［研究論文］

サーバ管理システムにおける高可用性ファイルサーバシステムの開発
Development of a High-availability File Server System for a Server Management System

北村 光芳†, 田中 龍馬‡
Mitsuyoshi KITAMURA, Ryoma TANAKA

† 東京工芸大学大学院 工学研究科
‡ 株式会社日立情報通信エンジニアリング
† Graduate School of Engineering, Tokyo Polytechnic University
‡ Hitachi Information & Telecommunication Engineering, Ltd.

要旨

サーバ管理システムにおいて，管理サーバの故障問題を解決するために，専用の管理サーバを必要としないサーバ管理システムが報告されている．このシステムではクライアントにサービスを提供する各サーバが管理サーバの機能（管理対象サーバの監視及び故障サーバの復旧処理）を有するため，管理ファイルを共有する必要があり，専用のファイルサーバを導入している．しかし，そのファイルサーバが故障した場合，サーバ管理が継続できなくなる．そこで，クライアントにサービスを提供するサーバをクラスタ構成し，簡易化されたデータの冗長化方式を採用した高可用性ファイルサーバシステムを提案する．サーバ故障時などの対処を自動化し，運用管理者の負担を軽減する構成としている．

Abstract

In server management, server management systems not requiring a dedicated management server have been reported to solve the problem of management server failures. In this system, because each server (target server) providing services to clients has the functions (monitoring the target server and recovering failed server) of a management server, management files must be shared. Therefore, a dedicated file server is introduced. However, if that file server fails, server management cannot continue. Therefore, we propose a high-availability file server system with a simplified system configuration in which servers providing services to clients are clustered. The system is designed to automate dealing with server failures. Thus, the burden on operations managers is reduced.

1. はじめに

近年，インターネットサービスは我々の日常生活に欠かせないものとなっている．そのサービスを提供するためにはサーバの役割が重要となる[1]．そのため，サーバの故障検出や復旧処理など，高可用性を考慮したシステムが必要となる．そこで，データセンタにおける蓄積データとサービス配信において，災害などによる故障の影響を低減するデータバックアップとサービスマイグレーションを統合した緊急保護方式[2]や Kubernetes と分散ストレージを使用することによる高可用性サーバシステムの実現手法[3]が報告されている．

高可用性を実現する方法として，一般的に High Availability（HA）クラスタシステム[4]が採用されているが，冗長化対象のサーバ群が同一サービスとなる構成が基本のため，様々なサービスを提供する高可用性サーバシステムを想定すると管理の複雑化が懸念される．また，Load Balancer（LB）による負荷分散クラスタシステム[4]も多く採用されているが，サーバ故障により冗長化を維持できなくなる問題を有している．そこで，単純化したサーバ管理プログラムを使用し，1 台の実サーバで複数台の実サーバの機能をバックアップ可能な動的バックアップサーバシステム[5]が提案されている．しかし，専用の管理サーバを使用しているため，そのサーバが故障した場合，サーバ管理が継続できなくなる．

そのため，専用の管理サーバを必要としないサーバ管理システムとして P2P 方式サーバ管理システム[6]やリング型動的バックアップサーバシステム（Ring-type Dynamic Backup Server System: RDBSS）[7]が報告されている．ここで，本論文で使用する RDBSS（[3.5. 導入対象とする RDBSS の概要]を参照）は，Microsoft 社の Redirected Drive Buffer Subsystem（RDBSS と略されることが多い）とは異なる．P2P 方式サーバ管理システムや RDBSS ではクライアントにサービスを提供するサーバが管理サーバの機能（管理対象サーバの監視及び故障サーバの復旧処理）を有するため，各サーバが管理ファイルを共有す

［研究論文］
2025 年 4 月 17 日受付, 2025 年 7 月 11 日改訂, 2025 年 8 月 20 日受理

る必要がある．一般的な IT システムにおいても複数台のサーバから構成されているため，サーバ間のデータ共有が重要となる．そのため，データ共有に関するシステムの実装や研究が行われている[8][9][10][11][12][13][14][15][16].

しかしながら，サーバ管理システムにおけるデータ共有システムでは低コストや高可用性の実現及びシステムトラブル時における運用管理者の負担軽減を考慮し，システム構成の複雑化を防ぐ必要がある．

そこで，本研究の目的として，RDBSS を導入対象とし，クライアントにサービスを提供するサーバをクラスタ構成し，簡易化されたデータの冗長化方式を採用した高可用性ファイルサーバシステムの開発を行う．提案システムではファイル共有を提供するサーバが故障した場合においてもデータの冗長化構成を構築しており，単一障害点（Single Point Of Failure: SPOF）を考慮した構造としている．その詳細な構成を示すとともに，共有エリアにおけるデータアクセスの監視に Linux の Inotify を採用し，効率的なアクセス（作成，編集，削除）の分析方法を示す．また，提案システムにおいてサーバが故障した場合の復旧方式を詳細に述べ，提案方式を採用した実験システムを構築し，サーバ故障時の復旧処理に要する時間を示す．

本論文の残りの部分は以下のように構成する．2．では実装システム及び先行研究について述べ，3．では提案システムの詳細構成及び導入対象とする RDBSS の概要を示す．4．ではサーバ故障時における復旧処理方式について述べ，5．では実験システムの構成や仕様を示し，動作実験により提案システムの評価を行い，その結果と考察を述べる．最後に結論を 6．で述べる．

2. 実装システム及び関連研究

実装システムにおけるデータ共有に関し，専用のファイルサーバを使用する方式やストレージエリアネットワークを構成し，専用のストレージを設置する方式が一般的に採用されている．しかし，これらの方式はデータを共有するサーバ台数に応じて仕様を考慮する必要があり，コスト面が懸念される．また，これらの方式では一般的にデータの冗長化を実現するために，ファイルサーバまたはストレージの共有エリアを Linux の rsync コマンドなどを使用してバックアップサーバと同期を行う[8]. しかし, rsync では共有エリアに保存されている全データのステータスを確認後，必要データの複製を行うため，複数台のバックアップサーバと rsync による接続を行う場合，ファイルサーバの負荷が懸念される．Software Defined Storage（SDS）[9]による高可用性ファイルサーバシステムの実装も多く採用されている．SDS ではクライアントにサービスを提供する各サーバをクラスタ化し，その空きストレージ容量を使用して論理的なファイルサーバを構築しており，クラスタ内のサーバ間で冗長化構成を構築してデータの高可用性を実現している．また，Linux の Distributed Replicated Block Device（DRBD）により複数のサーバにネットワークを通じてブロックデバイス単位でミラーリングを行うシステムも実装されている．しかしながら，DRBD によるクラスタシステムの構築は SDS と同様で，設定や運用の難易度が高く，システム構成が複雑なため，専門の SI ベンダーによる構築や保守が必要となり，導入及び運用コストが懸念される．

