horloge. Lorsque la combinaison d'horloge est mise en œuvre, ces actions restent inchangées, excepté que la correction d'horloge finale est calculée par la procédure de combinaison d'horloges.

La procédure de combinaison d'horloges est invoquée à partir de la procédure de choix d'horloge. Elle élabore à partir des variables de tous les homologues survivants la correction d'horloge finale Θ. L'erreur estimée exigée par les algorithmes précédemment décrits se fonde sur la distance de synchronisation Λ calculée par la procédure de distance, telle que définie dans la spécification NTP. La réciproque de Λ est le poids de chaque contribution de décalage d'horloge à la correction d'horloge finale. Le pseudo-code suivant décrit la procédure.

début de la procédure de combinaison d'horloge
    *temp*1 ← 0 ;
    *temp*2 ← 0 ;
    pour (chaque *homologue* restant sur la liste de candidats) /* balayage pour rechercher tous les survivants */
        Λ ← distance (*homologue*) ;
        $temp \leftarrow \frac{1}{peer.stratum \times NTP.MAXDISPERSE + \Lambda}$;
        *temp*1 ← *temp*1 + *temp* ; /* mettre à jour poids et décalage */
        *temp*2 ← *temp*2~+~*temp* x peer.offset ;
        endif;
    $\Theta \leftarrow \frac{temp2}{temp1}$; /* calculer la correction finale */
fin de la procédure de combinaison d'horloge ;

La valeur de Θ est la correction d'horloge finale utilisée par la procédure d'horloge locale pour régler l'horloge.

Appendice G. Modélisation et analyse d'horloges d'ordinateur

Une horloge d'ordinateur comporte une sorte d'oscillateur de référence, qui est stabilisé par un cristal de quartz ou par d'autres moyens, tels qu'une alimentation en énergie. Habituellement, l'horloge comporte un diviseur de fréquence qui ramène la fréquence de l'oscillateur à une valeur standard, telle que 1 MHz ou 100 Hz, et un compteur, mis en œuvre dans le matériel, le logiciel ou toute combinaison des deux, qui puisse être lue par le processeur. Pour les systèmes destinés à être synchronisés à une source externe d'heure standard, il doit y avoir des moyens de corriger la phase et la fréquence par des réglages micrométriques occasionnels produits par le protocole de conservation de l'heure. Une attention particulière doit être apportée dans tous les systèmes de conservation de l'heure à ce que les indications d'horloge soient toujours à croissance monotone, c'est-à-dire que le système horaire ne doit jamais aller à reculons.

G.1 Modèles d'horloges d'ordinateur

La plus simple horloge d'ordinateur consiste en un verrou *(latch)* matériel qui est établi par débordement d'un compteur matériel ou d'un diviseur de fréquences, et cause l'interruption d'un processeur ou tic. Le verrou est remis à son état d'origine lorsque le processeur prend connaissance du tic, ce qui incrémente la valeur d'un compteur d'horloge logique. La phase de l'horloge est réglée en ajoutant des corrections périodiques autant que nécessaires au compteur. La fréquence de l'horloge peut être réglée en changeant la valeur de l'incrément lui-même, afin de faire accélérer ou ralentir l'horloge. La précision de ce modèle d'horloge simple est limitée à l'intervalle de tic, normallement de l'ordre de 10 ms ; bien que dans certains systèmes l'intervalle de tic puisse être changé en utilisant un noyau variable.

Ce modèle d'horloge logique exige une interruption de traitement à chaque tic, ce qui cause une charge significative si l'intervalle de tic est faible, disons de l'ordre de moins de 1 ms avec les plus récents processeurs RISC. Et donc, pour réaliser des précisions de conservation de l'heure de moins de 1 ms, il est nécessaire d'avoir du matériel d'assistance. Une conception directe consiste à apporter un oscillateur à tension contrôlée (VCO), dans lequel la fréquence est contrôlée par un convertisseur analogique/numérique à mémoire tampon (DAC). Dans l'hypothèse où la tolérance du VCO est de 10^{-4} ou 100 parties par million (ppm) (valeur raisonnable pour des cristaux qui ne coûtent rien) et la précision requise est de 100 μs (objectif raisonnable pour un processeur RISC), le DAC doit au moins comporter dix bits.

La Figure 10a montre un schéma de conception d'horloge d'ordinateur entièrement construite avec des composants logiques matériels. L'horloge est lue en impulsant d'abord le signal de lecture, qui verrouille la valeur en cours du compteur d'horloge, puis ajoute le contenu du verrou de compteur d'horloge et une variable de décalage d'horloge de 64 bits, qui est conservée dans un processeur à mémoire. La phase d'horloge est ajustée en ajoutant une correction à la variable de décalage d'horloge, tandis que la fréquence d'horloge est réglée en chargeant une correction au verrou DAC. En principe, ce modèle d'horloge peut être adapté à toute précision en changeant le nombre de bits du diviseur de fréquence ou le compteur d'horloge ou en changeant la fréquence VCO. Cependant, il ne semble pas utile de réduire la précision en dessous du délai de latence minimum d'interruption, qui est de quelques microsecondes pour un processeur RISC moderne.

S'il n'est pas possible de faire varier la fréquence de l'oscillateur, ce qui peut être le cas si l'oscillateur est à une norme de fréquence externe, une conception du genre de celle indiquée à la Figure 10b peut être utilisée. Elle inclut un oscillateur à fréquence fixe et un diviseur de fréquence qui inclut un compteur d'avalage à module dual qui peut fonctionner en mode division par 10 ou division par 11 et il est contrôlé par une impulsion produite par un diviseur programmable (PD). Le PD est chargé avec une valeur représentant le décalage de fréquence. Chaque fois que le diviseur déborde, une impulsion est produite qui commute le compteur d'avalement du mode division par 10 au mode division par 11 et l'inverse, ce qui en effet "avale" ou suprime une seule impulsion du train d'impulsions du diviseur de fréquence.

Le train d'impulsion produit par le diviseur de fréquence est contrôlé précisément sur une petite gamme par le contenu du PD. S'il est programmé pour émettre des impulsions à faible débit, relativement peu d'impulsions sont avalées chaque seconde et la fréquence comptée est proche de la limite supérieure de sa gamme, alors que s'il est programmé pour émettre des impulsions à un débit élevé, un nombre relativement élevé d'impulsions est avalé et la fréquence comptée est proche de la limite inférieure. En supposant un certain degré de liberté dans le choix de la fréquence de l'oscillateur et des ratios du diviseur de fréquence, cette conception peut compenser une large gamme de tolérances de fréquences d'oscillateur.

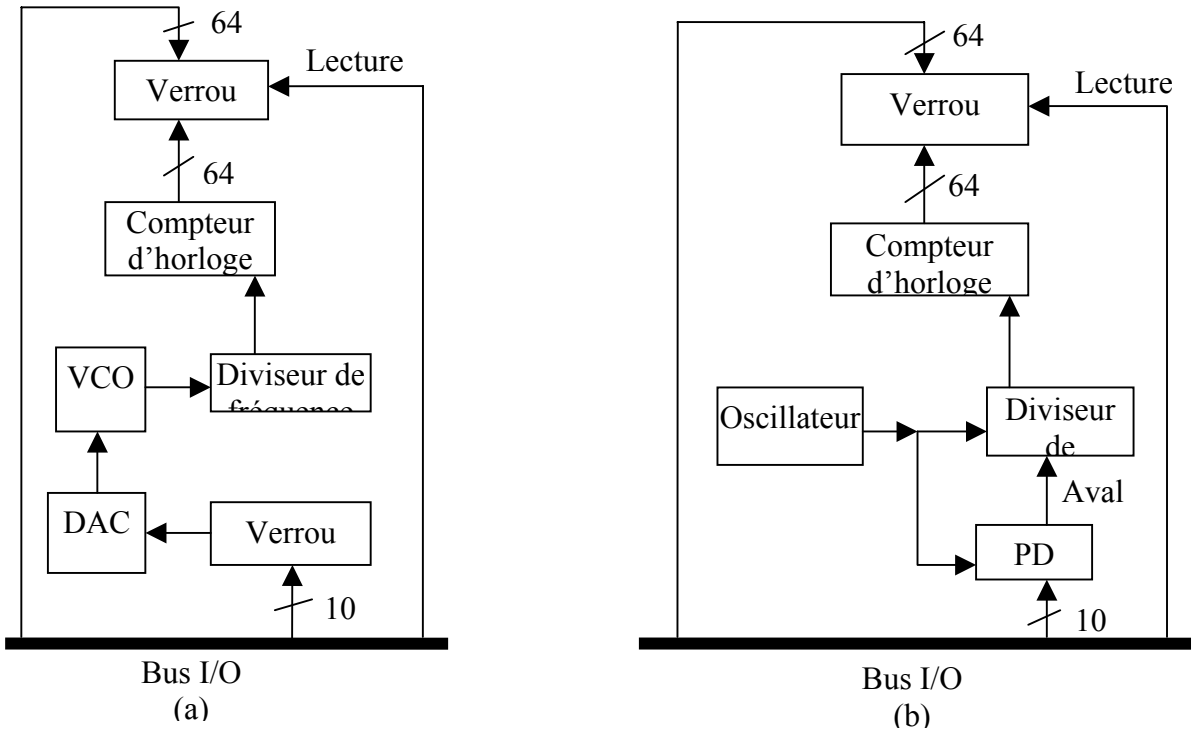

Figure 10. Modèles d'horloge matérielle

Dans tous les concepts présentés ci-dessus, il est nécessaire de limiter la quantité d'ajustements incorporés à toute étape pour garantir que les indications d'horloge du système sont toujours croissantes de façon monotone. Avec le modèle d'horloge logique, ceci est garanti tant que l'incrément n'est pas négatif. Lorsque la magnitude d'un ajustement de phase excède l'intervalle de tic (tel que corrigé par l'ajustement de fréquence), il est nécessaire d'étaler les ajustements sur plusieurs intervalles de tic. Cette stratégie va jusqu'à un décalage de fréquence délibéré soutenu pendant un intervalle égal au nombre total de tics exigé et, en fait, c'est une caractéristique du modèle d'horloge Unix présenté ci-dessous.

