Vol.2015-OS-134 No.15
2015/8/5

SMB Multichannelの分散ファイルシステムへの適用による並列I/O実現の検討

鈴木 光一朗[1,2,a)]　大山 恵弘[1,3,b)]

概要：爆発的に増大する非構造化データに柔軟に対応できることで注目されるスケールアウト型アーキテクチャを持つ Network Attached Storage（NAS）は，単一ノードでの I/O にボトルネックが存在する．本稿では，サーバ／クライアント間の複数のネットワーク接続を使用することで帯域幅の集約を実現する SMB Multichannel の仕組みを分散ファイルシステムに適用し，相互接続性の高い SMB プロトコルによるノードをまたいだ並列 I/O を実現する方法について検討する．

Investigation of implementing Parallel I/O on Distributed File Systems using SMB Multichannel

Koichiro Suzuki[1,2,a)]　Yoshihiro Oyama[1,3,b)]

Abstract: Much attention has been paid to Network Attached Storage (NAS) of the scale-out architecture because it can flexibly handle explosively increasing unstructured data. However, it contains a bottleneck in I/O operations handled by a single server node. In this paper, we describe a method of extending distributed file systems with the SMB Multichannel mechanism, which aggregates bandwidth by using multiple network connections between a server and a client. The method achieves parallel I/O operations across different server nodes using the highly interoperable SMB protocol.

1. はじめに

国際的なデジタルデータの量は急増を続けている [1]．その中でも，オフィス文書や音声，画像，映像といった非構造化データが占める割合は 8 割〜9 割に上るともいわれており，それらのデータはファイルとして格納されることが多い．そのため，ファイルシステム上に格納される非構造化データは増加の一途を辿っており，ストレージの容量や性能に対する要求は大きくなっている．

近年このような状況を受けて，ファイルベースのデータ増大に柔軟に対応可能なスケールアウト型アーキテクチャを備えた Network Attached Storage [2]（NAS）が注目されている．ここでスケールアウトとは，システムを構成するマシンの数を増やすことで全体性能の向上を図る概念である．また，スケールアウトと対比して，マシンの CPU やメモリ，ストレージといったハードウェアを増強することで性能の向上を図るスケールアップという概念も存在する．企業における従来の NAS 利用では，ユーザ数や使用容量の増加に対しては，マシンリソースを増強したり，より高い性能を持つ NAS への置き換えを実施したりする等，スケールアップのアプローチにて対応されてきた．しかし，スケールアップにより対応できる範囲を超えた際は，新たな NAS が追加で導入され，各 NAS へアクセスするユーザやファイルの配置等を別々に管理しなければならなくなる等，NAS のサイロ化による運用管理の煩雑さが大きな課題となっていた．スケールアウト型 NAS では，複数のマシンをクラスタリングし，単一ネームスペースのストレージプールを実現可能なスケールアウト型アーキテクチャを持

[1] 電気通信大学
The University of Electro-Communications
[2] IzumoBASE 株式会社
IzumoBASE, INC.
[3] 独立行政法人科学技術振興機構, 戦略的創造研究推進事業
JST, CREST
[a)] szk@ol.inf.uec.ac.jp
[b)] oyama@inf.uec.ac.jp

つ分散ファイルシステムが搭載される場合が多く，クラスタを構成するマシン（ノード）を追加することで，ストレージ全体で利用可能な総容量およびスループットや IOPS 等の合計性能の拡張が容易に実現できる．

一般に，NAS へアクセスする際には，相互接続性の高いプロトコルである SMB [3,4] や NFS [5] が利用される．これらのプロトコルを用いて NAS へアクセスする際，図 1 に示すように，通常はクライアントとサーバが 1 対 1 でプロトコルセッションを構成するため，クライアント-サーバ間のネットワーク性能やサーバの処理性能，ディスク性能等様々なボトルネックが存在する．スケールアウト型 NAS を利用する場合も一般的な NAS と同様に 1 対 1 でセッションが構成される．そのため，単一のプロトコルセッションのパフォーマンスにおける単一ノードの性能による頭打ちは依然として存在する．スケールアウト型 NAS のパフォーマンススケーラビリティは，プロトコルセッションが複数構成された状況下ではじめて発揮される．そしてそれは，DNS による振り分けや動的な IP テイクオーバー等をノード負荷等に応じてインテリジェントに行う等の負荷分散を実現することで最大化される．