サーバ管理システムにおいて，共有対象のデータの差分のみをフルメッシュ接続により転送し，データ共有を可能とする同期的編集方式（Synchronous Editing Method: SEM）[10]が報告されている．この方式は効率の良い転送方式を実現しているが，フルメッシュ接続を採用していることからサーバ台数が増加した場合の通信性能に懸念がある．更に，この方式は文献[6][7]のサーバ管理システムに特化しているため，汎用的な使用はできない．

データ共有に関する安全性の検討がされており，不正なファイル操作を防ぐためにソフトウエア設定の範囲で解決する報告[11]や IoT アプリケーション開発支援ツールである SINETStream を用いて，暗号化処理や研究グループ内のメンバ間での柔軟なアクセス制御など，データを安全に管理するための機能が提案されている[12]．サーバ管理システムにおいて，安全なデータ共有は非常に重要な要素となる．しかし，サーバ管理システムでは管理者権限でのファイル操作やデータ共有が行われるため，特別な安全機能の追加は不要となる．

データ共有の基本となるのがストレージシステムであり，この障害に関する検討も進められている．ドライブ障害やサーバ障害がシステムの信頼性に与える影響を分析するための信頼性モデルおよび評価式の提案[13]やサーバ管理システム用の高可用性ファイルサーバシステムに関する報告[14]もされているが，実システムにおける検証がされていない．

広域な範囲におけるデータ共有に関する研究も進められている．広域データ共有用ネットワークを開発するために，伝送装置による遅延要因を分析し，遠隔サーバ間でバス信号を超低遅延に伝送するバス接続方式が提案されている[15]．広域なデータ共有を低遅延で実現することは非常に重要である．しか

し，光バス装置及びその伝送路に要するコストの高額化が懸念される．また，広域環境でのデータ共有を実現する Gfarm とクラウドストレージを組み合わせた階層型ストレージシステム[16]が報告されている．このシステムは参照頻度が少ないデータをクラウドストレージに移行することにより効率的なデータ共有を実現している．この共有方式は非常に効率的であるが，管理ファイルなどの機密情報を保存する場合の配慮が懸念される．

サーバ管理の高可用性を確保するために，データ共有の確実性が求められるサーバ管理システムとして，Pacemaker と Corosync の協調動作による HA クラスタシステムや提案システムの導入対象としている RDBSS が挙げられる．これらのシステムは専用の管理サーバを必要としない．また，この HA クラスタシステムは冗長化対象のサーバリソースを詳細に調査でき，RDBSS に比べ高速な障害対策を実現している．しかし，クラスタに所属するサーバが故障した場合，冗長化対象のサーバ数が減少してしまう．RDBSS はサーバ故障が発生した場合，バックアップサーバを作成することで，冗長化対象におけるサーバ数の減少を防いでいる．また，Pacemaker と Corosync の設定は非常に柔軟である反面，設定が複雑であり，運用面においても広範な知識が要求される．RDBSS は 1 つの管理プログラムが 1 台のサーバを管理するシステムの集合体となっており，シンプルな構成となっている．

高可用性ファイルサーバシステムでは，「制御プログラム及びシステム構成が単純であること」，「システム故障に対して高速な復旧処理が可能なこと」，「コストを考慮していること」，「汎用性や拡張性を有すること」が重要であるが，上述した実装システムや関連研究では，このすべてを兼ね備えたシステムは存在しない．

3. 提案ファイルサーバシステムの構成及び導入対象システムの概要

3.1. 提案システムの設計概要

実サーバでシステムを構成する場合，各サーバにおいて SPOF を防ぐため，ストレージ障害対策として RAID 構成及び電源の 2 重化が必要となる．本提案システムでは RDBSS における管理用データの冗長化を行う．RDBSS ではシステム起動時にそのデータの参照を行い，サーバ故障等の復旧処理時にそのデータの参照や編集が行われる．この編集時に，提案システムにおけるデータの冗長化処理が発生する．また，一般的にサーバに採用するストレージはオーバースペックとなる傾向がある．そのため，クライアントにサービスを提供する各サーバの空きストレージ容量を有効的に活用する．一般的なサーバシステムにおいて，各サーバは共有すべきデータをローカルではなく，ネットワーク経由でファイルサーバなどに保存する．そのため，空きストレージ容量は数 GB 以上となることが予想される．提案システムでは今後のサービス拡張などを考慮し，各サーバにおける空き容量の 50%で最小となる値を使用可能容量とする．本論文では RDBSS の管理用のデータを冗長化するため，各サーバにおいて 10MB 程度の容量を確保する．

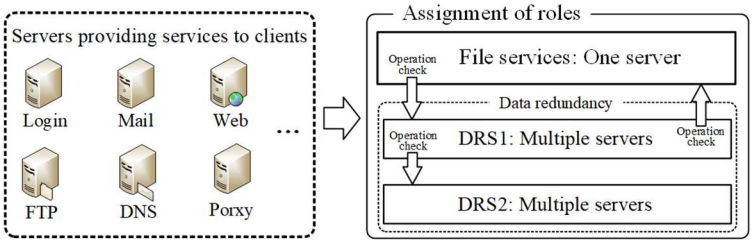

図 1　提案システムの基本構成

構成としては，図 1 に示すように各サーバをクラスタ化し，ファイルサービスを提供可能なシステムを構築する．システムにおける役割として，ファイル共有用サーバを 1 台とし，データの冗長化用サーバ（Data Redundancy Server：DRS）を複数台とする．データの冗長化構成としては，ファイル共有用サーバのデータを冗長化用サーバ 1（DRS1）に転送する．また，データの冗長化をより強固にするため，DRS1 のデータを更に冗長化用サーバ 2（DRS2）に転送する．システム動作時には，ファイル共有用サーバと DRS1 は相互に，DRS1 は DRS2 のネットワーク接続の確認を行う．提案システムは，ファイル共有用のデータにおける高可用性の実現方法に関し，システム構造等を単純化している．そのため，ネットワーク接続に関する監視のみを行い，接続先のサーバが故障，過負荷や異常動作などによりネットワーク接続ができなくなった場合に対処を行う．運用管理者の負担軽減を考慮し，その対処は自動化する．

