2004－DSM－34　(3)
2004／7／30

メールシステムの信頼性に関する一考察

中筋　香里 †　　泉　裕 ‡　　齋藤　彰一 ‡†　　塚田　晃司 ‡†
上原　哲太郎 ‡‡　　國枝　義敏 ‡‡†

† 和歌山大学大学院　システム工学研究科　　‡ 和歌山大学　システム情報学センター
‡† 和歌山大学　システム工学部　‡‡ 京都大学　　‡‡† 立命館大学

内容梗概 近年，インターネットの爆発的な普及により，電子メールは新たなコミュニケーションツールとして定着している．電子メールにリアルタイム性は求められないものの，確実にメールを届ける信頼性は非常に重要である．そのため，安定したサービスを提供し続けるための仕組みが不可欠である．サーバのメンテナンスやトラブル等により，やむを得ずシステムを停止する際，DNS(Domain Name System) の仕組みを利用し，メールサーバを冗長化することでサービスの可用性を向上させることが可能だが，処理中のメールデータを引き継ぐことは不可能である．本システムでは，処理中のデータをも引継ぎ，メールサーバの完全な冗長化手法を提案する．
キーワード　：　冗長化，メールサーバ

Stateful E-mail Redundant System

Kaori NAKASUJI †　　Yutaka IZUMI ‡　　Shoichi SAITO ‡†　　Kouji TSUKADA ‡†
Tetsutaro UEHARA ‡‡　　Yoshitoshi KUNIEDA ‡‡†

†Wakayama University Graduate School of Systems Engineering
‡Center for Information Science Wakayama University
‡†Faculty of System Engineering Wakayama University
‡‡Kyoto University　　‡‡†Ritsumeikan University

Abstract According to rapidly widespread of the Internet, we become generally use E-mail as an new communication system in global breadth. While E-mail system have no needs for severe realtime, but have to be an trusted one in the point of ensuring sending E-mail to the end.
E-mail redundant system have been designed and operated in the case of server maintenance or troubleshooting, such as several mail server compornents synchronized its mail spool data using NFS and/or DNS's MX record priority. However, these function still not be satisfied its reliability and availability because of no redundancy in stateful mail queue data.
In this paper, we address the elements for redundant mail system, then show the system including above stateful mail queue for efficient redundant mail system. Our system also have some advantages in the system management and long MTBF.
Key words : redundant，mail server

1　はじめに

我々が利用するインターネット上のサービスの一つに，電子メール（以下メール）が挙げられる．メールはインターネットにおける最もプリミティブな情報発信および情報共有の手法であり，現在は誰でも容易に利用できるサービスの一つである．近年は利用者にとって重要な情報がメールで通知されるケースも増大し，利用者におけるメールの重要性が高まるとともに，メールシステムへの高い信頼性が求められている．しかし，郵便と現在のメールサービスを比較すると，後者には相対的な信頼性の低さが認められる．これは，一般的なメールサービスでは転

送中の内容が平文であることや，転送経路がオーソライズされてないこと，および先方で受信したことが送信者に分からないこと等に起因する．

一方，インターネット上で利用される重要なサービスに World Wide Web（以下 WWW）がある．WWW サービスでは，信頼性を高めるためにサーバの冗長化や負荷分散という様々な手法が設計されている．例として，複数の WWW サーバ内のコンテンツをミラーリング等で共有し，同一のサービスを提供することが挙げられる．

メールサービスで上記と同様の手法を実現する場合，サービスを提供するシステムの構造および利用と運用形態の違いを十分に理解しなければならない．WWW システムでは情報取得者が直接サーバから情報を得るのに対し，メールシステムでは，メールは各サーバ間で転送（リレー）を繰り返され，送信者と受信者が直接に通信しない．さらに，WWW とメールの各サーバで取り扱うコンテンツを比較すると，WWW よりもメールのコンテンツの更新頻度が非常に高いため容易に冗長化できない．