NAS の企業利用におけるデファクトスタンダードなストレージアクセスプロトコルとして知られる SMB プロトコルの一機能である SMB Multichannel では，クライアント-サーバ間の複数のネットワーク経路を使用して SMB プロトコルの I/O を実現可能である．これにより，単一ノードのネットワーク性能による頭打ちを軽減できるものの，ディスク性能等のボトルネックは依然として解決されない．

本稿では，SMB Multichannel の仕組みを利用して，単一の SMB セッションの I/O 処理を分散ファイルシステムを構成する複数のノードへ展開する方法について検討する．その方法（図 2）により，単一ノードへのアクセスが引き起こしていたパフォーマンスボトルネックが低減され，クラスタワイドな並列 I/O が実現可能となる．

以降の章の構成を以下に述べる．はじめに 2 章では，本研究にて利用する SMB Multichannel の概要およびプロトコル上で行われるやりとりの分析結果について記述する．次に，3 章では，目標とするシステムの概要と実現方法を示し，4 章にて実装計画を述べる．その後，5 章で関連研究について述べ，最後に 6 章にて本稿をまとめる．

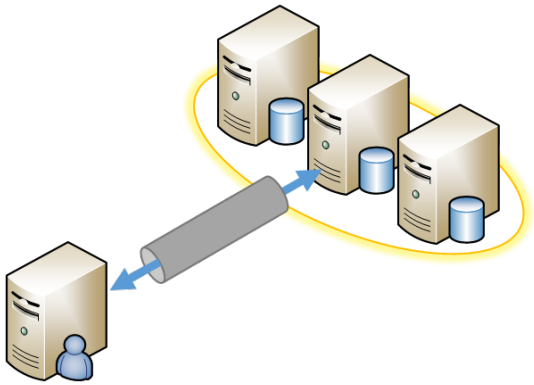

図 1　スケールアウト型 NAS のセッションボトルネック
Fig. 1　A session bottleneck of a scale-out NAS

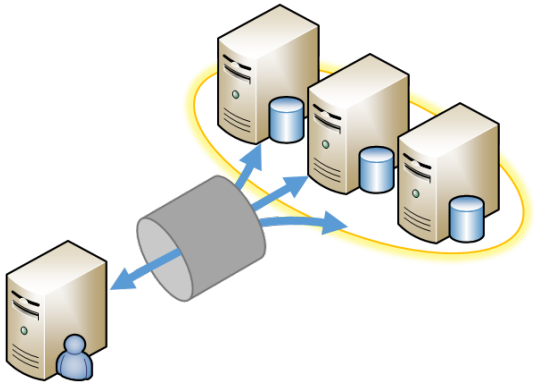

図 2　単一セッションの複数ノードへの展開イメージ
Fig. 2　An image of spreading a single session on multiple nodes

2. SMB Multichannel

2.1 機能概要

Server Message Block（SMB）は，Microsoft Windows に実装されたファイル共有プロトコルである．SMB プロトコルには大別して SMB [6] と SMB2 [7] が存在し，それぞれの仕様は公開されている．SMB プロトコルは，オープンソースの SMB サーバ実装である Samba [8] をはじめ，オープンソース，プロプライエタリの別に関わらず様々な OS やストレージシステムに実装されている．SMB プロトコルは Windows のリリースと併せてバージョンアップされることが多く，現在リリースされている最新のバージョンである SMB 3.02 は，Windows 8.1 および Windows Server 2012 R2 に搭載されている．現在公開されている SMB 2.0 以降のバージョンは，SMB2 プロトコル上で実現されており，以下では特に断りがない限り SMB2 プロトコルを SMB プロトコルとして表現することとする．

Windows 8 および Windows Server 2012 にて搭載された SMB 3.0 では，SMB プロトコルでのアクセスにおけるネットワークボトルネックを大きく低減させる，SMB Multichannel の機能追加が行われた [9]．これにより，SMB クライアント-サーバ間に複数のネットワーク経路がある場合に，複数経路のネットワーク帯域幅を集約し，単一の SMB セッションのスループット向上とネットワークのフォールトトレランスを同時に実現することが可能となった．以前も，NIC チーミングにてネットワークレベルのリンクアグリゲーション等の技術を用いることで，ネットワーク帯域幅の集約とフォールトトレランスを実現することは可能であったが，リンクアグリゲーションでは各 NIC の送受信データは単一のアドレスに対しては単一の NIC でのみ扱われるため，単一セッションの転送スループットが