3.2. 提案システムの初期設定

図 2 に提案システムにおける初期設定を示す．提案システムでは，クライアントにサービスを提供する各サーバをファイル共有用サーバ，DRS1，DRS2 や予備サーバ（サーバ故障時に提案システムとして動作）のいずれかに割り当てる．Server list ファイルには稼働しているサーバのホスト名，IP アドレス及び状態（提案システムとして動作するサーバは“0”，予備サーバは“1”，故障サーバは“2”）が記録されている．システムの初期設定としては，導入するシステムの過去の稼働状況やサーバストレージの空き容量などを参考にし，図に示すように FS（ファイル共有用サーバ）を 1 台，DRS1 と DRS2 は一組で構成し，“DRS1_1，DRS2_1”を 1 組目として複数組とする．ファイル共有用サーバに必要となる情報は File sharing server ファイル，データの冗長化用サーバに必要となる情報は Data redundancy server ファイルに記録する．File sharing server ファイルにはファイル共有用サーバ名，提案システム用の共有エリアのディレクトリ名，ファイル共有用サーバ内で RDBSS が使用する共有エリアのシンボリックリンク名，DRS1 とのマウントポイント，ファイル共有クライアントからの接続用仮想 IP アドレスが記録されている．Data redundancy server ファイルには DRS1 のホスト名，DRS2 のホスト名，提案システム用のマウントポイント，RDBSS 用のマウントポイントが記録されている．Data redundancy server ファイルは，行毎に DRS1 と DRS2 の組が構成され，全体のサーバ数に応じてこの組を増やすことでシステムの拡張が可能となる．

図では，File sharing server ファイルに，ファイル共有用サーバとして Mail サーバ，提案システム用の Shared area に“/nfs”，RDBSS 用の Link point に“/system”，DRS1 とのマウントポイントに“/mp/S1”や“/mp/S2”など，仮想 IP アドレスとして“192.168.1.254”が設定されている．また，Data redundancy server ファイルには，DRS1 と DRS2 として各サーバが設定されており，提案システム用の Mount point1 に“/nfs”，RDBSS 用の Mount point2 に“/system”が設定されている．

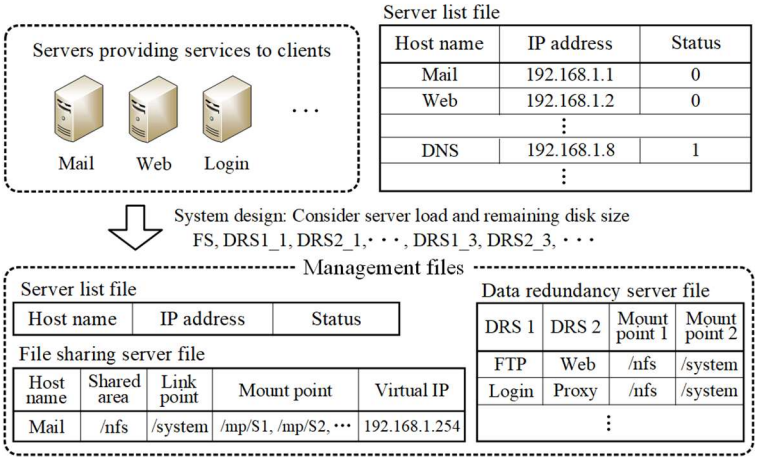

図 2　提案システムの初期設定

3.3. 提案システムの詳細構成

7 台のサーバ（Mail，FTP，Web，Login，Proxy，DNS と Database サーバ）を例とし，図 3 に提案システムの詳細構成を示す．図 2 で示した File sharing server ファイル及び Data redundancy server ファイルを参照し，図 3(a)に示すように，各サーバをクラスタ化し，提案システムを構成する．図では，Mail がファイル共有，FTP と Login が DRS1，Web と Proxy が DRS2 の役割を担当し，DNS はキャッシュされていないドメイン情報の要求やキャッシュ保持の有効期限切れなどの場合にキャッシュミスが発生し，ディスク I/O が実行される．また，Database サーバは，要求データの種類のばらつきが多いため，キャッシュミスが発生しやすい．このようにキャッシュミスを多発する可能性があるサーバは予備サーバとする．予備サーバは，提案システムとして動作しているサーバが故障した場合，提案システムとして動作し，そのクラスタ構成に組み込まれる．

提案システムは共有エリア（/nfs）に保存されているデータを DRS1 及び DRS2 に複製し，データの冗長化を行う．Mail のマウントポイント（/mp/S1，/mp/S2）と DRS1 との接続に関し，Mail がデータを 2 重に保存することを防ぐため，Mail がそのマウントポイントと DRS1 の Mount point1（/nfs）を SSHFS で接続する．DRS1 と DRS2 はともにデータを保存する必要がある．ここで，データを同期する方法と

して Linux の DRBD や rsync がある．DRBD はブロックデバイスでの同期を行うことから全サーバのストレージで同様のブロックデバイスを用意する必要があり，また，設定や運用の難易度が高い．rsync でのファイル共有は，ディレクトリ単位で設定でき，コマンドを実行するだけでファイル共有が可能となる．そのため，DRS1 は rsync で DRS2 の Mount point1 にデータ転送を行う．

本システムでは Mail の共有エリアへのデータアクセスに関し，Linux の Inotify を使用して監視を行う．共有エリアにアクセス（ファイル作成，削除，編集など）した場合，Inotify により検出され，ファイル作成や編集の場合には，対象データをマウントポイントに複製，削除の場合には，マウントポイントの対象データを削除する．また，ディレクトリを作成した場合は，マウントポイント内に同様のディレクトリを作成する．この図では，共有エリアのデータは 4 台（FTP，Web，Login，Proxy）のサーバに複製される．図 3(b)において，赤太線のサーバは提案システムが稼働していることを示す．各サーバはファイル共有クライアントとして Mail に設定されている仮想 IP アドレスに対して，Mount point2（/system）と Mail の共有エリア（/nfs）を SSHFS で接続する．

上述したように，提案システムはクラスタ構造が単純な構成となっているため，SDS や DRBD で構築されたシステムとは異なり，設定や運用の難易度が低いため，導入及び管理コストを低減できる．

