1 INTRODUCTION

Nous nous intéressons dans ce document à la conception d'un système de fichiers distribue pour un système de vote électronique à distance. Un système de vote électronique à distance est un système distribué dans lequel un électeur peut voter depuis n'importe quel endroit en utilisant un appareil électronique (smartphone, ordinateur, etc.) connecter à internet. Concevoir des protocoles de vote sécurisés pour ce type de système n'est pas une tâche aisée. Il est très difficile de répondre aux exigences du vote lorsque les protocoles sont utilisés sur un réseau. En particulier, les propriétés de l'incoercibilité, confidentialiter, eligibiliter, uniciter et l'anonymat sont des problèmes majeurs pour ces systèmes. Plusieurs solutions basées sur la cryptographie ont ete proposée dans la littérature.Nous proposons dans ce document un système de fichiers distribués ayant des propriétés pouvant être utilisé pour renforcer la sécurité dans ces systèmes . En effet, ces systèmes sont généralement composés d'un ensemble de machines à voter ou serveurs de votes ayant pour rôle d'enregistrer les données et votes des électeurs et aussi permettre leur lecture lors de la phase de décompte. Ces opérations sont réalisées avec l'aide d'un système de fichiers sous-jacent partagés entre les différentes machines ou serveurs de votes. Le système de fichiers que nous décrivons dans ce document visé à renforcer les propriétés d'éligibilité, et l'unicité sans pour autant nuire aux autres propriétés.Dans ce système de fichiers, un vote est un fichier et la structure des métadonnées de ces fichiers contient les informations relatives aux électeurs et contrairement à plusieurs systèmes de fichiers existants, l'ensemble des opérations que l'on peut effectuer sur un fichier est très réduit.

2 LES PROPRIETES D'UN VOTE SUR ET RELATION AVEC LE SYSTEME DE FICHIER

Les systèmes de vote électronique sont confrontés à de nombreux défis soulevés par les exigences fonctionnelles et constitutionnelles qui sont régies par le pays dans lequel ils fonctionnent. Les systèmes de vote électronique doivent respecter les principes électoraux constitutionnels. Pour les solutions technologiques, cela se traduit par des exigences de sécurité qui doivent être remplies par l'environnement opérationnel dans lequel se déroule le vote. Pour tout système de vote, certaines exigences sont essentielles comme l'authentification, l'unicité, la confidentialité, la fiabilité, la vérifiabilité et l'exactitude. D'autres exigences sont souhaitables, telles que la transparence, l'évolutivité et la rentabilité.Dans cette section, nous définissons les propriétés d'un vote sécurisent tel que décrit par plusieurs chercheurs et donnons les idées de comment certaines de ses propriétés peuvent être renforcé par un système de fichiers distribue.

- **Confidentialité**: tout au long du processus de vote, aucun des choix électoraux d'un électeur n'est connu des autres sauf pour l'électeur lui-même. Plus précisément, bien sûr, nous ne pouvons pas empêcher un électeur de révéler son choix. Cette exigence stipule seulement si un électeur peut suivre le protocole et a la capacité de garder un secret (p. ex. un reçu de vote, une clé privée) pour lui-même, alors non l'un d'entre eux, y compris l'autorité, devrait apprendre les choix de l'électeur.
- **Anonymat** : Personne ne devrait être en mesure de déterminer comment une autre personne a voté.
- **Unicité** : Aucun électeur ne devrait pouvoir voter plus d'une fois. Nul ne peut modifier ou voter a la place d'une autre personne.
- **Équité** : Aucun résultat partiel n'est disponible avant le résultat final.
- **Vérifiabilité** : Les électeurs peuvent vérifier que leur bulletin de vote est compté correctement. Il existe deux types de vérifiabilité : la vérifiabilité individuelle et la vérifiabilité universelle
- **L'incoercibilité** : Aucun électeur ne peut prouver a un autre electeur qu'il a voter pour un candidat specifique.
- **Éligibilité**: Seuls les électeurs répondant à certains critères prédéterminés ont le droit de voter.