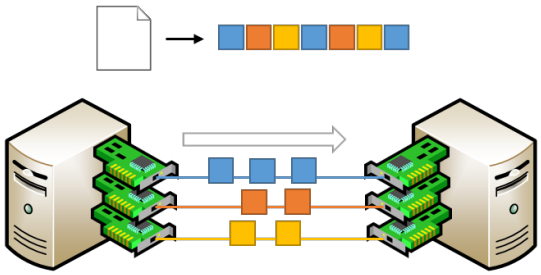

図 3　SMB Multichannel 転送イメージ
Fig. 3　An image of SMB Multichannel transfers

単経路のスループットより大きくなることはない．

SMB セッションで SMB Multichannel が有効化された場合，SMB クライアントはファイルを読み書きする際に発行する Read リクエストおよび Write リクエストを，アプリケーションレベルで複数の経路に振り分ける．このときそれぞれのコマンドよって送受信されるデータの大きさは，SMB プロトコルのブロックサイズに依存する．この機能によって，図 3 に示すように，分割された R/W リクエストおよびレスポンスは複数経路を用いて並列に伝送され，単経路のスループットを超える I/O スループットを実現することが可能となる．

2.2　Multichannel セッションの構築

SMB Multichannel を分散ファイルシステムに適用するために，SMB Multichannel が有効な SMB セッション（Multichannel セッション）がどのように構築されるか理解する必要がある．ここでは，図 4 に示すような，ネットワークインタフェースを 2 つ持つ Windows Server 2012 R2 を搭載したマシンを 2 台用意した．そして，各マシンに 2 つずつ搭載されたネットワークインタフェースを用いて，2 つのネットワーク経路で接続し，SMB サーバおよび SMB クライアントとして構成した．この環境を用い，SMB クライアントから SMB サーバへアクセスした際の Multichannel セッションが構築されるまでの SMB プロトコル上のやりとりを，ネットワークプロトコル分析ソフトウェアである Wireshark [10] によりキャプチャした．Wireshark により観測された SMB プロトコルでのやりとりのうち，Multichannel セッションを構成する上で有意なものについて図 5 に示す．サーバは 2 つのネットワークインタフェースでそれぞれチャネルを開き接続を待ち受けており，本分析でクライアントははじめにチャネル 1 の IP に対して接続を実施する．また，図 5 中の二重水平線は，(3) と (4) の間に実施される複数のやりとりの省略を表す．

SMB クライアントが SMB サーバにアクセスした際，ポート 445 番への TCP セッションを構築する．その後，はじめに，図 5 中 (1) にて，SMB プロトコルにおける 1 コマンドである，Negotiate Protocol コマンドによって，クライアント-サーバ間で利用する SMB バージョンの機能レベルのすり合わせが行われる．SMB クライントから送られる Negotiate Protocol リクエストには，クライントが対応する SMB バージョンを表す値のリストと，SMB Multichannel をはじめとしたオプション機能を表すフラグが含まれる．それを受けた SMB サーバは，自身が対応する SMB バージョンのリストおよびオプション機能のリストと照らし合わせ，クライアントおよびサーバが対応するバージョンのうち，最も機能レベルが高いバージョンと，自身の対応リストでフィルタした利用可能なオプション機能を表すフラグを返す．そのため，クライアント-サーバ間で SMB バージョンが異なる場合は，表 1 に示した対応表に基づき，ネゴシエーション時により低い SMB バージョンに合わせた結果となる．以後，SMB クライアントは，ネゴシエーションの結果として取得した，互いに解釈可能な SMB バージョンとオプション機能に合致する適切なリクエストのみを発行するようになる．