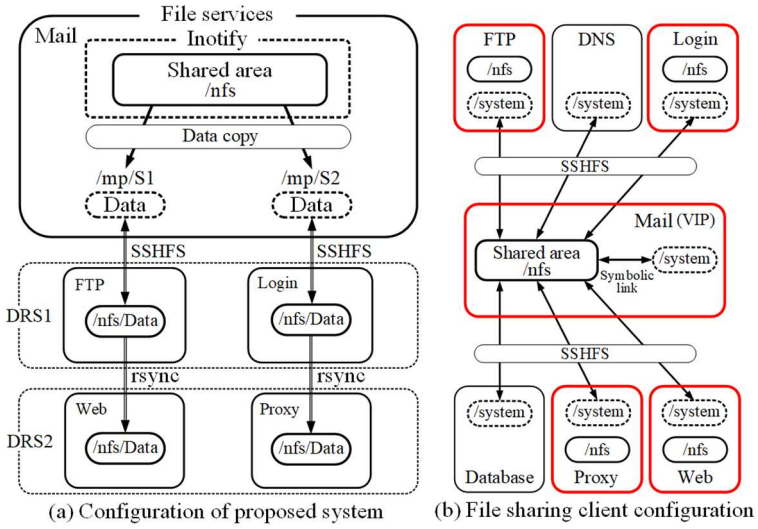

図 3　提案システムの詳細構成

3.4. 共有エリア内のデータアクセスの分析

上述したように，共有エリアの監視には Linux の Inotify を使用する．図 4 に共有エリア内におけるデータアクセスの効率的な分析方法を示す．

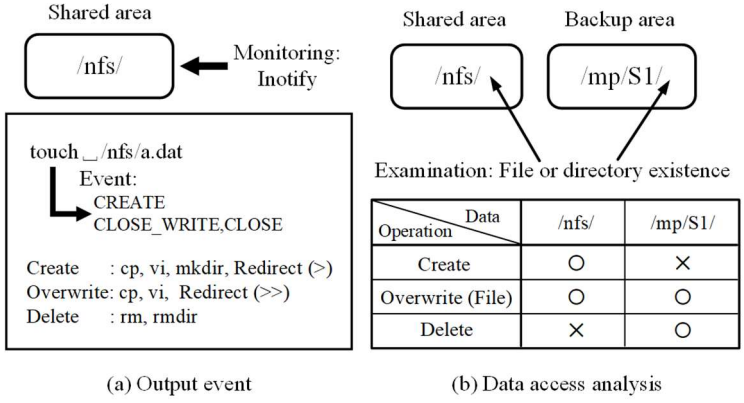

図 4　共有エリア内のデータアクセスの分析方法

共有エリアにファイルアクセスを行った場合，Inotify により出力されるイベントを図 4(a)に示す．図では Linux の touch コマンドで共有エリア(/nfs)に a.dat ファイルを作成している．これに対して Inotify

では“CREATE”と“CLOSE_WRITE_CLOSE”のイベントを出力する．この出力されるイベントはコマンド操作によって異なる．共有エリアにアクセスするためのコマンドは図に示すように代表的なものでも複数あり，すべての操作に応じたイベントに対する分析方法は非常に複雑となる．

そこで，図 4(b)に効率的な共有エリアへのアクセス分析方法を示す．共有エリアと同様のデータ等を保存するバックアップエリアを設定し，Inotify からイベントが発行された場合，共有エリアとバックアップエリアを調査する方式を採用する．ここで，バックアップエリアはファイル共有用サーバ内のマウントポイントとする．この方式では，データが共有エリアに存在し，バックアップエリアに存在しない場合は新規作成，データが両エリアに存在し，それらに差分がある場合は上書き，データがバックアップエリアのみに存在する場合は削除と分析する．このように，共有エリアで行われたコマンド操作に対する分析が単純化できる．

3.5. 導入対象とする RDBSS の概要

図 5 に提案システムの導入対象である RDBSS[7]の概要を示す．ここでは，RDBSS を LB により構成される仮想サーバシステムに導入する．仮想サーバである Mail，Login，FTP，Database，Proxy と Web サーバ（管理サーバかつ管理対象サーバ）がシステムを構成している．RS1～RS3 は実サーバ，VIP は仮想 IP アドレスを示す．ここでの RDBSS はサーバ同士が監視（ネットワーク及びサービス提供状態）を行う機能を廃止し，LB にサーバ監視機能を集約する．そのため，各仮想サーバにおける管理プログラムは，LB からの故障情報が送られるまで待機状態となる．

図に示すように，LB が Login の故障を検出した場合（図 5 の上段部分），その情報を各仮想サーバに送信し，各種管理ファイルを更新する（図 5 の下段部分）．各仮想サーバは管理構成を示す Operation ファイルを確認し，Login の管理サーバである Mail がそのサーバ機能の復旧処理を行う．Mail は，負荷が一番低い RS2 に Login と同様のサービスを提供可能なバックアップサーバ Loginb を作成制御し，VIP2 を設定する．これにより故障した Login のサーバ機能は復旧される．その後，Mail が Login を復旧困難と判断した場合，Loginb を管理構成に追加し，リング型管理構成を再構築する．

管理ファイルとして，仮想サーバの情報は Group ファイルに記録され，それをもとに管理構成を示す Operation ファイルが作成される．Login が故障した場合，Login の情報は Group ファイルから Separetion ファイルに移動され，Operation ファイルの再構築が行われる．Separetion ファイルには故障サーバの情報が記録され，Problem ファイルには Login の復旧処理の状況が記録される．Login が復旧困難と判断された場合，Group ファイルに Loginb の情報が追加され，Operation ファイルの再構築が行われる．各仮想サーバが管理ファイルを共有するために図 3(b)に示した“/system”を使用する．

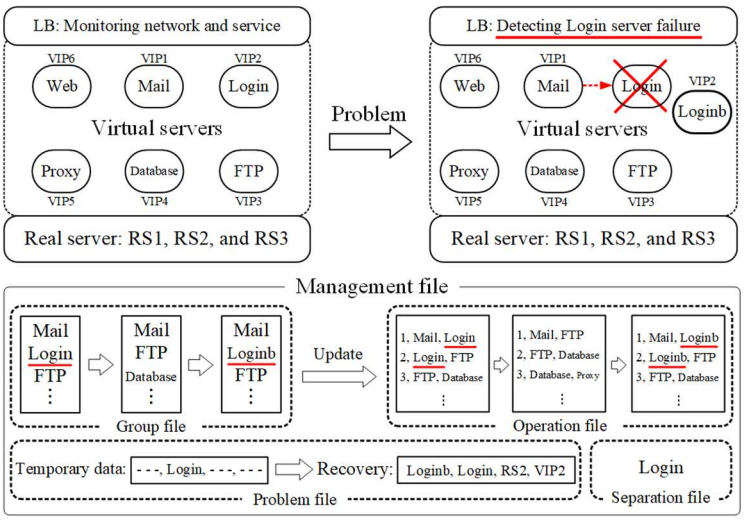

図 5　仮想サーバシステムに導入している RDBSS の概要

4. サーバ故障時における復旧処理

4.1. データの冗長化用サーバ故障時における復旧処理

図 6 に DRS2 が故障した場合の復旧処理について示す．DRS1 は DRS2 と rsync で接続している．その接続が切断された場合，DRS1 は DRS2 を故障と判断し，復旧処理を行う．図に示すように FTP（DRS1）