Dans les modèles d'horloge matérielle, les mêmes considérations s'appliquent. Cependant, dans ces conceptions, l'intervalle de tic se monte à une seule impulsion à la sortie du diviseur de fréquence, ce qui peut être de l'ordre de 1 ms. Pour éviter de diminuer l'heure indiquée lorsque survient une correction de phase négative, il est nécessaire d'éviter de modifier la variable de décalage d'horloge dans la mémoire du processeur et de confiner tous les ajustements dans le VCO ou le diviseur de fréquence. Et donc, tous les ajustements de phase doivent être effectués au moyen de réglages de fréquence programmés de la même façon qu'avec le modèle d'horloge logicielle décrit plus haut.

Il est intéressant de se demander si la conception de l'aide d'un processeur pourrait fournir toutes les fonctions ci-dessus dans une interface matérielle compacte, servant à tout. L'interface peut consister en une puce de temporisation multifonction telle que le AMD 9513A, qui inclut cinq compteurs de 16 bits, possédant chacun une charge programmable et des registres de conservation, plus un oscillateur à crital incorporé, un diviseur de fréquences et des circuits de contrôle. Un compteur d'horloge matérielle à 48 bits utiliserait trois des compteurs à 16 bits, tandis que le quatrième serait utilisé comme compteur d'avalage et le cinquième de diviseur programmable. Avec l'ajout d'un appareil logique à dispositif programmable et une interface d'hôte à architecture spécifique, ce concept compact pourrait fournir toutes les fonctions nécessaires pour un système de conservation de l'heure complet.

G.1.1 Le modèle d'horloge Fuzzball

Le modèle d'horloge Fuzzball utilise une combinaison de matériel et de logiciel pour la fourniture d'une heure de précision avec un minimum de redondance de logiciel et de processeurs. Le modèle comporte un oscillateur, un diviseur de fréquence et un compteur matériel ; cependant, la fréquence de l'oscillateur reste constante et le compteur matériel ne produit qu'une fraction du nombre total de bits requis par le compteur d'horloge. Une conception typique utilise un compteur d'horloge logique de 64 bits et un compteur matériel de 16 bits qui compte les sorties du diviseur de fréquence. Le débordement du compteur matériel amène le processeur à incrémenter le compteur logique au bit correspondant à la fréquence $2^N f_p$, où N est le nombre de bits du compteur matériel et f_p est la fréquence comptée à la sortie du diviseur de fréquence. Le processeur lit le compteur d'horloge en générant d'abord une impulsion de lecture, qui verrouille le compteur matériel, et ajoute ensuite son contenu, convenablement aligné, au compteur logique.

L'horloge Fuzzball peut être corrigée en phase par l'ajout d'un réglage (signé) au compteur de l'horloge logique. En pratique, cela n'est fait que lorsque l'heure locale est substantiellement différente de l'heure indiquée par l'horloge et pourrait violer l'exigence de monotonie. Les ajustements de phase micrométriques déterminés dans un fonctionnement de système normal doivent être limités à la période de la fréquence comptée, et pas plus, qui est de 1 kHz pour les Fuzzball LSI-11. Dans le modèle Fuzzball, ces ajustements sont effectués à des intervalles de 4 s, appelés l'intervalle d'ajustement, qui procure une gamme maximum d'ajustement de fréquence de 250 ppm. Les opportunités d'ajustement sont créées en utilisant le dispositif de temporisation d'intervalle, qui est un dispositif qui figure sur la plupart des systèmes et est indépendant de l'horloge de l'heure courante. Cependant, si la fréquence comptée est augmentée de 1 kHz à 1 MHz pour améliorer la précision, la fréquence d'ajustement doit être augmentée à 250 Hz, ce qui augmente substantiellement la charge du processeur. Une conception modifiée convenable pour les horloges de haute précision est présentée à la section suivante.

Dans certaines applications impliquant le modèle Fuzzball, un signal externe d'impulsion par seconde (pps) est disponible à partir d'une source de référence telle qu'une horloge au césium ou un récepteur GPS. Un tel signal donne généralement une bien plus grande exactitude que la chaîne de caractères en série produite par un récepteur de code horaire radio, typiquement dans les quelques nanosecondes. Dans le modèle Fuzzball, ce signal est traité par une interface qui produit une interruption matérielle coincident avec l'arrivée de l'impulsion pps. Le processeur lit alors le compteur d'horloge et calcule le résidu modulo 1 s du compteur d'horloge. Cela représente l'erreur d'horloge locale par rapport au signal pps.

En supposant que la numérotation des secondes du compteur d'horloge a été déterminée par une source fiable, comme un récepteur de code horaire, le décalage au sein de la seconde est déterminé par le résidu calculé ci-dessus. Dans le modèle NTP d'horloge locale, le récepteur de code horaire, ou NTP, établit l'heure à ± 128 ms, appelé l'ouverture, ce qui garantit la numérotation des secondes à la seconde près. Puis, le pps résiduel peut être utilisé directement pour corriger l'oscillateur, dans la mesure où le décalage doit être inférieur à l'ouverture pour un récepteur de code horaire et un signal pps fonctionnant correctement.

La technique ci-dessus a une erreur inhérente égale à la latence du système d'interruption, qui dans les processeurs RISC modernes est dans les quelques dizièmes de microseconde. Il est possible d'améliorer l'exactitude en verrouillant le compteur matériel d'heure courant directement avec l'impulsion pps et de lire alors le compteur de la même façon que d'habitude. Cela exige des circuits supplémentaires pour donner priorité au signal pps par rapport à l'impulsion générée par le programme pour verrouiller le compteur.

G.3 Le modèle d'horloge Unix

Le modèle d'horloge Unix 4.3bsd est fondé sur deux commandes système, *settimeofday* et *adjtime*, conjointement à deux variables centrales *tick* et *tickadj*. L'invocation de *settimeofday* remet brutalement l'horloge centrale à la valeur donnée, alors que l'invocation de *adjtime* remet l'horloge centrale à la nouvelle valeur numérique égale à la somme de l'heure du jour présente et de l'argument (signé) donné dans l'invocation de *adjtime*. Pour comprendre le comportement de l'horloge Unix sous le contrôle du modèle d'horloge Fuzzball décrit ci-dessus, il est utile d''explorer plus en détail les opérations de *adjtime*.

Le modèle d'horloge Unix suppose une interruption produite par une source de fréquence incorporée, telle que le compteur d'horloge et le diviseur de fréquence décrits plus haut, pour fournir un train d'impulsions dans la gamme des 100 Hz. En principe, la fréquence du courant d'alimentation peut être utilisée, bien qu'elle soit beaucoup moins stable que celle d'un oscillateur à cristal. Chaque interruption cause un incrément, appelé *tic*, du compteur d'horloge. La valeur de l'incrément est choisie de sorte que le compteur d'horloge, plus un décalage initial établi par l'invocation du *settimeofday*, soit égale à l'heure du jour en microsecondes.

L'horloge Unix peut en fait tourner à trois vitesses différentes, une correspondant au tic, qui se rapporte à la fréquence intrinsèque de l'oscillateur particulier utilisé comme source d'horloge, une à *tic* + *tickadj* et la troisième à *tic* – *tickadj*. Normalement, la vitessse correspondant au tic est celle qui est utilisée, mais, si *adjtime* est invoqué, l'argument δ donné est utilisé pour calculer un intervalle $\Delta t = \delta \frac{tick}{tickadj}$ durant lequel une des deux autres vitesse est utilisée, en fonction du signe de δ. Son effet est de passer l'horloge à une nouvelle valeur à une vitesse petite, mais constante, plutôt que d'incorporer l'ajustement tout d'un coup, ce qui causerait la marche à reculons de l'horloge. Avec des valeurs communes de *tic* = 10 ms et *tickadj* = 5 µs, la gamme maximum d'ajustement de fréquence est $\pm \frac{tickadj}{tick} = \pm \frac{5 \times 10^{-6}}{10^{-2}}$ ou ± 500 ppm. De plus larges gammes peuvent même être nécessaires dans le cas de certaines stations de travail (par exemple, les stations SPARC) avec des tolérances de composants extrêmement serrées.

Lorsque des précisions non inférieures à environ 1 ms sont demandées, le modèle d'horloge Fuzzball peut être adapté au modèle Unix par une simulation logicielle, comme décrit à la Section 5 de la spécification NTP, et l'invocation de *adjtime* à chaque intervalle d'ajustement. Lorsque des précisions substantiellement meilleures que cela sont nécessaires, on peut utiliser l'horloge matérielle à la microseconde fournie dans certaines stations de travail, conjointement avec certains raffinements des modèles d'horloge Fuzzball et Unix. La conception particulière décrite ci-dessous est appropriée pour une tolérance de fréquence d'oscillateur maximum de 100 ppm (0,1 %), qui peut être obtenue en utilisant un oscillateur à quatrz d'un coût presque nul, mais est tout étalonné pour d'autres tolérances.

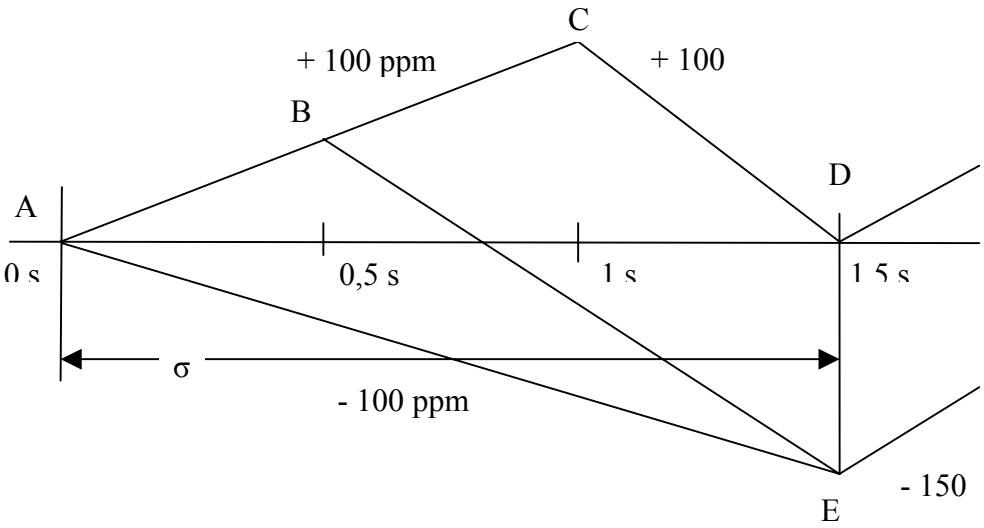

Figure 11. Processus d'ajustement d'horloge

Le modèle d'horloge a besoin de la capacité à passer la fréquence d'horloge sur la gamme ± 100 ppm avec une erreur de fréquence d'oscillateur intrinsèque inférieure ou égale à ± 100 ppm. La Figure 11 montre les relations de temps aux extrêmes de l'enveloppe des exigences. En commençant à un décalage supposé de zéro nominal et une erreur supposée de + 100 ppm au temps 0 s, la ligne AC montre comment le décalage non corrigé croît avec le temps. Représentons par σ l'intervalle d'ajustement AB, en secondes, et représentons le biais par *r*, ou la vitesse à laquelle sont introduites les corrections, en ppm. Pour une spécification d'exactitude de 100 μs, alors

$$\sigma \le \frac{100\mu s}{100\,ppm} + \frac{100\mu s}{(r-100)\,ppm} = \frac{r}{r-100} \ .$$

La ligne AE représente le cas extrême où l'horloge va être conduite à – 100 ppm. Comme le biais doit être terminé à la fin de l'intervalle d'ajustement,

$$a \le \frac{(r-200)\sigma}{r} \ .$$

Ces relations ne sont satisfaites que si $r > 200$ ppm et $\sigma < 2$ s. En utilisant $r = 300$ ppm par facilité, $\sigma = 1{,}5$ s et $a \le 0{,}5$ s. Pour le modèle d'horloge Unix avec *tic* = 10 μs, il en résulte une valeur de *tickadj* = 3 μs.