- **Intégrité :** Les votes ne devraient pas pouvoir être modifiés sans être détectés**.**
- **Fiabilité :** Le système doit être résistant aux dysfonctionnements générés de manière aléatoire.
- **Robustesse** : Le système de vote devrait continuer de fonctionner même en cas de défaillance partielle du système.
- **Évolutivité :** La complexité des protocoles utilisés dans un système de vote est un facteur majeur dans sa mise en œuvre pratique. Un système de vote efficace doit être évolutif en ce qui concerne les besoins de stockage, de calcul et de communication afin de s'adapter à un plus grand nombre d'électeurs.

Comme mentionner dans l'introduction, un fichier dans notre system de fichier représente un vote et la structure des métadonnées contient les informations du votant. Une façon de renforcer l'anonymat serait de cacher ou briser le lien qui existe entre la structure des métadonnées et le fichier (vote) correspondant sans pour autant perdre l'accès au fichier. Les fichiers étant des votes, la propriété de l'interroger pourrait être renforcé en n'autorisant qu'une seule opération d'écriture sur un fichier une fois qu'il a été créer et modifié une première fois, c'est-à-dire, après la première séquence d'ouverture, écriture et fermeture du fichier, ce dernier devient immuable c'est-à-dire verrouiller en écriture. La propriété de l'éligibilité dit que seuls les électeurs répondant à certains critères prédéterminés ont le droit de voter; Le structure de métadonnée d'un fichier contient les informations relatives à un électeur, par conséquent ces structures peuvent être initialisé dès la phase d'enregistrement des électeurs et ainsi le système de fichiers peut par la suite n'autoriser que la création du fichier pour ces structures-là. L'unicité est une autre propriété importante. Cette propriété s'exprime dans notre système de fichiers par le fait de ne pas autoriser l'existence de deux fichiers ayant la même structure de métadonnées, rappelons que ceci est possible dans la majorisait des système de fichiers existant via le mécanisme des liens physique.

3 CONCEPTS ET SYSTEME DE FICHIER EXISTANTS

Un système de fichiers distribue est un système de fichiers dont les clients, serveurs et périphériques de stockage sont dispersés sur des machines dans un système distribué. La configuration et la mise en œuvre d'un système de fichiers distribuent à varier. Il existe des configurations où les serveurs fonctionnent sur des machines dédiées, ainsi que des configurations ou une machine peuvent être à la fois un serveur et un client. Un système de fichiers distribué peut être implémenté comme module d'un système d'exploitation distribué ou, alternativement, par une couche logicielle dont la tâche est de gérer la communication entre les systèmes d'exploitation et les systèmes de fichiers. Les caractéristiques

tendres distinctives d'un système de fichiers distribués sont la multiplicité et l'autonomie des clients et des serveurs dans le système.Cette section est divisée en deux parties. Dans la première partie, nous discutons des concepts qui sous-tendent la conception d'un système de fichiers distribué. Dans la seconde partie, nous examinons deux exemples du système de fichiers, AFS et NFS. Les principales caractéristiques de chacun sont examinées.

3.1 SYSTEME DE NOMMAGE ET TRANSPARENCE

Le nommage est la correspondance entre les objets logiques et physiques. Les utilisateurs traitent des objets de données logiques représentés par des noms de fichiers, tandis que le système manipule des blocs physiques de données stockées sur des pistes de disque. Habituellement, un utilisateur se réfère à un fichier par un nom textuel. Ce dernier est mappé à un identificateur numérique de niveau inférieur, qui à son tour est mappé à des blocs de disque. Ce mappage à plusieurs niveaux fournit aux utilisateurs une abstraction d'un fichier qui cache les détails de la manière dont et où le fichier est réellement stocké sur le disque. Dans un systeme de fichier distribue transparent, une nouvelle dimension est ajoutée à l'abstraction, celle du masquage de l'emplacement du fichier dans le réseau. Dans un système de fichiers centralizer, la plage du mappage des noms est une adresse dans un disque ; dans un systeme de fichier distribue, elle est augmentée pour inclure la machine spécifique contenant le disque dans lequel le fichier est stocké. La notion de replication est une extension du concept de traiter les fichiers comme des abstractions. Avec un nom de fichier, le mappage renvoie un ensemble des emplacements des répliques de ce fichier. Dans cette abstraction, l'existence de copies multiples et leur emplacement sont cachés.