は Web（DRS2）の故障を検出した場合，図 2 で示した Server list ファイルの Status 欄が“1”（予備サーバ）となっているサーバを検索し，最初に検出された DNS を DRS2 として rsync で接続する．その後，FTP は Server list ファイルの Status 欄を DNS は“0”（提案システムとして動作），Web は“2”（故障サーバ）に変更し，Data redundancy server ファイルにおける DRS2 欄の Web を DNS に変更する．システムの初期設定やサーバ故障の影響により予備サーバが存在しない場合（Server list ファイルの Status 欄に“1”が存在しない）には，DRS2 なしの状態で動作を行い，Data redundancy server ファイルにおける DRS2 欄の Web を“trouble”に変更する．

図 7 に DRS1 が故障した場合の復旧処理について示す．ファイル共有用サーバと DRS1 は SSHFS で接続されている．その接続が継続できない場合，ファイル共有用サーバは DRS1 を故障と判断し，復旧処理を行う．図に示すように Mail（ファイル共有用サーバ）は FTP（DRS1）の故障を検出した場合，Data redundancy server ファイルで，FTP の DRS2 である Web を DRS1 とし，SSHFS で接続する．また，Server list ファイルの Status 欄が“1”（予備サーバ）となっているサーバを検索し，最初に検出された DNS を DRS2 として Web と DNS を rsync で接続させる．その後，Mail は Server list ファイルの Status 欄を DNS は“0”（提案システムとして動作），FTP は“2”（故障サーバ）に変更し，Data redundancy server ファイルにおける DRS2 欄の Web を DNS，DRS1 欄の FTP を Web に変更する．

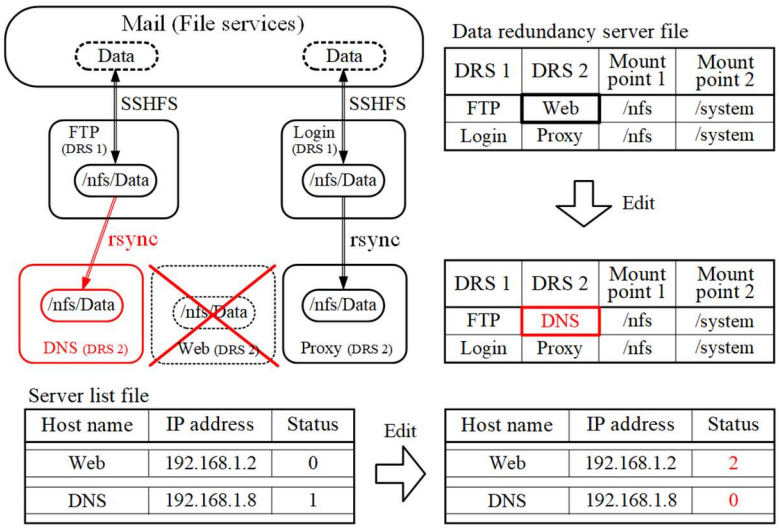

図 6　DRS2 の故障時における復旧処理

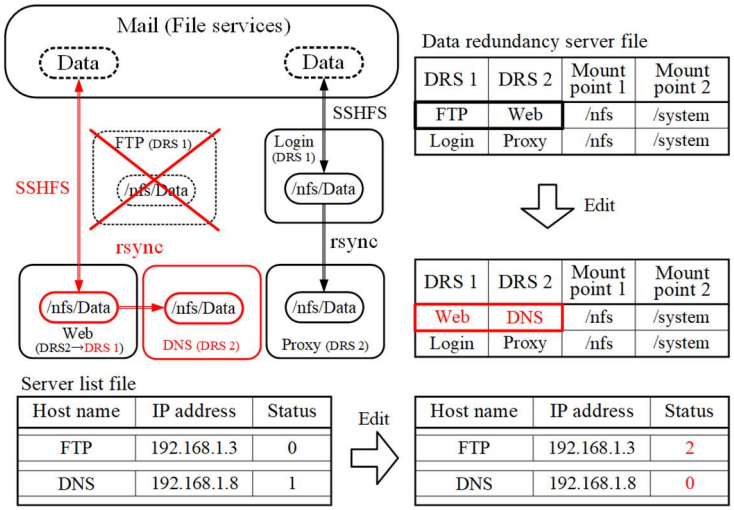

図 7　DRS1 の故障時における復旧処理

システムの初期設定やサーバ故障の影響により予備サーバが存在しない場合（Server list ファイルの Status 欄に“1”が存在しない）には，DRS2 なしの状態で動作を行い，Data redundancy server ファイルにおける DRS2 欄の Web を“trouble”，DRS1 欄の FTP を Web に変更する．

4.2. ファイル共有用サーバ故障時における復旧処理

図 8 にファイル共有用サーバが故障した場合の復旧処理について示す. ファイル共有用サーバと DRS1 は SSHFS で接続されている．その接続が切断された場合，DRS1 はファイル共有用サーバが故障したと判断する．図に示すように Mail（ファイル共有用サーバ）が故障した場合，DRS1 である FTP と Login がその故障を検出する．Data redundancy server ファイルにおいて，DRS1 の先頭である FTP が復旧処理を開始する．FTP はファイル共有用サーバとして動作するために，File sharing server ファイルを編集すると同時にマウントポイント（/mp/S1，/mp/S2）を作成し，Mail に設定されていた仮想 IP アドレスを設定する．FTP は Data redundancy server ファイルで，FTP の DRS2 である Web を DRS1 とし，SSHFS で接続する．また，Server list ファイルの Status 欄が“1”（予備サーバ）となっているサーバを検索し，最初に検出された DNS を DRS2 として Web と DNS を rsync で接続させる．その後，FTP は Server list ファイルの Status 欄を DNS は“0”（提案システムとして動作），Mail は“2”（故障サーバ）に変更し，Data redundancy server ファイルにおける DRS2 欄の Web を DNS，DRS1 欄の FTP を Web に変更する．システムの初期設定やサーバ故障の影響により予備サーバが存在しない場合（Server list ファイルの Status 欄に“1”が存在しない）には，DRS2 なしの状態で動作を行い，Data redundancy server ファイルにおける DRS2 欄の Web を“trouble”，DRS1 欄の FTP を Web に変更する．