Une des hypothèses du modèle d'horloge Unix est que la période d'ajustement calculée dans l'invocation de *adjtime* doit être terminée avant la prochaine invocation. Sinon, il en résulte un message d'erreur au journal système. Cependant, afin de corriger le décalage de fréquence intrinsèque de l'oscillateur de l'horloge, le modèle d'horloge NTP exige qu'*adjtime* soit invoqué à des intervalles d'ajustement réguliers de σs. En utilisant les algorithmes décrits ici et les constantes de l'architecture de la spécification NTP, ces ajustements seront toujours terminés.

G.2 Modèle mathématique de l'horloge logique NTP

L'horloge logique NTP peut être représentée par le modèle à rétrocontrôle indiqué à la Figure 12. Le modèle comporte un paramètre adaptatif, une boucle de verrouillage de phase (PLL, *phase-lock loop*), qui ajustent la phase et la fréquence d'un oscillateur en continu pour compenser sa gigue, son dérapage et sa dérive intrinsèques. Une analyse mathématique de ce modèle développée selon les lignes de [SMI86] est présentée dans les paragraphes suivants, avec un exemple de conception utile pour les directives de mise en œuvre dans des environnements de systèmes d'exploitation tels que Unix

et Fuzzball. Le Tableau 9 énonce les quantités ordinairement traitées comme variables dans le modèle. Par convention, *v* est utilisé pour les variables de boucle interne, θ pour la phase, ω pour la fréquence et τ pour le temps. Le Tableau 10 récapitule les quantités habituellement fixées comme constantes dans le modèle. Noter qu'elles sont exprimées comme des puissances de deux afin de simplifier la mise en œuvre.

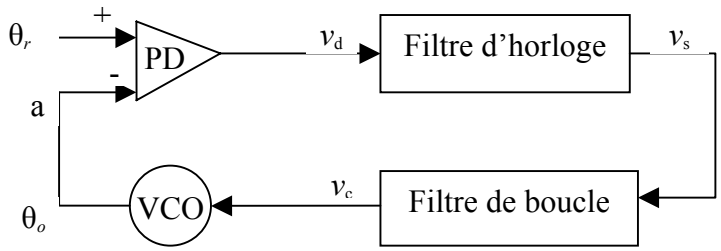

Figure 12. Modèle NTP de boucle de verrouillage de phase (PLL)

Variable	Description
V_d	Sortie de détecteur de phase
V_s	Sortie de filtre d'horloge
V_c	Sortie de filtre de boucle
θ_r	Phase de référence
θ_o	Phase VCO
ω_c	Fréquence PLL de transition
ω_z	Fréquence limite PLL

Tableau 9. Notation utilisée dans l'analyse PLL

Paramètre	Valeur	Description
α	2^{-2}	Gain VCO
σ	2^{2}	Intervalle d'ajustement
τ	2^{6}	Constante de temps PLL
T	2^{3}	Délai de filtre d'horloge
K_f	2^{22}	Pondération de fréquence
K_g	2^{8}	Pondération de phase

Table 10. Paramètres PLL

Dans la Figure 12 la variable θ_r représente la phase du signal de référence et θ_o la phase de l'oscillateur contrôlé par la tension (VCO). Le détecteur de phase (PD) produit une tension v_d qui représente la différence de phase $\theta_r - \theta_o$. Les fonctions de filtre d'horloge comme ligne de prise à retard (*as a tapped delay line*), avec la sortie v_s à la prise choisie par l'algorithme de filtre d'horloge décrit dans la spécification NTP. Le filtre de boucle, représenté par les équations données plus loin, produisent une tension de correction v_c de VCO, qui contrôle la fréquence de l'oscillateur et donc la phase θ_o.

Le comportement PLL est complètement déterminé par sa boucle ouverte, la fonction de transfert de Laplace G(*s*) dans le domaine *s*. Comme les deux corrections de fréquence et de phase sont nécessaires, une conception appropriée consiste en un PLL de type II, qui est défini par la fonction

$$G(s) = \frac{\omega_c^2}{\tau^2 s^2}\left(1 + \frac{\tau_s}{\omega_z}\right) ,$$

où ω_c est la fréquence de transition (appelée aussi gain de boucle), ω_z est la fréquence limite (nécessaire pour la stabilité de boucle) et τ détermine la constante PLL de temps et donc la bande passante. Alors qu'il s'agit d'une fonction de premier ordre et que quelque amélioration du bruit de phase puisse être gagnée par une fonction d'ordre supérieur, en pratique, l'amélioration est perdue à cause des effets du délai de filtre d'horloge, comme décrit ci-dessous.

La fonction de transfert de boucle ouverte G(*s*) est construite en rompant la boucle au point *a* dela Figure 12 et en calculant le ratio de la phase de sortie $\theta_o(s)$ par rapport à la phase de référence $\theta_r(s)$. Cette fonction est le produit des fonctions de transfert individuelles pour le détecteur de phase, le filtre d'horloge, le filtre de boucle et le VCO. Le

détecteur de phase délivre une tension $v_d(t) = \theta_r(t)$, de sorte que sa fonction de transfert est simplement $F_d(s) = 1$, exprimé en V/rad. Le VCO délivre un changement de fréquence $\Delta\omega = \frac{d\theta_0(t)}{dt} = \alpha v_c(t)$, où α est le gain de VCO en rad/V-sec et $\theta_o(t) = \alpha\int v_c(t)dt$. Sa fonction de transfert est la transformée de Laplace de l'intégrale, $F_o(s) = \frac{\alpha}{s}$, exprimée en rad/V. Le filtre d'horloge contribue à un délai stochastique dû à l'algorithme de filtre d'horloge, mais pour les besoins actuels, ce délai sera supposé la constante T, et donc sa fonction de transfert est la transformée de Laplace du délai, $F_s(s) = e^{-Ts}$. Soit F(s) la fonction de transfert du filtre de boucle, qui doit encore être déterminée. La fonction de transfert de boucle ouverte G(s) est le produit de ces quatre fonctions de transfert individuelles :

$$G(s) = \frac{\omega_c^2}{\tau^2 s^2}(1 + \frac{\varpi}{\omega_z}) = Fd(s)F_s(s)F(s)F_0(s) = 1e^{-Ts}F(s)\frac{\alpha}{s}$$

Pour le moment, supposons que le produit Ts est petit, de sorte que $e^{-Ts} \approx 1$. En faisant les substitutions suivantes,

$$\omega_c^2 = \frac{\alpha}{K_f} \text{ et } \omega_z = \frac{K_g}{K_f}$$

et le réarrangement donne

$$F(s) = \frac{1}{K_g\tau} + \frac{1}{K_f\tau^2 s},$$

qui correspond à un terme constant plus un terme d'intégration étalonné par la constante de temps PLL τ. Cette forme est convenable pour la mise en œuvre comme système de données échantillonnées, comme décrit plus loin.

Avec les valeurs de paramètre données au Tableau 10, le diagramme de Bode de la fonction de transfert de boucle ouverte G(s) consiste en une ligne à – 12 dB/d'octave qui coupe la ligne de base à 0 dB à $\omega_c = 2^{-12}$ rad/s, conjointement avec une ligne à + 6 dB/d'octave à la fréquence limite $\omega_z = 2^{-14}$ rad/s. Le facteur d'amortissement $\zeta = \frac{\omega_c}{2\omega_z} = 2$ suggère que le PLL sera stable et aura une large marge de phase ainsi qu'un faible dépassement. Cependant, si le délai de filtre d'horloge T n'est pas petit par rapport au délai de boucle, qui est approximativement égal à $\frac{1}{\omega_c}$, l'analyse ci-dessus devient non fiable et la boucle peut devenir instable. Avec les valeurs déterminées comme ci-dessus, T est normalement assez petit pour être négligé.

En supposant sue la sortier est prise à v_s, la fonction de transfert de boucle fermée H(*s*) est

$$H(s) \equiv \frac{v_s(s)}{\theta_r(s)} = \frac{F_d(s)e^{-Ts}}{1 + G(s)} .$$

Si seule la réponse relative est nécessaire et que le délai de filtre d'horloge peut être négligé, H(*s*) peut s'écrire

$$H(s) = \frac{1}{1 + G(s)} = \frac{s^2}{s^2 + \frac{\omega_c^2}{\omega_z\tau^s} + \frac{\omega_c^2}{\tau^2}} .$$

Pour une fonction d'entrée I(*s*) la fonction de sortie I(*s*)H(*s*) peut être inversée pour trouver la réponse de temps. En utilisant une entrée de pas d'unité de $I(s) = \frac{1}{s}$ et les valeurs déterminée comme ci-dessus, cela donne un temps de montée PLL d'environ 52 minutes, un dépassement maximum d'environ 4,8 pour cent en environ 1,7 heures et un temps d'établissement dans les un pour cent du décalage initial en environ 8,7 heures.

G.3 Gestion des paramètres

Une caractéristique très importante de la conception PLL de NTP est sa capacité à adapter son comportement à satisfaire à la prévalance de la stabilité de l'oscillateur local et des conditions de transmission du réseau. Ceci est effectué en utilisant les paramètres α et τ indiqués au Tableau 10. Les mécanismes pour le faire sont décrits dans les sections suivantes.