続いて，図 5 中 (2) にて，Session Setup コマンドによる SMB セッションの構築が行われる．Session Setup では，SMB サーバへの接続に求められる認証方法を決定するためのやりとりが行われる．その後，認証に必要なユーザ名やパスワード等の情報が送信される．サーバ側で接続が許可されると SMB セッションが確立され，それ以降 SMB クライアントは当該 SMB セッションを用いて SMB サーバへのアクセスを継続する．

SMB クライアントは，Negotiate Protocol によって SMB Multichannel が有効な SMB サーバだと判定した場合，SMB セッションが確立された後に，図 5 中 (3) にて，Ioctl コマンドによって SMB サーバが持つネットワークインタフェースの情報を要求する．これに対して SMB サーバは，自身の持つネットワークインタフェースにつき，IP アドレスやリンク速度，および Receive-Side Scaling（RSS）や Remote Direct Memory Access（RDMA）といった機能に対応しているか等の情報を返す．SMB クライアントは，受け取った SMB サーバのネットワークインタフェース情報と自身の持つネットワークインタフェース情報から利用可能なチャネルを見出す．その後，SMB セッションがバインドされていない利用可能なチャネルへ TCP セッションを構築した後，図 5 中 (4) および (5) に示すように，Negotiate Protocol コマンドと Session Setup コマンドによって，構築済みの SMB セッションを対応付ける．

以上の流れにより，SMB クライアント-サーバ間で SMB Multichannel が有効な SMB セッションが構成され，ファイルの読み込みおよび書き込みの際に複数のチャネルを用いて並列に送受信することが可能となる．

3.　分散ファイルシステムへの適用

3.1　概要

本研究で構築目標とするシステムのイメージを図 6 に示

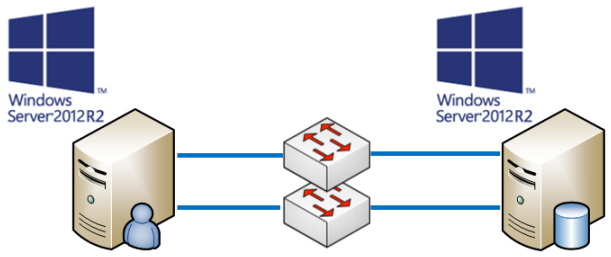

図 4　Multichannel セッション構築分析環境
Fig. 4　An environment for analyzing establishment of multi-channel sessions

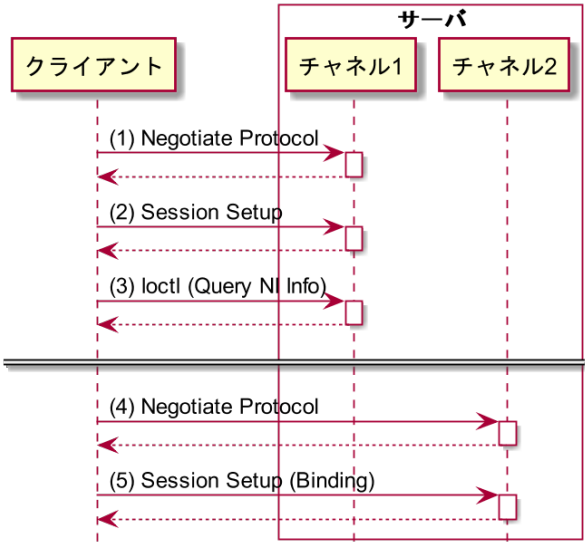

図 5　Multichannel セッション構築時のやりとり
Fig. 5　Interaction for establishing a multichannel session

表 1　SMB プロトコルの接続バージョンネゴシエーション
Table 1　Connection version negotiation of the SMB protocol

サーバ＼クライアント	3.02	3.0	2.1	2.0	1.0
SMB 3.02	3.02	3.0	2.1	2.0	1.0
SMB 3.0	3.0	3.0	2.1	2.0	1.0
SMB 2.1	2.1	2.1	2.1	2.0	1.0
SMB 2.0	2.0	2.0	2.0	2.0	1.0
SMB 1.0	1.0	1.0	1.0	1.0	1.0