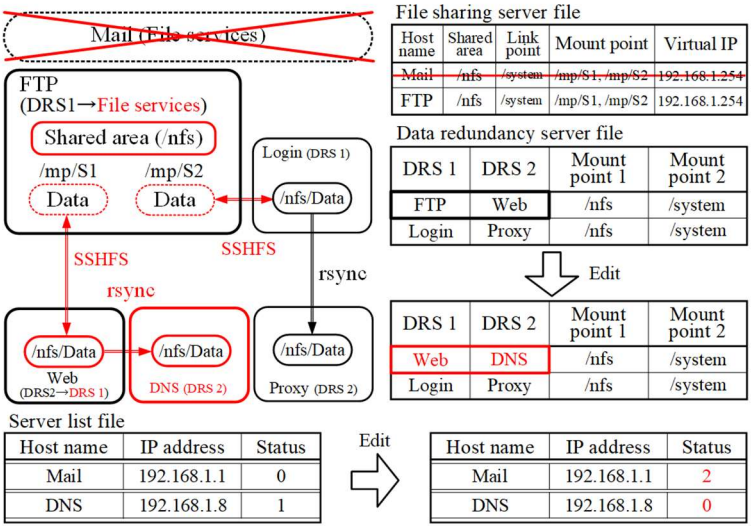

図 8　ファイル共有用サーバの故障時における復旧処理

4.3. 故障サーバの調査プログラム

本システムでは，故障サーバに対して調査プログラムが実行され，復旧した場合には，状況に応じて DRS1，DRS2 や予備サーバに設定する．図 9 に DRS2 用のサーバが故障した場合を示す．故障サーバが復旧した場合，Data redundancy server ファイルを調査し，“trouble”が記録されている欄（DRS1 または DRS2）に応じた役目を実行する. その状況に応じた内容に Data redundancy server ファイル及び Server list ファイルを編集する．また，Data redundancy server ファイルに“trouble”が記録されていない場合には予備サーバとし，Server list ファイルを編集する．

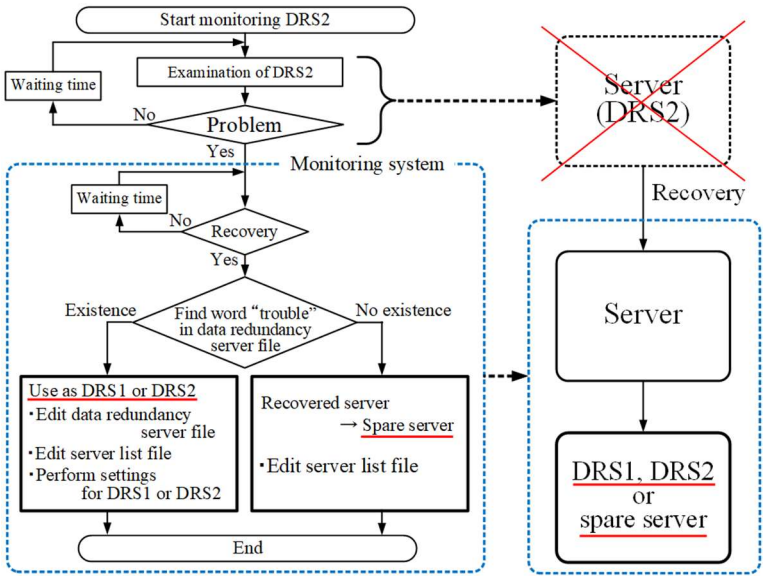

図 9　故障サーバの調査アルゴリズム

5.　動作実験

本実験は，仮想サーバシステム上で，提案システムの動作状態または RDBSS との協調動作の検証を行う．実サーバシステムでの検証と異なり，ハードウェア障害や電源障害などによるサーバ故障時における提案システムの調査は仮想サーバをシャットダウンすることで再現する．また，サーバに負荷を再現する実験を行った場合，仮想サーバでは実サーバのリソースを共有している他の仮想サーバに影響が出る可能性がある．更に，ネットワーク環境も実システムとは異なるため，サーバ間のネットワーク遅延による影響などを調査できない．以上から本実験では，各仮想サーバに対して負荷を与えず，サーバ故障時における提案システムの挙動を調査する範囲とし，実測する復旧時間は参考値として示す．

5.1. 実験システムとその仕様

図 10 に実験システムの構成を示す．3 台の実サーバ（RS1，RS2，RS3）が 9 台の仮想サーバを作成している．各仮想サーバはクライアントに Mail，Web，Login のサービスを提供し，それぞれのサーバグループは冗長化構成を構築している．Mail サーバグループには，M1，M2，M3，Web サーバグループには，W1，W2，W3，Login サーバグループには，L1，L2，L3 が所属し，各サーバでは，RDBSS と提案システムが動作している．ファイルサーバは，それぞれのサーバグループにおけるサービスの冗長化用に使用する．クライアントにサービスを提供するネットワークにはスイッチング HUB1，ファイルサーバへのアクセス用にはスイッチング HUB2 を使用する．

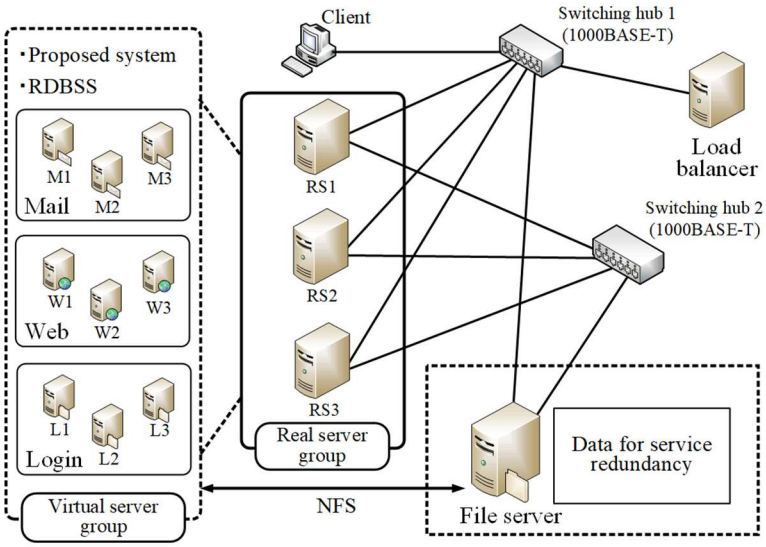

図 10　実験システム

表 1 に実験システムの仕様，特性及び設定値を示す．実サーバ（RS1，RS2，RS3）はサーバ仮想化ソフトとして，Linux で標準サポートされている Kernel-based Virtual Machine（KVM）を採用する．また，各仮想サーバでは Mail には postfix，Web には Apache，Login には sshd を採用する．LB にはソフトウエア方式を採用し，HAProxy をインストールする．ここで，仮想サーバの通常起動時間は約 16 秒で，サスペンド状態からの起動は 1.33 秒と非常に高速となる．そのため，サーバ機能の復旧を行うために使用するバックアップサーバは常時，サスペンド状態で待機させる．