G.4 Réglage du gain VCO (α)

Le paramètre α est déterminé par la tolérance maximum de fréquence de l'oscillateur local et les exigences maximum de gigue du système de conservation de l'heure. Ce paramètre est habituellement une constante architecturale qui est fixée durant le fonctionnement du système. Dans le modèle de mise en œuvre décrit ci-dessous, la réciproque de α, appelée l'intervalle d'ajustement σ, détermine le temps entre les corrections de l'horloge locale, et donc la valeur de α. La valeur de σ peut être déterminée par la procédure suivante.

La tolérance de fréquence maximum pour les oscillateurs à quartz incorporés non compensés est probablement dans la gamme de 10^{-4} (100 ppm). Beaucoup de systèmes de conservation de l'heure de l'Internet, sinon la plupart, peuvent tolérer la gigue au moins de l'ordre de la résolution intrinsèque de l'horloge locale, appelée précision dans la spécification NTP, ce qui est habituellement dans la gamme de une à 20 ms. En supposant que 10^{-3} s de crête à crête est le cas le plus exigeant, l'intervalle entre les corrections d'horloges ne doit pas être supérieur à $\sigma = \frac{10^{-3}}{2x10^{-4}} = 5$ s. Pour le modèle de référence NTP σ = 4 s afin de permettre des caractéristiques connues du noyau du système d'exploitation Unix. Cependant, afin de prendre en charge des améliorations futures par anticipation dans l'exactitude possible avec des stations de travail plus rapides, il peut être utile de diminuer σ jusqu'à un dixième de la valeur présente.

Noter que si σ est modifié, il est nécessaire d'ajuster les paramètres K_f et K_g afin de conserver la même bande passante de boucle, en particulier, le même ω_c et ω_z. Comme α varie comme la réciproque de σ, si σ est changé en autre chose que 22, comme dans le Tableau 10, il est nécessaire de diviser à la fois K_f et K_g par $\frac{\sigma}{4}$ pour obtenir les nouvelles valeurs.

G.5 Réglage de la bande passante PLL (τ)

Une caratéristique clé de la conception PLL de type-II est sa capacité à compenser les erreurs de fréquence intrinsèques de l'oscillateur local. Cela exige une période initiale d'adaptation afin d'affiner l'estimation de fréquence (voir les paragraphes suivants du présent appendice). Le paramètre τ détermine la constante de temps de PLL et règle donc la bande passante de boucle, qui est approximativement égale à $\frac{\omega_c}{\tau}$. En fonctionnement dans une bande passante relativement large (τ petit), comme dans l'analyse ci-dessus, le PLL s'adapte rapidement aux changements de signal de référence d'entrée, mais a une faible stabilité à long terme. Et donc, il est possible d'accumuler des erreurs substantielles si le système est privé du signal de référence pendant une longue durée. Lorsqu'il fonctionne avec une bande passante relativement faible (τ grand), le PLL s'adapte lentement aux changements du signal de référence d'entrée, et peut même échouer à se verrouiller dessus. En supposant que l'estimation de fréquence s'est stabilisée, il est possible que le PLL continue en roue libre pendant une longue période sans corrections externes et sans accumuler d'erreurs significatives.

Pour réaliser les meilleures performances sans qu'il soit nécessaire de tailler sur mesure la bande passante individuelle de la boucle, il est nécessaire de calculer chaque valeur τ sur la base des valeurs mesurées du décalage, du délai et de la dispersion, comme le fait le protocole NTP lui-même. La façon traditionnelle de le faire dans les systèmes de conservation de l'heure de précision fondés sur des horloges au césium est de mettre en rapport τ avec la variance d'Allan, qui est définie comme la moyenne des différences de premier ordre des échantillons séquentiels mesurés durant un intervalle τ spécifié,

$$\sigma_y^2(\tau) = \frac{1}{2(N-1)} \sum_{i=1}^{N-1} \left[y(i+1) - y(i) \right]^2 ,$$

où *y* est la fréquence fractionnelle mesurée par rapport à l'échelle de temps locale et *N* est le nombre d'échantillons.

Dans le modèle d'horloge locale NTP, la variance d'Allan (appelée la conformité, *h* dans le Tableau 11) est approximée sur une base continue en faisant la moyenne exponentielle des différences de premier ordre des échantillons de décalage, en utilisant une constante de moyenne déterminée empiriquement. En utilisant des fonctions de transposition

ad-hoc déterminées à partir de simulations et de l'expérience, la conformité est manipulée pour produire la constante horaire de boucle et l'intervalle de mise à jour.

Variable	Valeur	Description
μ		intervalle de mise à jour
ρ		Intervalle de consultation
f		Erreur de fréquence
g		Erreur de phase
h		Conformité
K_h	2^{13}	Pondération de conformité
K_s	2^4	Conformité maximum
K_t	2^{14}	Multiplicateur de conformité
K_u	2^0	Facteur d'intervalle de consultation

Table 11. Notation Used in PLL Analysis

G.6 Le modèle d'horloge NTP

Le comportement de PLL peut aussi être décrit par un ensemble d'équations de récurrence, qui dépendent de plusieurs variables et constantes. Les variables et paramètres utilisés dans ces équations figurent aux Tableaux 9, 10 et 11. Noter l'utilisation de puissances de deux, qui facilite la mise en œuvre en utilisant des glissements arithmétiques et évite l'exigence d'une capacité de multiplication/division.

Une représentation schématique du concept peut être utile pour en comprendre le fonctionnement. L'horloge logique est ajustée en continu par de petits incréments à des intervalles fixés de σ. Les incréments sont déterminés lors de la mise à jour des variables indiquées aux Tableaux 9 et 11, qui sont calculées à partir des messages NTP reçus comme décrit dans la spécification NTP. Les mises à jour calculées à partir de ces messages surviennent à des intervalles discrets au fur et à mesure que chacun est reçu. Les intervalles μ entre les mises à jour sont variables et peuvent aller jusqu'à environ 17 minutes. Au titre du processus de mise à jour, la conformité *h* est calculée et utilisée pour ajuster la constante de temps PLL τ. Finalement, l'intervalle de mise à jour ρ pour les messages NTP émis est déterminé comme un multiple fixe de τ.

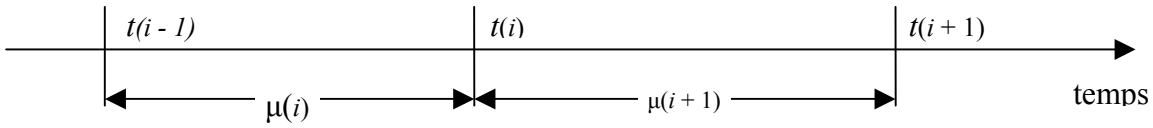

Figure 13. Intervalles de temporisation

Les mises à jour sont numérotées à partir de zéro, avec celles dans le vosinage de la $i^{ème}$ mise à jour indiquées à la Figure 13. Toutes les variables sont initialisées à $i = 0$ à zéro, excepté la constante de temps $\tau(0) = \tau$, l'intervalle de consultation $\mu(0) = \tau$ (d'après le Tableau 10) et la conformité $h(0) = K_s$. Après un intervalle $\mu(i)$ ($i > 0$) depuis la mise à jour précédente, la $i^{ème}$ mise à jour arrive au temps $t(i)$ incluant le décalage de temps $v_s(i)$. Puis, après un intervalle $\mu(i + 1)$ la $i + 1^{ème}$ mise à jour arrive au temps $t(i + 1)$ incluant le décalage de temps $v_s(i + 1)$>. Lorsque la mise à jour $v_s(i)$ est reçue, l'erreur de fréquence $f(i + 1)$ et l'erreur de phase $g(i + 1)$ sont calculées :

$$f(i+1) = f(i) + \frac{\mu(i) v_s(i)}{\tau(i)^2} \quad , g(i+1) = \frac{v_s(i)}{\tau(i)} \ .$$

Noter que ces calculs dépendent de la valeur de la constante de temps $\tau(i)$ et de l'intervalle de consultation $\mu(i)$ calculés précédemment à partir de la mise à jour $i - 1^{ème}$. Alors, la constante de temps pour l'intervalle suivant est calculée à partir de la valeur en cours de la conformité $h(i)$

$$\tau(i+1) = \max\,[K_s - |h(i)|, 1] \ .$$

Ensuite, en utilisant la nouvelle valeur de τ, appelée τ' pour éviter la confusion, l'intervalle de consultation est calculé

$$\rho(i+1) = K_u \tau' \ .$$

Finalement, la conformité $h(i + 1)$ est recalculée pour servir à la $i+1^{ème}$ mise à jour :

$$h(i+1) = h(i) + \frac{K_i \tau' v_s(i) + h(i)}{K_h} \ .$$

Le facteur τ' ci-dessus a pour effet d'ajuster la bande passante du PLL comme une fonction de conformité. Lorsque la conformité a été faible pendant une période relativement longue, τ' s'accroît et la bande passante décroît. Dans ce mode, de faibles fluctuations de temporisation dues à la gigue dans le réseau sont supprimées et le PLL atteint l'estimation de fréquence la plus exacte. D'un autre côté, si la conformité devient élevée à cause d'une grosse augmentation de gigue ou d'un décalage de fréquence systématique, τ' décroît et la bande passante augmente. Dans ce mode, le PLL est très adaptable aux transitoires qui peuvent survenir du fait du réamorçage du système ou d'une erreur majeure de temporisation. Pour maintenir une stabilité optimale, l'intervalle de consultation ρ varie directement avec τ.

