1L-04

コミットベースファイルシステム Elton の提案

加藤　快†　　田胡　和哉‡

東京工科大学大学院バイオ・情報メディア研究科コンピュータサイエンス専攻†　　東京工科大学コンピュータサイエンス学部‡

1 はじめに

近年クラウドコンピューテングの普及より，クラウドサービスとオンプレミスサーバを連携させたハイブリットクラウドなどの複雑な分散処理形態が登場してきている．また，多くのクラウドサービスはネットワークトラフィックによる従量課金で利用するため，トラフィックの削減によるコストダウンの効果は大きくなる．

本稿では，このような複雑な分散処理形態に対応したファイルシステムとして，immutable なデータを管理するコミットベースファイルシステム Elton の提案を行う．

2 提案手法

2.1 Immutable なデータ管理

全てのデータを immutable データとして管理する．ファイルを更新する場合，既存のファイルを上書き保存するのではなく，バージョン番号を付与した別のファイルとして保存する．作成された全てのファイルは Read Only File となり変更されないことが保証される．これにより，キャッシュが容易となり，一度取得してしまえばローカルファイルと同様に SSD 等の高速なディスクで扱うことが可能である．

2.2 Commit 機構

ファイルの同期に対して Commit の動作を加え，Commit のタイミングでデータ共有が可能となる仕組みをとる．Commit 時にバージョン番号を発行することで，バージョンを意識したファイル管理が可能になる．

これら二つの手法を取り入れたコミットベースファイルシステムは，バージョン番号を指定することでデータが一意に決まるため，既存の分散ファイルシステムに比べて，キャッシュコヒーレンシを保つ機構が不要となる．これにより，ネットワークトラフィックの減少が見込まれる．

proposal of commit-based file system Elton
† Kai KATO
Bionics, Computer and media science, Entrepreneurship program, Tokyo University of Technology Graduate School.
‡ Kazuya TAGO
School of Computer Science, Tokyo University of Technology.

3 Elton

Elton とは，前述のコミットベースファイルシステムを実装したシステムである．Elton は以下の要素で構成されている．

3.1 Elton Master

Elton Master は，バージョン管理・発行を行うメタサーバとして動作する．図 1 に示すように，各ノードで保存されているファイルの Key と最新バージョン番号，ファイルの各バージョンがどのノードに保存されているかを管理している．この情報を元に新規バージョンの発行・ファイル取得リクエスト時にファイルの保存ノードの探索,担当ノードへの取得リクエストプロキシ，バックアップ・キャッシュのスケジューリングなどの機能がある．

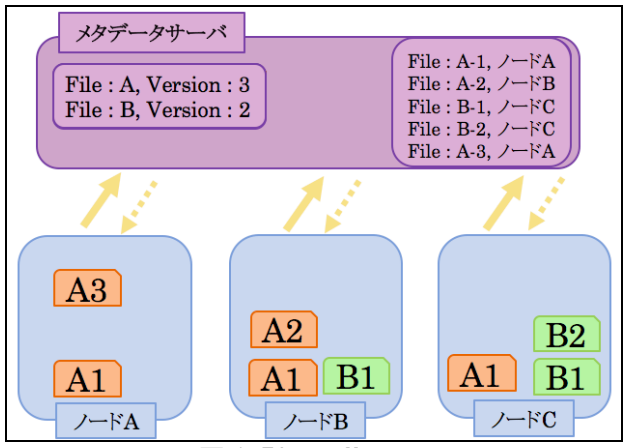

図 1 Elton Master

3.2 Elton Slave

Elton Slave は Elton でファイルを扱うための HTTP インタフェースを提供するキャッシュサーバである．Elton Master に対してファイルの作成・更新等のメタデータの送信，ファイル取得時には Elton Master へのリクエストの送信やローカルでのファイルキャッシュを行う機能がある．