表 1　実験システムの仕様，特性及び設定値

Specifications	Real server (RS1, RS2, RS3)	File server
CPU	Intel Core i7-3770 3.40 GHz (TB: 3.90 GHz)	Intel Core i7-2600 3.40 GHz (TB: 3.80 GHz)
Memory	8,192 MB	
Storage	SSD (SATA 3)	
Virtualization software	KVM 1.5.3	nfsd
OS	Rocky Linux 9.5	

Specifications	Load balancer
CPU	Intel Core i5-4430 3.00 GHz
Memory	4,096 MB
System software	HAProxy 2.4
OS	Rocky Linux 9.5

Specifications	Virtual server / Backup server
CPU	1 Core
Memory	2,048 MB
System software	postfix, Apache, sshd, sshfs
OS	Rocky Linux 9.5

Characteristics	Virtual server / Backup server
Start time (s)	16.08
Restart time (s)	18.19
Time to activate suspended server (s)	1.33

Setting parameters	
Network examination time (Disconnection) (s)	2.00

実験システムにおける提案システムの構成を図 11 に示す．仮想サーバが 9 台作成されていることから，ファイル共有用サーバ及び DRS1 と DRS2 で 7 台使用し，2 台は予備とする．ここで，赤字で示すサーバ（L1，W3，M2）は実験において故障を再現する対象とし，提案システムのみが動作している場合及び RDBSS と提案システムが動作している場合における復旧時間の実測を行う．

また，サーバ故障時におけるクライアントからサーバに対する接続アクセスの影響を調査するため，クライアントから LB 経由で Mail，Web，Login のサービスを提供しているサーバに対して 1 秒間隔でアクセスを行う．

更に，4.3 で示した故障サーバの調査プログラム及び予備サーバが存在しない場合における動作実験は，予備サーバである M3 を DRS1，W2 を DRS2 と初期設定（予備サーバが存在しない環境を作成）し，提案システムのみが動作している環境で故障対象の仮想サーバを再起動する．

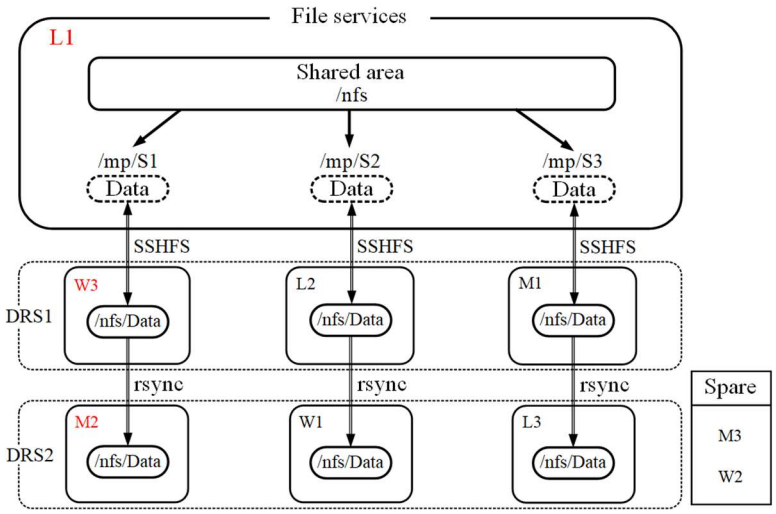

図 11　実験システムにおける提案システムの構成

5.2. 実験結果及び考察

表 2 に実験結果として，提案システムのみが動作している場合及び RDBSS と提案システムが動作している場合の 2 種類を示す．実験では故障対象毎に 10 回行い，平均復旧時間及び変動係数を示す．提案システムのみが動作している場合の復旧時間は，提案システムが故障を検出した時点から，そのシステム構成が復旧するまでとし，RDBSS と提案システムが動作している場合の復旧時間は，故障を検出した時点から，提案システム及び故障サーバの機能が復旧するまでとする．

提案システムのみが動作している場合において，DRS2 である M2 の故障を再現した場合，システム構成の復旧時間は 0.561 秒，DRS1 である W3 の故障を再現した場合には 0.468 秒となった．また，ファイル共有用サーバである L1 の故障を再現した場合，復旧時間は 1.732 秒となった．RDBSS と提案システムが動作している場合において，DRS2 である M2 の故障を再現した場合，そのサーバ機能の復旧時間は 3.435 秒，DRS1 である W3 の故障を再現した場合には 3.855 秒となった．また，ファイル共有用サーバである L1 の故障を再現した場合，復旧時間は 6.745 秒となった．いずれの場合も変動係数は低い値を示しており，復旧処理は安定していることが分かる．

RDBSS と提案システムが動作した場合において，DRS1 と DRS2 の故障は，管理ファイルの取得が可能なため，RDBSS の管理動作に影響しない．また，ファイル共有用サーバの故障は，各サーバ（ファイル共有クライアント）が管理ファイルを取得できるまでの待ち時間（提案システムの復旧時間及びファイル共有用サーバにマウントするまでの時間）が発生する．しかし，この待ち時間は RDBSS の管理動作に影響するが，各サーバは冗長化構成のため，クライアントからサーバへの接続アクセスには影響しない．

表 2　実験結果

(a) Trouble reproduction experiment in proposed system

Target	Part	Recovery time (s)	Coefficient of variation
M2	DRS2	0.561	0.041
W3	DRS1	0.468	0.041
L1	File services	1.732	0.018

(b) Trouble reproduction experiment in RDBSS (Shared area: Proposed system)

Target	Part	Recovery time (s)	Coefficient of variation
M2	DRS2	3.435	0.013
W3	DRS1	3.855	0.024
L1	File services	6.745	0.007

Recovery time of file services: RDBSS (3.521 s) + wait time (3.224 s)
(Wait time: Recovery time in proposed system (1.732 s) + mount time (1.492 s))

故障サーバの調査プログラム及び予備サーバが存在しない場合の動作実験では，DRS2 である M2 を再起動した場合, DRS2 を設定できないため Data redundancy server ファイルの W3（DRS1）に対する DRS2 欄に“trouble”が記述される. その後, M2 が復旧するため, 調査プログラムによって Data redundancy server ファイルの DRS2 欄で “trouble” となっている部分を “M2” に変更し，DRS1 である W3 と M2 が rsync 接続された．この一連の動作状態を確認した．M2 の故障を検出してから，システムに復帰するまでの時間は 19.987 秒（変動係数：0.016）となった．また，DRS1 である W3 を再起動した場合，DRS2 である M2 を DRS1 に変更する．DRS2 を設定できないため Data redundancy server ファイルの M2（DRS1）に対する DRS2 欄に “trouble” が記述される．その後，W3 が復旧するため，調査プログラムによって Data redundancy server ファイルの DRS2 欄で “trouble” となっている部分を “W3” に変更し，DRS1 となった M2 と W3 が rsync 接続された．この一連の動作状態を確認した．W3 の故障を検出してから，システムに復帰するまでの時間は 20.005 秒（変動係数：0.014）となった．更に，ファイル共有用サーバである L1 を再起動した場合，DRS1 である W3 をファイル共有用サーバ，DRS2 である M2 を DRS1 に変更する．DRS2 を設定できないため Data redundancy server ファイルの M2（DRS1）に対する DRS2 欄に “trouble” が記述される．その後，L1 が復旧するため，調査プログラムによって Data redundancy server