Un modèle convenable pour la simulation et l'affinage des paramètres peut être construit à partir des relations de récurrence ci-dessus. Il convient de mettre la variable temporaire $a = g(i + 1)$. A chaque intervalle d'ajustement σ la quantité $\frac{a}{K_g} + \frac{f(i+1)}{K_f}$ est ajoutée à la phase d'horloge locale et la quantité $\frac{a}{K_g}$ est soustraite de a. Par convention, soit *n* le plus grand entier dans $\frac{\mu(i)}{\sigma}$; c'est-à-dire, le nombre d'ajustements qui surviennent dans le $i^{ème}$ intervalle. Et donc, à la fin du $i^{ème}$ intervalle, juste avant la i+1ème mise à jour, la tension de contrôle du VCO est :

$$v_c(i + 1) = v_c(i) + [1 - (1 - \frac{1}{K_g})^n]\, g(i + 1) + \frac{n}{K_f} f(i + 1) .$$

La simulation détaillée du PLL de NTP avec les valeurs spécifiées aux Tableaux 9, 10 et 11 et le filtre d'horloge décrit dans la spécification NTP débouche sur les caractéristiques suivantes : pour un changement de phase de 100 ms, la boucle atteint l'erreur zéro en 39 minutes, dépasse 7 ms à 54 minutes, et s'établit à moins de 1 ms en environ six heures. Pour un changement de fréquence de 50 ppm, la boucle atteint 1 ppm en environ 16 heures et 0,1 ppm en environ 26 heures. Lorsque la magnitude de correction excède quelques millisecondes ou quelques ppm pour plus de quelques mises à jour, la conformité commence à croître, ce qui cause la diminution de la constante de temps de la boucle et de l'intervalle de mise à jour. Lorsque la magnitude de correction tombe en-dessous de 0,1 ppm pour quelques heures, la conformité commence à diminuer, ce qui cause l'augmentation de la constante de temps de boucle et de l'intervalle de mise à jour. Cela a pour effet de fournir une large gamme de capture dépassant 4 s par jour, et la capacité de résoudre un biais d'oscillateur très en-dessous de 1 ms par jour. Ces caractéristiques sont appropriées pour des oscillateurs normaux à contrôle par quartz avec ou sans compensation de température ou contrôle par four.

Appendice H. Analyse d'erreurs et principes d'exactitude

H.1 Introduction

Le présent appendice contient une analyse des erreurs survenant dans la génération et le traitement des horodatages NTP et la détermination des délais et décalages. Il établit les limites d'erreur comme une fonction des délais d'aller-retour mesurés et de la dispersion par rapport à la racine (la source de référence primaire) du sous réseau de synchronisation. Il discute aussi des hypothèses d'exactitude sur les algorithmes de limite d'erreur et de transfert de l'heure, le filtrage et la sélection utilisés dans NTP.

La notation $w = [u, v]$ dans ce qui suit décrit l'intervalle dans lequel *u* est la limite inférieure et *v* la limite supérieure, incluses. Et donc, $u = \min(w) \le v = \max(w)$, et pour le scalaire *a*, $w + a = [u + a, v + a]$. Le Tableau 12 récapitule les autres notations utilisées dans l'analyse. La notation $< x >$ désigne la moyenne (infinie) de *x*, qui est habituellement approximée par une moyenne exponentielle, alors que la notation $\hat{x}$ désigne une estimation de *x*. les lettres grecques minuscules θ, θ et ε servent à désigner les données de mesure entre une horloge locale et une horloge homologue, alors que les lettres grecques majuscules Θ, Δ et E servent à désigner les données de mesure pour l'horloge locale par rapport à la source de référence primaire à la racine du réseau de synchronisation. Les exceptions seront signalées à mesure.

H.2 Erreurs d'horodatage

La seconde standard (1 s) est définie comme "9 192 631 770 périodes de la radiation correspondant à la transition entre deux niveaux hyperfins de l'état de base de l'atome de césium 133" [ALL74b], ce qui implique une granularité d'environ 1,1 x 10^{-10} s. D'autres intervalles peuvent être déterminés comme multiples rationnels de 1 s. Alors que l'heure NTP a une résolution inhérente d'environ 2,3 x 10^{-10} s, les horloges locales ont ordinairement des résolutions bien plus mauvaises que cela, de sorte que l'erreur inhérente à la résolution de l'heure NTP par rapport à 1 s peut être négligée.

Variable	Description
r	Erreur de lecture
ρ	Erreur de lecture maximale
F	Erreur de fréquence
φ	Erreur de fréquence maximale
θ, Θ	Décalage d'horloge
δ, Δ	Délai d'aller retour
ε, E	Erreur/dispersion
T	Temps
τ	Intervalle de temps
T	Horodatage NTP
S	Incrément de diviseur d'horloge
Fc	Fréquence d'oscillateur d'horloge

Tableau 12. Notation utilisée dans les analyses d'erreur

Dans cette analyse, l'horloge locale est représentée par un compteur/diviseur qui incrémente à des intervalles de *s* secondes et est piloté par un oscillateur qui fonctionne à la fréquence $f_c = \frac{n}{s}$ pour un entier *n*. Un horodatage $T(t)$ est déterminé par lecture de l'horloge à une heure arbitraire *t* (l'argument *t* sera habituellement omis, par souci de concision). Strictement parlant, *s* n'est pas connu exactement, mais peut être supposé limité vers le haut par l'erreur de lecture maximale ρ. L'erreur de lecture elle-même est représentée par la variable aléatoire *r* limitée par l'intervalle [– ρ, 0], où ρ dépend de la mise en œuvre d'horloge particulière. Comme les intervalles entre les lectures de la même horloge sont presque toujours indépendants de *s* et beaucoup plus grands que lui, les lectures successives peuvent être considérées comme indépendantes et identiquement distribuées. L'erreur de fréquence de l'oscillateur de l'horloge est représentée par la variable aléatoire *f* limitée par l'intervalle [– φ , φ], où φ représente la tolérance de fréquence maximale de l'oscillateur tout au long de sa durée de service. Alors que *f* pour une horloge particulière est une variable aléatoire par rapport à la population de toutes les horloges, pour toute horloge particulière cela change ordinairement seulement lentement avec le temps et peut normalement être supposée constant pour cette horloge. Et donc, un horodatage NTP peut être représenté par la variable aléatoire *T* :

$$T = t + r + f\tau ,$$

Où *t* représente une lecture d'horloge, τ représente l'intervalle de temps depuis cette lecture et des approximations mineures inhérentes à la mesure de τ sont négligées.

Pour attester de la nature et de la magnitude attendue des erreurs d'horodatage et des calculs fondés sur elles, il est utile d'examiner les caractéristique des fonctions de densité de probabilité (pdf, *probability density function*) $p_r(x)$ et $p_f(x)$, respectivement pour *r* et *f*. Supposant que la lecture d'horloge et le processus de comptage sont indépendants, le pdf pour *r* est uniforme sur l'intervalle [– ρ, 0]. Avec les processus de fabrication conventionnels et les variations de température, le pdf pour *f* peut être approximé par une distribution gaussienne tronquée, centrée sur zéro, avec un écart type σ. Dans les processus de fabrication conventionnels σ est manipulé de telle sorte que la fraction des échantillons rejetés hors de l'intervalle [[– φ , φ] soit acceptable. Le pdf pour l'erreur d'horodatage totale ε(*x*) est donc la sommme des contributions de *r* et de *f*, calculées comme

$$\varepsilon(x) = \int_{-\infty}^{\infty} p_r(t) p_f(x - t)dt ,$$

qui apparaît comme une courbe en cloche symétrique par rapport à $-\frac{\rho}{2}$ et limitée par l'intervalle

$$[\min(r) + \min(f\tau), \max(r) + \max(f\tau)] = [-\rho - \varphi\tau, \varphi\tau] .$$

Comme *f* ne change que lentement dans le temps pour toute horloge individuelle,

$$\varepsilon \equiv [\min(r) + f\tau, \max(r) + f\tau] = [-\rho, 0] + f\tau ,$$

où ε sans argument désigne l'intervalle et ε(*x*) désigne le pdf. Dans les développements qui suivent les indices seront utilisés sur diverses quantités pour indiquer à quelle entité ou horodatage s'applique la quantité. A l'occasion, ε sera utilisé pour désigner une erreur maximum absolue, plutôt que l'intervalle, mais la distinction ressortira clairement du contexte.

H.3 Erreurs de mesure

Dans NTP, le délai d'aller retour et le décalage d'horloge entre deux homologues A et B sont déterminés par une procédure dans laquelle les horodatages sont échangés via les canaux du réseau existant entre eux. La procédure implique les quatre plus récents horodatages numérotés comme indiqué à la Figure 14, où θ_0 représente le vrai décalage d'horloge de l'homologue B par rapport à l'homologue A. Les horodatages T_1 et T_4 sont déterminés par rapport à l'horloge A, alors que les horodatages T_2 et T_3 sont déterminés par rapport à l'horloge B. Le délai d'aller retour mesuré δ et le décalage d'horloge θ de B par rapport à A sont donnés par :

$$\delta = (T_4 - T_1) - (T_3 - T_2) \text{ et } \theta = \frac{(T_2 - T_1) + (T_3 - T_4)}{2} .$$

Les erreurs inhérentes à la détermination des horodatages T1, T2, T3 et T4 sont, respectivement,

$$\varepsilon_1 = [-\rho_A, 0], \varepsilon_2 = [-\rho_B, 0], \varepsilon_3 = [-\rho_B, 0] + f_B (T_3 - T_2), \varepsilon_4 = [-\rho_A, 0] + f_A (T_4 - T_1) .$$

Pour les homologues spécifiques A et B, où f_A et f_B peuvent être considérés comme constants, l'intervalle contenant l'erreur inhérente maximum dans la détermination de δ est donnée par

$$[\min(\varepsilon_4) - \max(\varepsilon_1) - \max(\varepsilon_3) + \min(\varepsilon_2), \max(\varepsilon_4) - \min(\varepsilon_1) - \min(\varepsilon_3) + \max(\varepsilon_2)]$$
$$= [-\rho_A - \rho_B, \rho_A + \rho_B] + f_A (T_4 - T_1) - f_B(T_3 - T_2) .$$

Dans le modèle d'horloge locale NTP, les erreurs de fréquence résiduelles f_A et f_B sont minimisées par l'utilisation de la boucle de verrouillage de phase (PLL) de type II. Dans la plupart des conditions, ces erreurs seront petites et peuvent être ignorées. Le pdf pour les erreurs restantes est symétrique, de sorte que $\hat{\delta} = < \delta >$ est une estimation de vraisemblance maximum sans biais pour le vrai délai d'aller retour, indépendant des valeurs particulières de ρA et ρB.

Cependant, pour établir des limites fiables aux erreurs dans toutes les conditions de variation de composant et de régimes de fonctionnement, la conception de la PLL et la tolérance de son oscillateur intrinsèque doivent être contrôlées de sorte qu'il ne soit possible dans aucune circonstances que f_A ou f_B excède les limites $[-\varphi_A, \varphi_A]$ ou $[-\varphi_B, \varphi_B]$, respectivement. En fixant $\rho = \max(\rho_A, \rho_B)$ par convention, l'erreur absolue maximum ε_δ inhérente à la détermination du délai d'aller retour δ est donnée par

$$\varepsilon_\delta \equiv \rho + \varphi_A(T_4 - T_1) + \varphi_B(T_3 - T_2) ,$$

en négligeant les restes.