メールが消失（ロスト）する原因の多くは，経路上よりも送受信および転送する各メールサーバ上に存在する．メールシステムでは，メールを受信したサーバから受信者が取得するまで保存するメール（スプールデータ）を冗長化する手法が数多く提唱されている．具体的には，NFS（Network File System）やミラーリングによって物理的に複数の箇所で受信メールを保存する．しかし，メールサーバ・プログラムの一つである Mail Transfer Agent（以下 MTA）が停止した場合，処理中であるメールの遷移状態を含めて冗長化されていないため，結果的に多くのメールがロストしてしまう．

本研究では，MTA がメールを送信する場合や，受信してからスプールに配信される場合の中間処理状態も冗長化の対象とする．単に MTA を MX（Mail eXchange）の優先度設定によって冗長化するだけでなく，遷移状態も含めて冗長化するため，これまでのメールシステムの冗長化より信頼性が高まると考える．更新頻度が非常に高い中間処理の状態を同期させる手法は，一般的な WWW サーバの冗長化と違い緻密なトランザクションを必要とする．

本紙では，メールシステムの信頼性を高めるため，上記の手法を実現するシステムを提案する．メールサーバおよび中間処理データの同期によって冗長化を図り，メールシステム全体としての MTBF（平均故障間隔）を長くする．さらに，運用管理面と送受信者からみたシステムの透過性と今後の課題について考察する．

2　冗長化の問題点

可用性と信頼性を得るための簡単で効果的な方法は，複数のサーバを使用することである．

本章では，既存の冗長化システムについて考察し，メールサーバの冗長化システムを設計する上での問題点を述べる．

2.1　WWW サーバとの比較

複数のサーバが設置されている場合，そのうちの 1 つがクラッシュしても，処理中のリクエストを別のサーバにリダイレクトできる．これにより，WWW サイトの可用性を高めている．信頼性およびスケーラビリティの観点から，複数のサーバに自身の Web サイトを処理させられる点は非常に重要である．

冗長化がされているサーバでは，複数個の設備がほぼ同時に故障しない限りサービスが停止しないため，冗長化がなされていない場合に比べて MTBF が大幅に長くなる．サービス停止を避けるため，WWW サーバではミラーリングや NFS を用いてコンテンツの同期をとり，多重化することでサービスの可用性を高めている．

WWW サーバの多くのコンテンツは，基本的に更新頻度が低い．したがって，複数サーバ間で同じコンテンツの共有が容易に行える．WWW サーバは，DNS の MX レコードによる優先度付けやロードバランサーなどを用いて，複数のサーバを同時に運営する方法を用いて冗長化されている．

一方，メールサーバ内で処理されるデータは，中間処理データも含めると非常に変化が激しく，コンテンツの更新頻度が低い WWW サーバで使われる同期方法を用いるだけでは，充分に同期できないと考える．

メールは，メールデータを送信者と受信者間でやりとりするだけなので，リレー中のサーバ内でロストした場合両者による発見が遅れてしまう．よって，

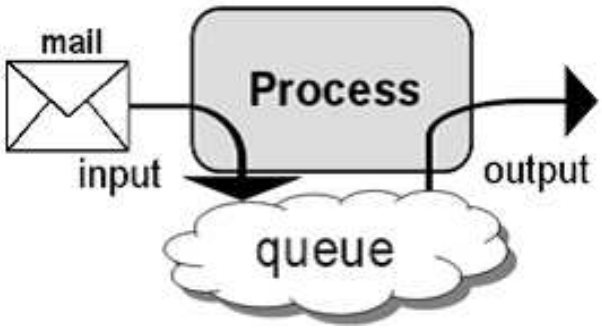

図 1: メール送信処理の流れ

個人間や企業間の連絡が行き届かずに不利益を受ける可能性が高い.

2.2 メールサーバ冗長化における問題点

メールサーバ停止を回避するために，現状では DNS の MX レコードによる優先度付けやロードバランサーなどを用いて，複数のサーバを同時に運営する等の手法がとられている.

一般的に，メールサーバのスプールの冗長化は NFS やミラーリングなど WWW サーバと同じような同期手法を用いて同期しているが，MTA が持つ中間処理データの同期は行われていない．したがって，メールシステムの冗長構成を考える場合，従来方式のようにスプールデータについてのみ同期を取るだけでは不十分である.