3.2 SÉMANTIQUE DU PARTAGE

La sémantique du partage est un critère important pour évaluer tout système de fichiers qui permet à plusieurs clients de partager des fichiers. Il s'agit d'une caractérisation du système qui spécifie les effets de l'accès simultané de plusieurs clients à un fichier partagé. En particulier, cette sémantique devrait préciser quand les modifications d’un fichier par un client sont observables, ou pas du tout, par des clients distants. Il est important de rappeller que les applications qui utilisent le système de fichiers pour stocker des données et imposent des contraintes aux accès simultanés afin de garantir la cohérence sémantique de leurs données (c.-à-d. les applications de base de données) devraient utiliser des moyens spéciaux (ex. des verrous) et ne pas se fier à la sémantique sous-jacente de partage que fournit le système de fichiers.

3.3 NFS (Network File Sytem)

NFS considère un ensemble de postes de travail interconnectés comme un ensemble de machines indépendantes avec des systèmes de fichiers indépendants. L'objectif est de permettre un certain degré de partage entre ces systèmes de fichiers d'une manière transparente. Le partage est basé sur la relation serveur-client. Une machine peut être, et est souvent, à la fois un client et un serveur. Le partage est autorisé entre n'importe quelle paire de machines, pas seulement avec des machines serveur dédiées. Le partage d'un système de fichiers distant n'affecte que la machine cliente et aucune autre machine. Par conséquent, il n'y a aucune notion de système de fichiers partagé à l'échelle globale comme dans Locus, Sprite, UNIX United, et Andrew.

3.3.1 Architecture

NFS suit le modele client/serveur pour permettre le partage de fichier entre plusieurs clients distribués. Les serveurs exportent les systèmes de fichiers ou sous-répertoires à monter par les clients. Cependant, il n'y a pas de distinction réelle entre les clients et les serveurs. Les machines peuvent être à la fois clients et serveurs d'autres machines. Cependant, chaque machine doit connaître les serveurs d'où elle importe et les clients vers lesquels elle exporte. Les clients sont au courant de chaque serveur individuel et les serveurs doivent être au courant des clients. Les serveurs définissent également un ensemble d'appels de procédure à distance (RPC) qui permettent aux clients d'accéder aux fichiers à distance. Les RPC comprennent la recherche d'un fichier, la lecture des répertoires, la modification des liens et des répertoires, l'accès aux attributs de fichier et la lecture et l'écriture des fichiers. Tous les appels système du client passent par une couche du système de fichiers virtuel (VFS) qui se trouve au-dessus du système de fichiers local du client. Le VFS intercepte les appels du client qui sont dirigés vers des fichiers du répertoire monté. Ces appels sont ensuite envoyés au client NFS, puis au serveur NFS approprié. Les serveurs sont pour la plupart des serveurs sans état. La version 4 de NFS, cependant, introduit un certain état aux serveurs en gérant le verrouillage des fichiers.

3.3.2 Systeme de nommage

NFS est transparent mais non indépendant de l'emplacement. Chaque serveur peut choisir d'exporter l'ensemble de son système de fichiers ou seulement un sous-arbre. Le serveur doit également indiquer où il exporte et avec quelles permissions. Chaque client qui veut accéder aux fichiers partagés doit monter les fichiers du serveur dans un répertoire sur sa machine locale. Cela peut être à un

emplacement différent dans l'arborescence des répertoires de chaque client. Des montages en cascade peuvent également se produire dans la plupart des implémentations. C'est là que les répertoires se montent sur les répertoires déjà montés répertoires. Cela peut conduire à une hiérarchie très peu structurée. La figure 1 en montre deux les clients qui ont monté des fichiers partagés à partir des deux mêmes serveurs sur un emplacement différent