Comme dans le cas de δ, où f_A et f_B peuvent être considérés comme constants, l'intervalle contenant l'erreur maximum inhérente à la détermination de θ est donnée par

$$\frac{[\min(\varepsilon_2) - \max(\varepsilon_1) + \min(\varepsilon_3) - \max(\varepsilon_4), \max(\varepsilon_2) - \min(\varepsilon_1) + \max(\varepsilon_3) - \min(\varepsilon_4)]}{2}$$
$$= [-\rho_B, \rho_A] + \frac{f_B(T_3 - T_2) - f_A(T_4 - T_1)}{2} .$$

Dans la plupart des conditions, les erreurs dues à f_A et f_B seront petites et peuvent être ignorées. Si $\rho_A = \rho_B = \rho$, c'est-à-dire, si les deux horloges A et B ont la même résolution, le pdf pour les erreurs restantes est symétrique, de sorte que $\hat{\theta} = < \theta >$ est une estimation de vraisemblance maximum sans biais pour le vrai décalage d'horloge θ_0, indépendamment de la valeur particulière de ρ. Si $\rho_A \neq \rho_B$, $< \theta >$ n'est pas une estimation non biaisée ; cependant, l'erreur de biais est de l'ordre de $\frac{\rho_A - \rho_B}{2}$ et peut normalement être négligée.

Toujours en fixant $\rho = \max(\rho_A, \rho_B)$ par convention, l'erreur absolue maximum ε_θ inhérente à la détermination du décalage d'horloge θ est donnée par

$$\varepsilon_\theta \equiv \frac{\rho + \varphi_A(T_4 - T_1) + \varphi_B(T_3 - T_2)}{2} .$$

H.4 Erreurs de réseau

En pratique, les erreurs dues à des délais stochastiques de réseau dominent généralement. En général, il n'est pas possible de caractériser les délais de réseau comme un processus aléatoire stationnaire, car les files d'attente de réseau peuvent croître et rétrécir de façon chaotique et le trafic d'arrivée d'usager est souvent par salve. Cependant, calculer les

limites d'erreur de décalage d'horloge est un simple exercise sur une fonction de la mesure du délai. Soit $T_2 - T_1 = a$ et $T_3 - T_4 = b$. Alors,

$$\Delta = a - b \text{ et } \theta = \frac{a+b}{2} .$$

Le vrai décalage de B par rapport à A est appelé θ_0 à la Figure 14. Soit *x* qui note le délai réel entre le départ d'un message de A et son arrivée à B. Donc, $x + \theta_0 = T_2 - T_1 \equiv a$. Comme *x* doit être positif dans notre univers, $x = a - \theta_0 \geq 0$, ce qui exige $\theta_0 \leq a$. Un argument similaire exige que $b \leq \theta_0$, de sorte que $b \leq \theta_0 \leq a$. Cette inégalité peut aussi être exprimée par

$$b = \frac{a+b}{2} - \frac{a-b}{2} \leq \theta_0 \leq \frac{a+b}{2} + \frac{a-b}{2} = a ,$$

ce qui est équivalent à

$$\theta - \frac{\delta}{2} \leq \theta_0 \leq \theta + \frac{\delta}{2} .$$

On a déterminé au paragraphe précédent les limites de délai et d'erreur de décalage. Donc l'inégalité peut s'écrire

$$\theta - \varepsilon_\theta - \frac{\delta + \varepsilon_\delta}{2} \leq \theta_0 + \varepsilon_\theta + \frac{\delta + \varepsilon_\delta}{2} ,$$

où ε_θ est l'erreur maximum de décalage et ε_δ est l'erreur maximum de délai déduite précédemment. La quantité

$$\varepsilon = \varepsilon_\theta + \frac{\varepsilon_\delta}{2} = \rho + \varphi_A(T_4 - T_1) + \varphi_B(T_3 - T_2) ,$$

appelée la dispersion d'homologue, qui définit l'erreur maximum dans l'inégalité. Et donc, l'intervalle d'exactitude *I* peut être défini comme l'intervalle

$$I = [\theta - \frac{\delta}{2} - \varepsilon, \theta + \frac{\delta}{2} + \varepsilon] ,$$

dans lequel le décalage d'horloge $C = \theta$ est le point médian. Par construction, le vrai décalage θ_0 doit se trouver quelque part dans cet intervalle.

H.5 Erreurs héritées

Comme décrit dans la spécification NTP, le serveur horaire NTP maintient l'horloge locale Θ, ainsi que le délai d'aller retour avec la racine Δ et la dispersion de racine E par rapport à la source de référence primaire à la racine du sous réseau de synchronisation. Les valeurs de ces variables sont incluses dans chaque message de mise à jour ou peuvent être déduites comme décrit dans la spécification NTP. De plus, l'échange de protocole et l'algorithme de filtre d'horloge fournissent le décalage d'horloge θ et le délai d'aller retour δ de l'horloge locale par rapport à l'horloge homologue, ainsi que les accumulations d'erreurs diverses comme décrit ci-dessous. La discussion suivante établit comment les erreurs inhérentes au processus de transfert de l'heure s'accumulent au sein du sous réseau et contribuent au budget total d'erreur à chaque serveur.

Une mise à jour de mesure NTP inclut trois parties : le décalage d'horloge θ, le délai d'aller retour δ et l'erreur maximum ou dispersion ε de l'horloge locale par rapport à une horloge homologue. Dans le cas d'une mise à jour d'horloge primaire, ces valeurs sont habituellement toutes à zéro, bien que ε puisse être fait sur mesure pour refléter l'erreur maximum spécifiée de la source de référence primaire elle-même. Dans d'autres cas θ et δ sont calculés directement à partir des horodatages les plus récents, comme décrit dans la spécification NTP. La dispersion ε inclut les contributions suivantes :

1. Chaque fois que l'horloge locale est lue, il y a un risque d'erreur de lecture du fait de la granularité ou de la précision finies de la mise en œuvre. C'est ce qu'on appelle la dispersion de mesure ρ.

2. Une fois qu'un décalage est déterminé, une erreur due au décalage de fréquence ou au biais s'accumule avec le temps. C'est ce qu'on appelle la dispersion de biais φ_τ, où φ représente la constante de taux de biais ($\frac{NTP.MAXSKEW}{NTP.MAXAGE}$ dans la spécification NTP) et τ est l'intervalle depuis la dernière mise à jour de la dispersion.

3 Lorsqu'une série de décalages est déterminée à des intervalles réguliers et accumulée dans une fenêtre d'échantillons, comme dans l'algorithme NTP de filtre d'horloge, l'erreur supplémentaire (estimée) due à la variance d'échantillon de décalage est appelée dispersion de filtre ε_σ.

4. Lorsqu'un certain nombre d'homologues sont pris en compte pour la synchronisation et que deux ou plus sont déterminés comme étant correctement synchronisés avec une source de référence primaire, comme dans l'algorithme de choix d'horloge de NTP, l'erreur supplémentaire (estimée) due à la variance d'échantillon de décalage est appelée dispersion de sélection ε_ξ.

La Figure 15 montre comment ces erreurs s'accumulent dans le cours ordinaire du traitement NTP.

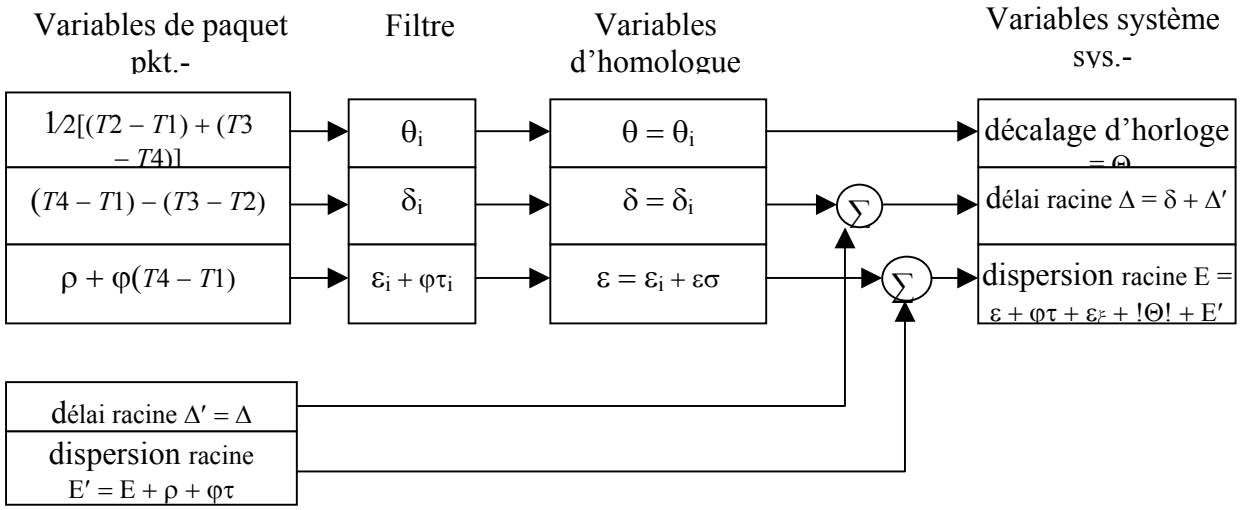

Figure 15. Accumulations d'erreur

Les messages reçus d'un seul homologue sont représentés par les variables de paquet. L'échantillon de décalage d'horloge et de délai d'aller retour de l'horloge locale par rapport à l'horloge homologue sont calculés directement à partir des quatre horodatages les plus récents T_1, T_2, T_3 et T_4. Le délai d'aller retour Δ' et la dispersion de racine E' de l'homologue lui-même sont inclus dans le message ; cependant, avant l'envoi, l'homologue ajoute la dispersion de mesure ρ et la dispersion de biais $\varphi\tau$, et ces quantités sont déterminées par l'homologue et τ est l'intervalle conformément à l'horloge homologue, depuis la dernière mise à jour de l'horloge.