ファイルの DRS2 欄で“trouble”となっている部分を“L1”に変更し，DRS1 となった M2 と L1 が rsync 接続された．この一連の動作状態を確認した．L1 の故障を検出してから，システムに復帰するまでの時間は 20.477 秒（変動係数：0.009）となった．この実験では仮想サーバの再起動を行うことから，いずれの時間も表 1 の再起動時間 18.19 秒に近い値となっている．

実験結果から，提案システムによるファイル共有の状態やサーバ故障時における復旧動作が確認できた．ファイル共有用サーバが故障した場合，RDBSS の管理動作に影響するため，これを最小限に抑える検討が必要となる．また，実サーバシステムでの環境や各サーバへ負荷を与えた場合の実験がなされていない．今後は実システム上で稼働実験ができる環境を構築し，より現実的な動作環境（各サーバへの負荷やネットワーク転送負荷を再現）において稼働状況のデータ収集及びその分析を行う必要がある．

6. おわりに

サーバ管理システム用に，クライアントにサービスを提供するサーバをクラスタ構成し，簡易化されたデータの冗長化方式を採用した高可用性ファイルサーバシステムを提案した．このシステムでは，各サーバをファイル共有用サーバ，データの冗長化用サーバ（DRS1，DRS2）や予備サーバのいずれかに割り当てる構成とした．また，運用管理者の負担を軽減するため，それぞれの役割におけるサーバが故障した場合の対処を自動化しており，実験によりその動作確認を行った．この研究により，サーバ間でのファイル共有が安全に実行可能となり，管理ファイルの共有が確実となるため，サーバ管理システムの安定動作が実現できる．これにより，各種インターネットサービスを支える基盤システムの信頼性が向上する．その結果として，我々の生活に必要なインターネットサービスの安定した提供が期待できる．

謝辞

本論文の執筆にあたり，数々の有益で大変貴重なご指摘やアドバイスをいただきました査読者の方々に感謝申し上げます．

参考文献

[1] 北村光芳, 田中龍馬, 東本崇仁, “サーバ構築学習の理解度を向上する自動調査システムと自己評価にもとづくフィードバックシステムの開発,” 電子情報通信学会論文誌 B, Vol.J107-B, No.9, 2024, pp.461-470.
[2] Ma, L. and Yang, B., “Time Constrained Emergency Data Backup and Service Migration in Cloud Data Centers,” IEEE Conference Proc., Vol.2022, No.NaNA, 2022, pp.92-95.
[3] Khatami, A.A., Purwanto, Y., and Ruriawan, M.F., “High Availability Storage Server with Kubernetes,” IEEE Conference Proc., Vol.2020, No.ICITSI, 2022, pp.74-78.
[4] 奥村恭弘, 西村徹, 横関大子郎, 由良俊介, “高可用性技術－－24 時間 365 日休まず運転するために,” NTT 技術ジャーナル, https://journal.ntt.co.jp/backnumber2/0602/files/jn200602079.pdf, 2023.6.14 参照.
[5] Kitamura, M., “Configuring a Low-Cost, Power-Saving Multiple Server Backup System: Experimental Results,” IEICE Trans. Commun., Vol.E95-B, No.1, 2012, pp.189-197.
[6] Kitamura, M., Udagawa, Y., Nakagome, H., and Shimizu, Y., “Development of a Server Management System Incorporating a Peer-to-Peer Method for Constructing a High-availability Server System,” JISSJ, Vol. 13, No. 2, 2018, pp.14-40.
[7] Kitamura, M., Takeshita, T., and Okazaki, T., “Development of Ring-type Dynamic Backup Server System not Requiring Dedicated Management Server,” JISSJ, Vol. 18, No. 1, 2022, pp. 22-42.
[8] Rsync, https://rsync.samba.org/, 2013.05.20 参照.
[9] 斎藤彰宏, 南聖二, 岡野一恵, 倉前裕成, “Software Defined Storage 分散ストレージにおけるデータ可用性/保護の実装と実機検証に基づく提言,” 情報処理学会ディジタルプラティクス, Vol. 9, No. 2, 2018, pp.573-591.
[10] Kitamura, M. and Tani, K., “Development of File Management System for a Peer-to-Peer Method Server Management System”, JISSJ, Vol.16, No.1, 2020, pp.1-16.
[11] 中村隆喜, “共用計算機・占有計算機間での安全なデータ共有方式の検討,” 情報処理学会研究報告, Vol.2024, No.HPC-195, 2024, pp.1-8.
[12] 北川直哉, 竹房あつ子, 合田憲人, “IoT アプリケーションシステムのための安全なユーザ間データ共有機構の開発,” 情報処理学会研究報告, Vol.2022-SPT-46, No.23, 2022, pp. 1-6.
[13] 山本貴大, 千葉武尊, 大平良徳, 斎藤秀雄, “分散ストレージシステムにおける障害パタンを考慮した信頼性の分析,” 情報処理学会論文誌 Comp. Sys., Vol.15, No.2, 2022, pp. 1-14.
[14] 北村光芳, 新城幸也, 田中龍馬, “サーバ管理システム用高可用性 ファイルサーバシステムの開発,”

情報システム学会 第 20 回全国大会・研究発表大会, P008, 2024.
[15]松田俊哉, 関剛志, 宮村崇, “広域データ共有に向けた超低遅延遠隔バス接続方式の検討,” 電子情報通信学会研究技法, Vol.122, No.398, 2023, pp.96-100.
[16]北澤昴大, 建部修見, “Gfarm とクラウドストレージによる階層型ストレージシステムの研究,” 情報処理学会研究報告, Vol.2019-HPC-170, No.25, 2019, pp. 1-7.

著者略歴

北村 光芳（きたむら　みつよし）
1984 年東京工芸大学工学部電子工学科卒．同年シャープ（株）入社．1990 年東京工芸大学助手，同大学講師を経て 2022 年より同大学准教授，現在に至る．博士（工学）．省電力，高可用性サーバシステムの構築，運用および計算機シミュレーションに関する研究に従事．

田中 龍馬（たなか　りょうま）
2025 年東京工芸大学大学院工学研究科博士前期課程卒．同年（株）日立情報通信エンジニアリング入社．現在に至る．ストレージ製品の開発に従事．