す．本研究の提案を適用する対象として，複数の I/O サーバを持ち，単一のネームスペースを構成可能な分散ファイルシステムを想定している．各 I/O サーバは，1 つ以上のネットワークインタフェースを持ち，1 つ以上の IP を付与されたマシンである必要がある．また，クライアントは，SMB Multichannel の機能が利用可能な Windows 8 および Windows Server 2012 以降の Windows OS が搭載された，複数のネットワークインタフェースを持つマシンを想定している．さらに，Multichannel セッションを開始する Windows OS の SMB クライアントの仕様により，各クライアントマシンから，複数のセグメントで 1 台以上の I/O サーバにアクセス可能なネットワーク構成を構築する必要がある．同一セグメント内の複数チャネルを用いた Multichannel セッションを構築できる SMB クライアントが存在すればこの限りではない．

3.2　クラスタワイドな Multichannel セッションの構築

本研究の提案である，SMB Multichannel の分散ファイルシステムへの適用によるクラスタワイドな並列 I/O を実現するためには，分散ファイルシステムを構成するノードをまたいだ複数のチャネルを束ねた SMB セッションを構成する必要がある．図 7 にノードをまたいだ Multichannel セッションを構成するためのアイディアを示す．ここでは，簡単のためノードを 2 台としているが，同手順を拡張することでより高い並列度にも対応可能である．図 7 中で，ノード 1 およびノード 2 は，SMB サーバ用のチャネルをそれぞれ 1 つずつ開いており，クライアントはそれらのチャネルを束ねたセッションを構成することとなる．

利用開始時，SMB クライアントは分散ファイルシステムを構成するノードのうち，いずれかのノード（図 7 中ノード 1）にアクセスする．この際，図 7 中 (1) では，図 5 と同様のネゴシエーションが行われる．その後，図 7 中 (2) にて作成される SMB セッションをその後のバインディングのために，(a) のようにノード 2 に共有する．そして，図 7 中 (3) の Ioctl コマンドによるネットワークインタフェース情報の取得リクエスト受信時には，ノード 1 が持つネットワークインタフェースの情報に加えて，実際にはアクセス中の SMB サーバには存在しない，分散ファイルシステムを構成する他のノード（図 7 中ノード 2）のネットワークインタフェース情報を (b) にて取得し，それをマージして返答を行う．これにより，図 7 中 (4) および (5) にて，SMB クライアントはセッション作成時とは異なるノードのチャネルへセッションのバインドを行い，結果としてノードをまたいだ複数のチャネルを用いた Multichannel セッションを構成することとなる．以上のように Multichannel セッションの構築を拡張することにより，以後 SMB クライアントが Multichannel セッションを用いてファイルの読み書きを実施する際には，ノードをまたいだ並列 I/O が実施される．

4.　実装計画

本研究では，前述した SMB Multichannel の分散ファイルシステムへの適用による並列 I/O 実現のために，将来の拡張性を考慮してスクラッチで SMB サーバを実装中である．直近の実装計画は以下のとおりである．

(1) SMB サーバのミニマム実装
(2) 単一ノードでの SMB Multichannel 対応
(3) バックエンド接続部のインタフェース化
(4) 分散ファイルシステムバックエンドへの対応
(5) 複数ノードでの SMB Multichannel 対応