dans leur arborescence de répertoires. Le client 1 a dir1 du serveur 1 monté sur /usr/local/dir1 et a dir2 du serveur 2 monté sur /usr/dir2. le client 2 a dir1 du serveur 1 monté sur /usr/dir2. /usr/dir1 et a dir2 du serveur 2 monté sur /usr/local/dir2. pour plus de clarté, dir1 de Le serveur 1 est représenté par une ligne pointillée et dir2 du serveur 2 est représenté par un pointillé. ligne. Le Client 1 et le Client 2 voient deux vues différentes des mêmes fichiers. Cela ne donne pas un espace de nom uniforme, comme dans AFS, dont il sera question plus loin. De plus, une fois que les fichiers sont montés, ils ne peuvent pas être déplacés vers un autre serveur. Ceci aurait pour conséquence que les noms des chemins d'accès soient deviennent invalides sur les machines clientes. Cette caractéristique ne permet pas à la NFS à équilibrer la charge des serveurs. usr/ usr/ usr/

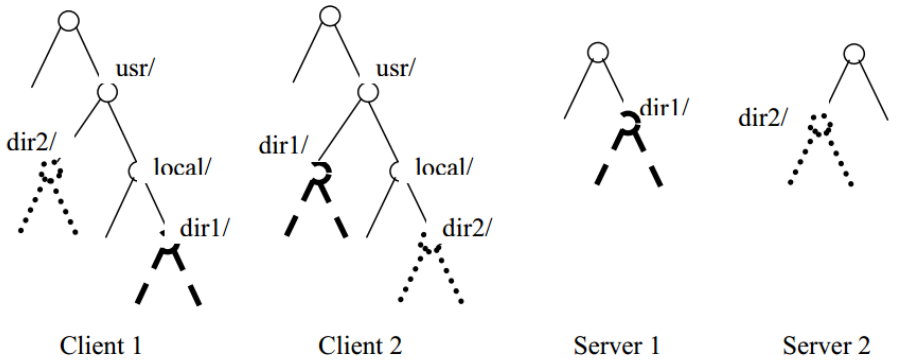

Fig 1

3.3.3 Replication

Il n'y a pas de protocole de réplication standard dans la plupart des implémentations de NFS. Cependant, la version 4 de NFS fournit une migration très limitée du système de fichiers et une réplication en lecture seule. Si le système de fichiers est déplacé vers un nouveau serveur, les clients sont notifiés par un code d'erreur spécial.

Le client peut alors accéder au nouvel emplacement en regardant un champ dans le champ qui peuvent contenir les autres emplacements. Cet attribut peut également contenir les autres emplacements d'un système de fichiers en lecture seule. Cependant, pour qu'un client NFS puisse accéder aux fichiers sur un autre serveur, il doit passer officiellement sur ce serveur. Le nouveau serveur traite alors toutes les requêtes vers les fichiers montés. La version 4 de NFS ne traite pas de la migration des systèmes de fichiers de serveur à serveur atomique ni de la cohérence de la réplication. Cependant, comme la migration et la réplication ne sont disponibles qu'en version 4, lorsqu'un serveur n'est pas disponible dans d'autres implémentations, ses fichiers seront également indisponibles. De plus, comme des montages en cascade peuvent se produire, si un serveur tombe en panne, les répertoires d'autres serveurs encore actifs peuvent également devenir indisponibles. La seule possibilité est de copier des répertoires entiers sur d'autres serveurs et de les monter explicitement sur les machines clientes.