3.3 Eltonfs

Eltonfs は Elton でファイルを扱うためのファイルシステムインタフェースで，FUSE ベースのファイルシステムである．特徴として，ファイルシステムのディレクトリツリーも 1 つのファイルとして Elton で管理を行う．これにより，さまざまなディレクトリツリーそのものを共有可能となり，ファイルシステムのバックアップ・複製が容易になる．

3.4 Docker Volume Plugin for Eltonfs

Docker Volume Plugin for Eltonfs は Eltonfs を使って Docker コンテナ内で利用するファイル群を共有する機能を持つ．Docker で提供されている Volume Plugin 機能を Eltonfs 用に用意して利用可能である．

4. 検証実験

4.1 ファイルシステム性能評価

Elton と NFS を比較してファイルシステムとしての評価を行った.実験に用いた環境を表 1 に示す．測定は，bonnie++[1]を利用する．Eltonfs・NFS Client 用途サーバに対して I/O 性能測定を行った．測定は各環境のメモリサイズ（プライベートクラウド: 4GB，プライベートクラウド: 1GB）に合わせて，s オプション（メモリサイズの 2 倍），r オプション（メモリサイズ)に調整する.測定結果を表 3,表 4 に示す.

表 1 検証実験に利用したプレイベートクラウド環境

項目	環境	ディストリビューション
Elton Master・NFS Server 用途サーバ	4 コア 4GB VM ＊ 1 台	CentOS 7.2.1511 (Kernel 3.10.0-327.3.1.el7.x86_64)
Eltonfs・NFS Client 用途サーバ	4 コア 4GB VM ＊ 1 台	CentOS 7.2.1511 (Kernel 3.10.0-327.3.1.el7.x86_64)

表 2 検証実験に利用したプレイベートクラウド環境

項目	環境
パブリックベンダー	Amazon Web Service EC2
Elton Master, Eltonfs・NFS Client 用途サーバ	t2.micro インスタンス＊1 台
ディストリビューション	CentOS Linux 7 x86 64 HVM EBS 20150928 01

表 3 Sequential Read の測定結果

環境	XFS	Eltonfs	NFS
プライベートクラウド	107170 K/sec	86950 K/sec	74165 K/sec
パブリッククラウド	72060 K/sec	63823 K/sec	14833 K/sec

表 4 Sequential Write の測定結果

環境	XFS	Eltonfs	NFS
プライベートクラウド	103250 K/sec	103050 K/sec	18565 K/sec
パブリッククラウド	64285 K/sec	67253 K/sec	28709 K/sec

読み込み，書き込みともに Eltonfs の方が NFS よりも性能が高いことがわかる．ベースのファイルシステムである XFS と同等の性能が出ているため高速に動作している．

4.2 ネットワークトラフィック性能評価

ネットワークトラフィックの性能評価を行う．測定のシナリオは，以下の通りである．検証環境は表 1，表 2 を利用する．

1. dd コマンドを用い，プライベートクラウド上の Eltonfs・NFS Client 用途サーバの対象ディレクトリにファイルを書き込む
2. dd コマンドを用い，パブリッククラウド上の Eltonfs・NFS Client 用途サーバの対象ディレクトリに共有されたファイルを読み込む
3. 2 を 5 分間隔で 3 回繰り返す

測定した結果, NFS では書き込みを行った際に NFS Server に対して通信があるのに対して Eltonfs では読み込みが行われるまで通信が行われていないことがわかる.それぞれ初回のトラフィックがほぼ同等であること確認できた．また，書き込み時ではなく読み込み時にファイル共有が行われていることから，不要なファイル同期によるネットワークトラフィックも抑えられていることが確認できた.

5. おわりに

本稿では，コミットベースファイルシステム Elton の提案・検証を行った.検証結果より，提案手法を用いることで既存のローカルファイルシステムに近い性能を持ったまま,トラフィックを抑えたファイルシステムの構築が可能であることが確認できた.

参考文献

[1] OSDN Corporation: “Bonnie++を使ったファイルシステム性能のベンチマーク | OSDN Magazine” https://mag.osdn.jp/08/08/04/0134256(2019/01/11).