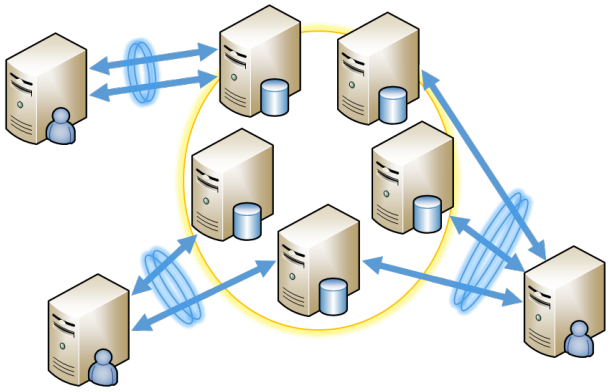

図 6　構築目標システムイメージ
Fig. 6　An image of the system to implement

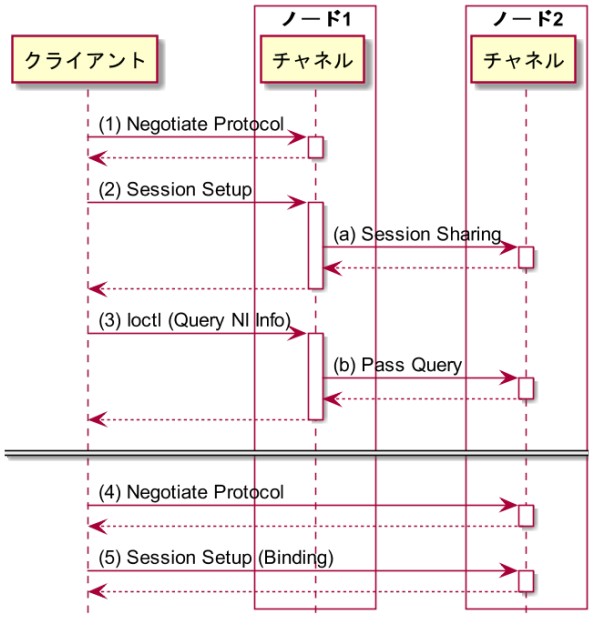

図 7　ノードをまたいだ Multichannel セッション構築
Fig. 7　Establishment of a multichannel session across server nodes

はじめに，ローカルファイルシステムをバックエンドとして動作する SMB サーバのミニマム実装を行う．ここで言うミニマム実装とは，実際の利用においては重要であるが提案手法を検証する上では必要のない認証機構やファイルのアクセス許可等についての実装を省いたものである．その後，SMB クライアントと単一の SMB サーバ間の複数のネットワーク経路を利用した SMB Multichannel による I/O 処理を実装する．ローカルファイルシステムでの SMB サーバの動作を実現できた後に，バックエンド接続部のインタフェース化によるバックエンドの切り替えを実現する．そして，分散ファイルシステムをバックエンドとして利用できるようにインタフェースを拡張し，複数ノードを束ねた SMB Multichannel によるアクセスへの対応を行う．ここで，複数のノードへ単一のファイルに対する並列 I/O が行われるため，書き込みを受けた際はノード間でファイルデータの結合を行うか，読み込み時に他のノードの持つデータをリレーするなど，適用対象とする分散ファイルシステムのアーキテクチャに依存した拡張が必要となる場合もある．

5. 関連技術

GPFS [11] や Lustre File System [12]，GlusterFS [13] 等の分散ファイルシステムでは，並列 I/O 可能な独自仕様のプロトコル実装を持つ．独自仕様のプロトコルでは，それぞれの分散ファイルシステムに合わせた設計および実装が可能で，I/O パフォーマンスを引き出しやすい一方，相互接続性が低く，標準的なプロトコル上での並列 I/O を提案する本研究とは異なる．

パラレル NFS [14]（pNFS）は，NFSv4.1 の一仕様として策定された規格であり，複数台のデータサーバにストライピングされたファイルに直接並列にアクセスすることが可能である．pNFS アーキテクチャは，各ファイルの所在を示すメタデータを保持するメタデータサーバ（MDS）とファイルデータを格納するデータサーバによって構成される．pNFS クライアントは MDS から受け取ったファイルレイアウトから，データサーバへアクセスする際のストレージアクセスプロトコルをはじめ，ファイルの各ストライプの格納場所に関する情報を得る．現時点で，NFS プロトコルでアクセスするファイルレイアウト，FC や iSCSI [15] といったブロックベースのプロトコルでアクセスするブロックレイアウト [16]，オブジェクトベースのプロトコル（OSD [17]）でアクセスするオブジェクトレイアウト [18] の 3 種類のレイアウトが存在する．pNFS のアーキテクチャを用いれば，各分散ファイルシステムの仕組みに合ったレイアウトを用いて効率的な並列 I/O を実現できる．しかし，Windows 向けのクライアントが存在しない [19] 等，相互接続性が低く，Microsoft が主導し Windows に実装され，より高い相互接続性を誇る SMB プロトコルを利用する本研究とは異なる．

Cluster Trivial Database [20]（CTDB）は，Samba のための一時的なデータ保存に用いられる Trivial Database（TDB）のクラスタ実装であり，GPFS や GlusterFS，Lustre 等の分散ファイルシステムと組み合わせて All-Active な SMB サーバクラスタを実現する．All-Active で SMB サーバクラスタを公開する際には，SMB セッションの状態やファイルのロック情報等の管理情報をノード間で共有する処理のオーバーヘッドが，SMB アクセスのパフォーマンスに影響するという問題がある．CTDB では，管理情報の共有において，分散ファイルシステムを管理情報の実体を同期させるために用いるのではなく，各ノードがローカルに保持する TDB の一貫性担保のためのセマフォとしてのみ利用し，管理情報の実体はノード間でネットワーク越しに IPC メッセージングを行うことによりパフォーマン