3.4 AFS (Andrew's File System)

Le système de fichiers AFS est un système de fichiers distribué qui utilise un ensemble de serveurs de confiance pour présenter un espace nom de fichier homogène et transparent à tous les postes de travail clients. Il a été développé par l'Université Carnegie Mellon dans le cadre du projet Andrew, initialement appelé "Vice",AFS porte le nom d'Andrew Carnegie et Andrew Mellon. Son utilisation principale est l'informatique distribuée. AFS présente plusieurs avantages par rapport aux systèmes de fichiers en réseau traditionnels, en particulier dans les domaines de la sécurité et de l'évolutivité

3.4.1 Architecture

Comme la plupart des systèmes de fichiers distribués, AFS utilise le modèle informatique client/serveur. Contrairement à NFS, AFS établit une distinction claire entre les machines serveurs et les machines clients. Les serveurs sont peu nombreux et dignes de confiance, tandis que les clients sont nombreux et moins dignes de confiance. Sur les machines clientes ou les postes de travail, l'espace fichier local et l'espace fichier partagé sont séparés. L'espace de fichier partagé est situé sous un répertoire /afs . L'espace partagé sous ce répertoire est géré par un ensemble de serveurs de fichiers dédiés sur un réseau local (LAN) appelé Vice. Les serveurs Vice se connectent également à d'autres serveurs Vice via un WAN (Wide Area Network). Sur chaque machine cliente, il existe un processus appelé Venus qui contrôle l'accès aux fichiers partagés entre le client et Vice. Vice et Venus donnent à chaque utilisateur l'accès à l'espace de noms partagé comme s'il était présent sur leur serveur local. machine. AFS divise les partitions de disque sur les serveurs en unités appelées volumes. Volumes sont petits et se composent d'un ensemble de répertoires et de fichiers connexes. Ils sont formés de la même

façon que les de la même façon qu'une monture fonctionne en NFS. Les volumes sont également une abstraction clé qui permet à AFS de être extrêmement flexible et efficace. Les volumes peuvent être répliqués et déplacés sans la fonction connaissance de l'utilisateur. Cela permet d'équilibrer la charge des serveurs, ce qui n'est pas possible avec NFS

3.4.2 Systeme de nommage

Contrairement à NFS, AFS illustre à la fois la transparence d'accès et l'indépendance de l'emplacement. Il fournit un espace de nom uniforme et du point de vue des utilisateurs, un systeme de fichier existe. La figure 2 illustre cette idée. Le Client 1 et le Client 2 voient tous les deux le
même système de fichiers même si les répertoires dir1 et dir2 sont sur des serveurs différents. Le les noms de fichiers ne révèlent pas l'emplacement physique des fichiers et ils peuvent être déplacés sans qu'il soit nécessaire de changer leurs noms. Les volumes peuvent être déplacés d'un serveur à l'autre complètement
transparent pour l'utilisateur. C'est différent de NFS, qui ne montre que l'emplacement transparence. Ceci est également avantageux car il donne à chaque utilisateur un seul point de montage, ce qui signifie qu'ils peuvent tous être configurés de la même façon. C'est très important pour les grandes entreprises. organisations ayant de nombreux clients

3.4.3 Replication

AFS offre une autre amélioration par rapport à NFS en ce sens qu'il ne replique que les fichiers en lecture seule. La conception de base d'AFS en fait un outil très efficace. Comme AFS est indépendant de l'emplacement, les volumes peuvent exister sur n'importe quel serveur et le même point de vue que celui donné aux clients. Cela permet de répliquer les volumes et de les placer sur différents serveurs. Si Vice détermine que le serveur avec le volume demandé est il peut diriger la demande vers l'une des répliques. C'est important à deux égards. Tout d'abord, elle rend les systèmes AFS plus tolérants aux pannes. C'est parce qu'il y a plus de chances que le le volume désiré sera disponible même si un serveur est inaccessible. Deuxièmement, il améliore la performance. La réplication permet d'acheminer les demandes vers la copie la plus proche, ce qui permet d'obtenir la copie la plus proche. réduit le délai réseau. Il n'est pas non plus nécessaire de contrôler la cohérence de la réplication. parce que seuls les volumes en lecture seule sont répliqués