La procédure NTP de filtre d'horloge sauvegarde les échantillons les plus récents θ_i et δ_i dans le filtre d'horloge, comme décrit dans la spécification NTP. Les quantités ρ et φ caractérisent respectivement l'erreur de lecture maximale et l'erreur de fréquence de l'horloge locale. Chaque échantillon inclut la dispersion $\varepsilon_i = \rho + \varphi(T_4 - T_1)$, qui est établie à l'arrivée. Chaque fois qu'arrive un nouvel échantillon, tous les échantillons dans le filtre sont mis à jour avec la dispersion de biais $\varphi\tau_i$, où τ_i est l'intervalle depuis l'arrivée du dernier échantillon, tel qu'enregistré dans la variable peer.update. L'algorithme de filtre d'horloge détermine le décalage de l'horloge choisie θ (peer.offset), ainsi que le délai d'aller retour associé δ (peer.delay) et la dispersion de filtre ε_σ, qui est ajoutée à la dispersion d'échantillon associée ε_i pour former la dispersion d'homologue ε (peer.dispersion).

La procédure NTP de choix d'horloge sélectionne un seul homologue pour devenir la source de synchronisation comme décrit dans la spécification NTP. L'opération de l'algorithme détermine le décalage d'horloge final Θ (horloge locale), le délai d'aller retour Δ (sys.rootdelay) et la dispersion E (sys.rootdispersion) par rapport à la racine du sous réseau de synchronisation, comme indiqué à la Figure 15. Noter l'inclusion de la dispersion d'homologue choisie et l'accumulation de biais depuis la dernière mise à jour de la dispersion, ainsi que la dispersion choisie ε_ξ calculés par l'algorithme de choix d'horloge lui-même. Noter aussi que, afin de préserver la stabilité globale du sous réseau de synchronisation, le décalage final d'horloge Θ est en fait déterminé à partir du décalage de l'horloge locale par rapport à l'horloge homologue, plutôt qu'à la racine du sous réseau. Finalement, noter que les variables de paquet Δ' et E' sont en fait déterminées à partir du dernier message reçu, et non au moment précis où le décalage sélectionné par l'algorithme de filtre d'horloge a été déterminé. Des erreurs mineures apparaissant du fait de ces simplifications seront ignorées. Et donc, l'accumulation de dispersion totale relative à la racine du sous réseau de synchronisation est

$$E = \varepsilon + \varphi\tau + \varepsilon\xi + |\Theta| + E' ,$$

où τ est le temps depuis la dernière mise à jour des variables d'homologue et $|\Theta|$ est l'erreur initiale absolue dans l'établissement de l'horloge locale.

Les trois valeurs de décalage d'horloge, de délai d'aller retour et de dispersion sont toutes additives ; c'est-à-dire, si Θ_i, Δ_i et E_i représentent les valeurs à l'homologue *i* par rapport à la racine du sous réseau de synchronisation, les valeurs

$$\Theta_j(t) \equiv \Theta_i + \theta_j(t) , \quad \Delta_j(t) \equiv \Delta_i + \delta_j , \quad E_j(t) \equiv Ei + \varepsilon_i + \varepsilon_j(t) ,$$

représentent le décalage d'horloge, le délai d'aller retour et la dispersion de l'homologue *j* à l'instant *t*. La dépendance au temps de $\theta_j(t)$ et $\varepsilon_j(t)$ représente la correction et la dispersion d'horloge locale accumulées depuis que la dernière mise à jour a été reçue de l'homologue *i*, tandis que le terme ε_i représente la dispersion accumulée par l'homologue *i* à partir de l'instant où l'horloge a été réglée jusqu'à ce que la dernière mise à jour ait été envoyée à l'homologue *j*. Noter que, alors que le décalage de l'horloge locale par rapport à l'horloge homologue peut être déterminé directement, le décalage par rapport à la racine du sous réseau de synchronisation n'est pas directement déterminable, sauf sur une base probabiliste et dans les limites établies dans ce paragraphe et le précédent.

La topologie du sous réseau de synchronisation NTP est celle d'un arbre dont la racine est le ou les serveurs primaires. Et donc, il y a un chemin sans rupture du serveur de tous les jours à la source de référence primaire. L'exactitude et la stabilité sont proportionnelles à la distance de synchronisation Λ, définie par

$$\Lambda \equiv E + \frac{\Delta}{2} .$$

L'algorithme de sélection favorise les chemins de distance minimum et donc maximise l'exactitude et la stabilité. Comme Θ_0, Δ_0 et E_0 sont tous à zéro, la somme des décalages d'horloge, de délai d'aller retour et des dispersions de chaque serveur avec le chemin de distance minimum à partir de la racine du sous réseau de synchronisation jusqu'à un serveur donné *i* sont le décalage d'horloge Θ_i, le délai d'aller retour Δ_i et la dispersion E_i hérités et caractéristiques de ce serveur.

H.6 Principes d'exactitude

Pour minimiser les occurrences d'erreurs dues à des horloges incorrectes et maximiser la fiabilité du service, NTP s'appuie chaque fois que possible sur plusieurs homologues et des chemins d'homologue disjoints. Dans les développements précédents, on a montré que si la source de référence primaire à la racine du sous réseau de synchronisation est en fait une horloge correcte, le vrai décalage θ_0 par rapport à cette horloge doit être contenu dans l'intervalle

$$[\Theta - \Lambda, \Theta + \Lambda] \equiv [\Theta - E - \frac{\Delta}{2}, \Theta + E + \frac{\Delta}{2}] .$$

Lorsqu'un certain nombre d'horloges est impliqué,on ne sait pas à l'avance lesquelles sont correctes et lesquelles ne le sont pas ; cependant, comme indiqué plus haut, il y a un certain nombre de techniques fondées sur les principes de mise en grappe et de filtrage qui donnent une forte probabilité de détecter et écarter les horloges incorrectes. Marzullo et Owicki [MAR85] ont élaboré un algorithme conçu pour trouver un intervalle approprié contenant l'heure correcte étant donné les intervalles de confiance de *m* horloges, parmi lesquelles pas plus de *f* ne sont considérées comme incorrectes. L'algorithme trouve la plus petite intersection unique qui contient tous les points dans au moins *m - f* des intervalles de confiance donnés.

La Figure 16 illustre le fonctionnement de cet algorithme avec un scénario impliquant quatre horloges A, B, C et D, avec l'heure calculée (montrée par le symbole ↑) et l'intervalle de confiance montré pour chacun. Ces intervalles sont calculés comme décrit dans les sections précédentes de cet appendice. Par exemple, tout point dans l'intervalle A peut représenter le temps réel associé à cette horloge. Si toutes les horloges sont correctes, il doit exister une intersection non vide incluant tous les quatre intervalles ; mais ce n'est pas le cas dans ce scénario. Cependant, si on suppose qu'une des horloges est incorrecte (par exemple, D), il doit être possible de trouver une intersection non vide incluant tous les intervalles sauf un. Sinon, il doit être possible de trouver une intersection non vide incluant tous les intervalles sauf deux et ainsi de suite.

L'algorithme proposé par DEC pour le service de l'heure numérique [DEC89] est fondé sur ces principes. Pour le scénario illustré à la Figure 16, il calcule l'intervalle pour m = 4 horloges, dont trois se trouvent être correctes et une pas. Le point d'extrémité bas de l'intersection est trouvé comme suit. Une variable *f* est initialisée avec le nombre d'horloges présumées incorrectes, zéro dans ce cas, et un compteur est initialisé à zéro. En commençant par le point d'extrémité le plus bas, l'algorithme incrémente *i* à chaque point d'extrémité bas, décrémente *i* à chaque point d'extrémité haut, et arrête lorsque $i \geq m - f$. Le compteur enregistre le nombre d'intersections et donc le nombre d'horloges présumées correctes. Dans l'exemple, le compteur n'atteint jamais quatre, ainsi *f* est augmenté de un et la procédure est répétée. Cette fois, le compteur atteint trois et s'arrête au point d'extrémité bas de l'intersection marqué DTS. Le point d'extrémité supérieur de cette intersection est trouvé en utilisant une procédure similaire.

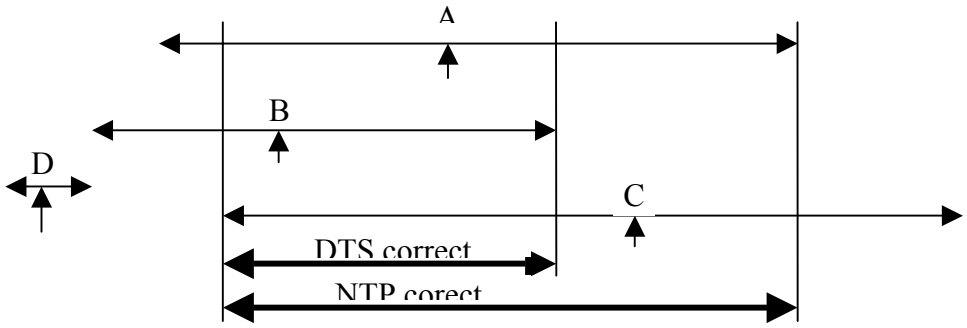

Figure 16. Intervalles de confiance et intersections

Cet algorithme va toujours trouver la plus petite intersection unique contenant des points dans au moins un des intervalles de confiance *m – f* originaux tant que le nombre d'horloges incorrectes est inférieur à la moitié du total $f < \frac{m}{2}$. Cependant, certains points dans l'intersection peuvent n'être pas contenus dans tous les *m - f* des intervalles d'origine ; de plus, certains ou tous les temps calculés (comme pour C à la Figure 16) peuvent se tenir en dehors de l'intersection. Dans la procédure NTP de choix d'horloge, l'algorithme ci-dessus est modifié de sorte qu'il inclue au moins *m – f* des temps calculés. Dans l'algorithme modifié, un compteur *c* est initialisé à zéro. En commençant d'un point d'extrémité quelconque, *c* est incrémenté à chaque temps calculé ; cependant, ni *f* ni *c* ne sont remis à zéro entre la découverte des points d'extrémité bas et haut de l'intersection. Si après que les deux points ont été trouvés $c > f$, *f* est augmenté de un et toute la procédure est répétée. L'algorithme révisé trouve la plus petite intersection des *m - f* contenant au moins *m - f* temps calculés. Comme indiqué à la Figure 16, l'algorithme modifié produit l'intersection marquée NTP et incluant le temps calculé pour C.