スの低下を抑えることを目的としている．しかし，CTDB における SMB サーバとしては各ノードで Samba が実行されており，単一の SMB セッションは単一のノードに対して構成され，複数ノードを束ねた I/O を実現することはできず本研究とは異なる．

6. おわりに

本稿では，スケールアウト型 NAS にアクセスする際の単一セッションのボトルネックについて述べ，相互接続性の高いプロトコルとして知られる SMB プロトコルの一機能である SMB Multichannel を分散ファイルシステムに適用するための手法を提案した．今後は，提案内容である Multichannel セッションの構築を実現するために，SMB サーバの実装と分散ファイルシステムとの結合を行い，クラスタワイドな並列 I/O 実行時のパフォーマンスを検証する．

謝辞 本研究の一部は，科学技術振興機構戦略的創造研究推進事業（JST CREST）の研究課題「ポストペタスケールデータインテンシブサイエンスのためのシステムソフトウェア」の支援を受けている．

参考文献

[1] 総務省 ：平成 26 年版情報通信白書，日本政府 (2014).
[2] Gibson, G. A. and Van Meter, R.: Network attached storage architecture, *Communications of the ACM*, Vol. 43, No. 11, pp. 37–45 (2000).
[3] Microsoft: Server Message Block overview, `https://technet.microsoft.com/en-us/library/hh831795.aspx`.
[4] Hertel, C. R.: *Implementing CIFS: The Common Internet File System*, Prentice Hall Professional (2004).
[5] Nowicki, B.: NFS: Network File System Protocol specification, RFC 1094 (Informational) (1989).
[6] Microsoft: [MS-SMB]: Server Message Block (SMB) Protocol, `https://msdn.microsoft.com/en-us/library/cc246231.aspx`.
[7] Microsoft: [MS-SMB2]: Server Message Block (SMB) Protocol Versions 2 and 3, `https://msdn.microsoft.com/en-us/library/cc246482.aspx`.
[8] The Samba Team: Samba, `https://www.samba.org/`.
[9] Kruse, D. and George, M.: SMB 2.2: Bigger. Faster. Scalier, SNIA, 2011 Storage Developer Conference, (online), available from ⟨http://www.snia.org/events/storage-developer2011⟩.
[10] Wireshark Foundation: Wireshark, `https://www.wireshark.org`.
[11] Schmuck, F. and Haskin, R.: GPFS: A Shared-Disk File System for Large Computing Clusters, *Proceedings of the 1st USENIX Conference on File and Storage Technologies (FAST '02)* (2002).
[12] Schwan, P.: Lustre: Building a File System for 1,000-node Clusters, *Proceedings of the 2003 Linux Symposium* (2003).
[13] Gluster community: GlusterFS, `http://www.gluster.org/`.
[14] Shepler, S., Eisler, M. and Noveck, D.: Network File System (NFS) Version 4 Minor Version 1 Protocol, RFC 5661 (Proposed Standard) (2010).
[15] Satran, J., Meth, K., Sapuntzakis, C., Chadalapaka, M. and Zeidner, E.: Internet Small Computer Systems Interface (iSCSI), RFC 3720 (Proposed Standard) (2004).
[16] Black, D., Fridella, S. and Glasgow, J.: Parallel NFS (pNFS) Block/Volume Layout, RFC 5663 (Proposed Standard) (2010).
[17] Weber, R. O.: SCSI Object-Based Storage Device Commands -2 (OSD-2), *T10 Working Draft* (2009).
[18] Halevy, B., Welch, B. and Zelenka, J.: Object-Based Parallel NFS (pNFS) Operations, RFC 5664 (Proposed Standard) (2010).
[19] McDonald, A.: NFS 4.2 Q&A, `http://sniaesfblog.org/?p=424`.
[20] The Samba Team: Samba CTDB, `https://ctdb.samba.org/`.