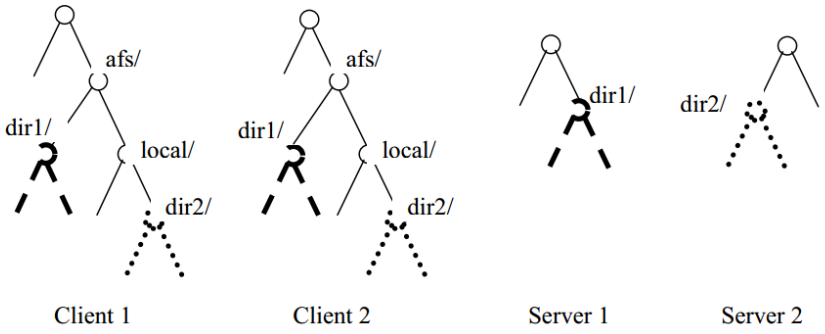

Fig 2

4 LE SYSTEME DE FICHIER PROPOSE

Nous proposons un système de fichiers distribue dédié au vote électronique à distance. Les utilisateurs du système de fichiers sont les processus de votes s'exécutant sur un ou plusieurs serveurs de vote distribuer. L'objectif principal du système de fichiers est de renforcer les propriétés d'intégrité, d'unicité et d'éligibilité d'un vote. Dans ce système de fichiers, les fichiers contiennent le vote des électeurs et la structure de métadonnées de chaque fichier contient les informations relatives à chaque électeur. Dans cette section nous décrivons l'architecture et le fonctionnement global du système de fichiers

4.1 Vue d'ensemble

Le système de fichiers présente un espace de nommage unique et global à tous les processus de votes. Le format de noms des fichiers est identique à celui-ci utiliser dans les systèmes UNIX. Lorsqu'un fichier est créer, celui-ci est répliquer sur tous les serveurs de votes, ceci afin d'augmenter la fiabilité du système. En effet, le système de fichiers étant dédié au vote, la perte de fichiers en cas de panne d'un support de stockage n'est pas toléré car ceci impliquerait la perte des votes correspondants, le système de réplication utiliser est détaillé dans la section 4.2. Pour des besoins de synchronisation entre les serveurs de votes, le système et composer d'un ensemble de serveur de synchronisation. Un ou plusieurs serveurs de votes sont élu pour jouer le rôle de serveur de synchronisation. Dans ce système

de fichiers, un fichier a trois états possibles: invalide, valide, ou en attente de validation. Après la création d'un fichier celui-ci est dans l'état d'attente de validation, la validation ou l'invalidation d'un fichier est effectué par un daemon qui s'exécute de façon périodique (section 4.4). Un fichier est invalidé par le daemon si la structure (l'Inode, voire section 4.1) de ce dernier était déjà rattaché à un fichier au moment de la création du fichier (c'est-à-dire, l'électeur avait déjà voté), dans le cas contraire le fichier est validé. Chaque site où serveur de vote maintient une table de fichier valide, une table invalide et une autre des fichiers en attente de validation. Seuls les fichiers valides sont pris en compte dans la phase de décompte.

4.1) La structure d'inode utiliser

La structure d'Inode est utilisée pour stocker les métadonnées de chaque fichier (taille, date de dernière modification, mode, etc.). Deux différences existe entre l'Inode manipuler dans ce système de fichiers et l'Inode manipulé par les systèmes Linux:

- La présence de deux champs supplémentaires: l'identifiant du votant et l'état de l'Inode. L'identifiant est initialisé lors de la phase d'enregistrement des électeurs et l'état de l'Inode indique si le fichier correspondant est valide ou non. En effet, lors de la phase d'enregistrement des électeurs, une binode est créer et initialise pour chaque électeur. Après la création l'Inode ne pointé sur aucun ne bloque de données, indiquant que l'électeur n'a pas encore voté, l'état de l'Inode est en attente de validation.

- La présence d'une seule adresse se bloque de données et aucune adresse d'indirections. Les fichiers étant des votes, ceux-ci ne sont pas destinés à évoluer où à augmenter en taille et donc un bloc de données d'une taille de l'ordre de kilos octets seront suffisant pour les stocker.