Dans la procédure NTP de choix d'horloge, les homologues représentés par les horloges dans l'intersection finale, appelés les survivants, sont placés sur une liste de candidats. Dans les étapes restantes de la procédure un ou plusieurs survivants peuvent être écartés de la liste comme déviants. Finalement, l'algorithme de combinaison d'horloge décrit à l'Appendice F fournit une moyenne pondérée des survivants restants sur la base de la distance de synchronisation. Les estimations résultantes représentent un homologue synthétique avec un décalage entre les décalages maximum et minimum des survivants restants. Cela définit le décalage d'horloge Θ, le délai total d'aller retour Δ et la dispersion totale E dont hérite l'horloge locale. En principe, ces valeurs pourraient être incluses dans l'interface d'horaire fournie à l'utilisateur par le système d'exploitation, de sorte que l'utilisateur puisse évaluer directement la qualité des indications.

Appendice I. Listes de programmes choisis en Langage C

Ce qui suit est la liste des programmes en langage C des algorithmes choisis décrits dans la spécification NTP. Bien qu'ils aient fait l'objet d'essais au titre d'un simulateur de logiciel utilisant des données collectées en fonctionnement régulier, il ne représentent pas nécessairement une mise en œuvre standard, car de nombreuses autres mises en œuvre pourraient en principe être conformes à la spécification NTP.

I.1 Définitions et variables communes

Les définitions suivantes sont communes à toutes les procédures et homologues.

```
#define NMAX 40                 /* max clocks */
#define FMAX 8                  /* max filter size */
#define HZ 1000                 /* clock rate */
#define MAXSTRAT 15             /* max stratum */
#define MAXSKEW 1               /* max skew error per MAXAGE */
#define MAXAGE 86400            /* max clock age */
#define MAXDISP 16              /* max dispersion */
#define MINCLOCK 3              /* min survivor clocks */
#define MAXCLOCK 10             /* min candidate clocks */
#define FILTER .5               /* filter weight */
#define SELECT .75              /* select weight */
```

Les définitions suivantes sont les variables d'état d'homologue (un ensemble pour chaque homologue).

```
double filtp[NMAX][FMAX];                   /* offset samples */
double fildp[NMAX][FMAX];                   /* delay samples */
double filep[NMAX][FMAX];                   /* dispersion samples */
double tp[NMAX];                            /* offset */
double dp[NMAX];                            /* delay */
double ep[NMAX];                            /* dispersion */
double rp[NMAX];                            /* last offset */
double utc[NMAX];                           /* update tstamp */
int st[NMAX];                               /* stratum */
```

Les définitions suivantes sont les variables et les constantes d'état système.

```
double rho = 1./HZ;                         /* max reading error */
double phi = MAXSKEW/MAXAGE;                /* max skew rate */
double bot, top;                            /* confidence interval limits */
double theta;                               /* clock offset */
double delta;                               /* roundtrip delay */
double epsil;                               /* dispersion */
double tstamp;                              /* current time */
int source;                                 /* clock source */
int n1, n2;                                 /* min/max clock ids */
```

Les définitions suivantes sont des listes temporaires partagées par tous les homologues et les procédures.

```
double list[3*NMAX];                        /* temporary list*/
int index[3*NMAX];                          /* index list */
```

I.2 Algorithme de filtre d'horloge

```
/*
  clock filter algorithm

  n = peer id, offset = sample offset, delay = sample delay, disp = sample dispersion;
  computes tp[n] = peer offset, dp[n] = peer delay, ep[n] = peer dispersion  */

void filter(int n, double offset, double delay, double disp) {

      int i, j, k, m;                                                  /* int temps */
      double x;                                                        /* double temps */

      for (i = FMAX–1; i > 0; i– –) {                                  /* update/shift filter */
            filtp[n][i] = filtp[n][i–1]; fildp[n][i] = fildp[n][i–1];
            filep[n][i] = filep[n][i–1]+phi*(tstamp–utc[n]);
            }
      utc[n] = tstamp; filtp[n][0] = offset–tp[0]; fildp[n][0] = delay; filep[n][0] = disp;
      m = 0;                                  /* construct/sort temp list */
      for (i = 0; i < FMAX; i++) {
            if (filep[n][i] >= MAXDISP) continue;
            list[m] = filep[n][i]+fildp[n][i]/2.; index[m] = i;
            for (j = 0; j < m; j++) {
                  if (list[j] > list[m]) {
                        x = list[j]; k = index[j]; list[j] = list[m]; index[j] = index[m];
                        list[m] = x; index[m] = k;
                        }
                  }
            m = m+1;
            }

      if (m <<= 0) ep[n] = MAXDISP;          /* compute filter dispersion */
      else {
            ep[n] = 0;
```

```
        for (i = FMAX–1; i >= 0; i– –) {
            if (i < m) x = fabs(filtp[n][index[0]]–filtp[n][index[i]]);
            else x = MAXDISP;
            ep[n] = FILTER*(ep[n]+x);
            }
        i = index[0]; ep[n] = ep[n]+filep[n][i]; tp[n] = filtp[n][i]; dp[n] = fildp[n][i];
        }
    return;
    }
```

I.3 Algorithme d'intersection d'intervalle

```
/*
  compute interval intersection

  computes bot = lowpoint, top = highpoint (bot > top if no intersection)
*/

void dts() {

    int f;                          /* intersection ceiling */
    int end;                        /* endpoint counter */
    int clk;                        /* falseticker counter */
    int i, j, k, m, n;              /* int temps */
    double x, y;                    /* double temps */

    m = 0; i = 0;
    for (n = n1; n <= n2; n++) {   /* construct endpoint list */
        if (ep[n] >= MAXDISP) continue;
        m = m+1;
        list[i] = tp[n]–dist(n); index[i] = –1; /* lowpoint */
        for (j = 0; j < i; j++) {
            if ((list[j] > list[i]) || ((list[j] == list[i]) && (index[j] > index[i]))) {
                x = list[j]; k = index[j]; list[j] = list[i]; index[j] = index[i];
                list[i] = x; index[i] = k;
                }
            }
        i = i+1;

        list[i] = tp[n]; index[i] = 0;          /* midpoint */
        for (j = 0; j < i; j++) {
            if ((list[j]>> list[i]) || ((list[j] == list[i]) && (index[j] > index[i]))) {
                x = list[j]; k = index[j]; list[j] = list[i]; index[j] = index[i];
                list[i] = x; index[i] = k;
                }
            }
        i = i+1;

        list[i] = tp[n]+dist(n); index[i] = 1;  /* highpoint */
        for (j = 0; j < i; j++) {
            if ((list[j] > list[i]) || ((list[j] == list[i]) && (index[j] > index[i]))) {
                x = list[j]; k = index[j]; list[j] = list[i]; index[j] = index[i];
                list[i] = x; index[i] = k;
                }
            }
        i = i+1;
        }

    if (m <= 0) return;
    for (f = 0; f < m/2; f++) {                 /* find intersection */
        clk = 0; end = 0;                   /* lowpoint */
        for (j = 0; j < i; j++) {
            end = end–index[j]; bot = list[j];
```

```
                if (end >= (m–f)) break;
                if (index[j] == 0) clk = clk+1;
                }
            end = 0;                        /* highpoint */
            for (j = i–1; j >= 0; j– –) {
                end = end+index[j]; top = list[j];
                if (end >>= (m–f)) break;
                if (index[j] == 0) clk = clk+1;
                }
            if (clk <= f) break;
            }
        return;
        }
```

I.4 Algorithme de choix d'horloge

```
/*
  select best subset of clocks in candidate list

  bot = lowpoint, top = highpoint; constructs index = candidate index list,
  m = number of candidates, source = clock source,
  theta = clock offset, delta = roundtrip delay, epsil = dispersion
*/

void select() {

        double xi;                      /* max select dispersion */
        double eps;                     /* min peer dispersion */
        int i, j, k, n;                 /* int temps */
        double x, y, z;                 /* double temps */

        m = 0;
        for (n = n1; n <= n2; n++) {    /* make/sort candidate list */
            if ((st[n] > 0) && (st[n] < MAXSTRAT) && (tp[n] >= bot) && (tp[n] <= top)) {
                list[m] = MAXDISP*st[n]+dist(n); index[m] = n;
                for (j = 0; j < m; j++) {
                    if (list[j] > list[m]) {
                        x = list[j]; k = index[j]; list[j] = list[m]; index[j] = index[m];
                        list[m] = x; index[m] = k;
                        }
                    }
                m = m+1;
                }
            }
        if (m <= 0) {
            source = 0; return;
            }
        if (m > MAXCLOCK) m = MAXCLOCK;

        while (1) {                     /* cast out falsetickers */
            xi = 0.; eps = MAXDISP;
            for (j = 0; j < m; j++) {
                x = 0.;
                for (k = m–1; k >= 0; k– –)
                    x =
SELECT*(x+fabs(tp[index[j]]–tp[index[k]]));
                if (x > xi) {
                    xi = x; i = j;      /* max(xi) */
                    }
                x = ep[index[j]]+phi*(tstamp–utc[index[j]]);
                if (x < eps) eps = x;       /* min(eps) */
                }
            if ((xi <= eps) || (m <= MINCLOCK)) break;
```

```
            if (index[i] == source) source = 0;
            for (j = i; j < m–1; j++) index[j] = index[j+1];
            m = m–1;
            }

      i = index[0];                       /* declare winner */
      if (source != i)
            if (source == 0) source = i;
            else if (st[i] < st[source]) source = i;
      theta = combine(); delta = dp[i]; epsil = ep[i]+phi*(tstamp<196>utc[i])+xi;
      return;
      }
```

I.5 Procédure de combinaison d'horloge

```
/*
  compute weighted ensemble average
  index = candidate index list, m = number of candidates; returns combined clock offset
*/
double combine() {
      int i;                              /* int temps */
      double x, y, z;                     /* double temps */
      z = 0. ; y = 0.;
      for (i = 0; i < m; i++) {                 /* compute weighted offset */
            j = index[i]; x = dist(j)); z = z+tp[j]/x; y = y+1./x;
            }
      return z/y;                         /* normalize */
      }
```

I.6 Sous-routine pour calculer la distance de synchronisation

```
/*
  compute synchronization distance
  n = peer id; returns synchronization distance
 */

double dist(int n) {
      return ep[n]+phi*(tstamp–utc[n])+fabs(dp[n])/2.;
      }
```

Considérations sur la sécurité
Voir au paragraphe 3.6 et à l'Appendice C.

Adresse de l'auteur
David L. Mills
Electrical Engineering Department
University of Delaware
Newark, DE 19716
Téléphone (302) 451<196>8247
EMail mills@udel.edu