4.2) Systeme de Replication

À la création d'un fichier ce dernier est répliqué sur tous les serveurs de votes. Normalement répliquer un fichier sur tous les postes de travail dans un système de fichiers distribuent n'est pas une bonne idée car plus on a des copies d'un fichier plus la mise à jour de celui-ci et complexe. Étant données que les fichiers que nous manipulons sont des votes et que ces fichiers sont immuables (verrouiller en écriture) nous n'avons pas cette contrainte de mise à jour et répliquer un vote sur tous les postes ont pour effet d'augmentés la fiabilité du système de fichiers. Le processus de réplication est fait de façon

asynchrone, c'est-à-dire, a la création d'un fichier celui-ci n'est pas immédiatement répliquée. Ce processus est réalisé par un processus (daemon) qui s'exécute de façon périodique (section 4.4). Seuls les fichiers valides sont répliqués par le daemon.

4.3) Processus de creation d'un fichier

L'opération de création est la plus importante du système de fichiers car c'est celle qui consiste à enregistrer un vote et le système de fichiers étant distribuer beaucoup de precaution doivent être prise afin d'éviter les problèmes liés à l'unicité du vote.Pour créer un fichier le processus de vote doit passer comme paramètre le nom du fichier et l'identifiant du votant associer.Le système de fichiers vérifie bien qu'il existe une binode ayant cet identifiant. Si aucune binode existe, l'opération échoue. Sinon, le système vérifie que l'état de linode. Le fichier est créer seulement si l'Inode est en attente de validation dans le cas contraire l'opération d'écriture échoue.Une fois le fichier créer, le processus peut ensuite ouvrir et écrire le vote de l'électeur dans le fichier.Après l'écriture et la fermeture du fichier par le processus, le fichier devient immuable (verrouiller en écriture) par le processus de vote.

4.4) Detection et suppression de doublons

Des doublons peuvent survenire dans le système de fichiers si deux serveurs de votes reçoivent à un moment donnée deux requêtes de votes ayant le même identifiant du votant.La détection et l'élimination de doublons sont effectuées par un processus (un daemon) qui s'exécutent de façon périodique sur chaque serveur de votes indépendants des processus de vote. À chaque exécution, le daemon inspecte la table des fichiers en attente de validation du serveur sur lequel il s'exécute. Si la table contient des fichiers, le daemon traité chacun de ses fichiers. Pour chaque fichier, les étapes suivantes sont effectuées:

- Étape 1: le daemon inspecte la table des fichiers valides pour vérifier s'il y a une binode ayant déjà l'identifiant du votant du fichier. Si un fichier est trouvé, le fichier en cours de traitement est invalidé (mise à jour du champ de l'état de l'Inode, suppression du fichier dans la table des fichiers en attente et ajout dans la table des fichiers invalides) sinon le daemon procède a l'étape 2

- Étape 2: le daemon communique à un des serveurs de synchronisation le numéro d'inonder du fichier, l'identifiant du votant et la date de création du fichier.

- Étape 3: le serveur de synchronisation inspecte dans la table des fichiers valides de la machine sur laquelle elle s'exécute s'il n'y a pas déjà une binode ayant l'identifiant du votant passer par le daemon. S'il existe une binode avec cet identifiant le serveur de synchronisation notifie le daemon qui par la suite va invalider le fichier. Si la table des fichiers valide est vide, le serveur de synchronisation inspecte la table des fichiers en attente. Si un fichier dans cette table contient l'identifiant du votant, le serveur valide le fichier dont la date de création est la plus ancienne et invalide l'autre fichier. Le daemon est ensuite notifié pour appliquer les mêmes changements de son coter. Si aucune binode ne contient cet identifiant, le daemon valide le fichier et notifie tous les autres serveurs de votes pour la réplication du fichier.

Le daemon repete cette opération pour chaque fichier dans la table des fichiers en attente.