メール送信側は Input 処理が完了した時点で送信完了と認識してしまうが，実際は Input 処理が終了しても Output 処理が完了するまでは送信完了とは言えない．つまり，Input 処理が完了して，Output 処理を行うまでの中間処理状態の間にデータを紛失してしまう可能性がある．したがって，処理経過をトレースすることで中間処理状態のデータを保護する必要がある（図１）.

sendmail をはじめとするいくつかの MTA では，サーバ機の負荷（CPU 等）によって output を意図的に遅らせる機能がある．これは一般的な機能であり，メールがリアルタイムに配送される必要性を重視していないためである.

しかし，内容によっては数十分から数時間の停止がミッションクリティカルを招くケースもある.

メールサーバだけでなく多くのシステムでは，停電やハードウェアの故障という形でシステムが停止する．さらに，停電や故障などの機械的な要因だけでなく，システム変更・メンテナンス・バージョンアップを行うために管理者の手によって停止される可能性がある.

サーバが停止しただけならば，復旧後にサーバ内のキューディレクトリに保存されていたメールデータを送信することができる．しかし，その状態から送信者がはじめにメールを出したときから時間がかなり経過してしまう.

万一，ハードウエアが故障してしまうと，復旧されないメールデータが残る，あるいはメールデータがロストしてしまう．MTBF をのばし安定したサービス提供するためには，可能かぎりシステムの流れもとめないことが重要である.

上記で述べたメールサーバの冗長化について，以下に要点を述べる.

- メールシステム全体としてメールデータをロストさせない.
- 同データの流れを止めない
- 送受信者にとってミッションクリティカルとなる停止時間を提供しない.

2.3 サーバ停止を回避するために

サーバが停止した場合に取り残されるメールをなくし，メールシステムの流れをとめないためには，停止したメールサーバに取り残されるメールデータをなくし，メールがロストする可能性をなくすために，システムの動作・処理の流れを同期させる必要がある.

冗長化を行うには，以下の機能が同期システムに必要となるため，これらを考慮したシステム提案を行う.

動作の同期 処理中のメッセージを同期させることでメールシステムの同期を実現

死活監視 メインサーバが停止したことを知る

フェイルオーバ メインサーバからバックアップサーバへ処理を移行

フェイルバック バックアップサーバからメインサーバに処理を移行

3　提案システム

3.1　目的

本研究では，メールシステムの信頼性確保に重点をおき，メールデータのロストを最小限にすると共に，運用管理面，ユーザへの透過性を考慮したシステムの提案，実装を行う．

3.2　提案システムの概要

本研究では，MTA における実装の一つである Postfix を利用して，システムの実装を行う．代表的な MTA の実装として挙げられる Sendmail は柔軟性に富むという利点があるが，単一プロセス内でキューの状態遷移が行われるため，予期しないシステム停止時にはメールデータがロストする可能性が高い．対して，Postfix はモジュール化されたプロセスとメールキューディレクトリシステムを採用することにより，メールキューデータをトレースし，処理工程でのメールデータ保護機能を強化している．さらに，状態遷移を行う際に実行プロセスが切り替わるため，システム全体に変更処理を加えることなく，同期処理が必要なモジュールにのみシステムを実装すればよい．上記のように，メールデータの保護，実装時の利便性という観点から見た場合，Postfix がより適切であると判断した．

Postfix のメールディレクトリシステムとは，処理経過を保持するために，遷移状態をあらわす名前が付けられたディレクトリ（メールキュー）にメールデータを記録し，処理が中断された場合にもメールデータをロストすることなく処理を再開するための仕組みである．この仕組みを利用して，複数サーバ間でメールキューデータの同期を取ることにより，メールデータの保護も視野に入れたシステムの冗長化を行う．

3.3　Postfix のメールデータ処理工程

システムを実装するにあたり，メールキューディレクトリシステムにおける全メールキューの同期をとる必要性があるのか検討する．

全てのメールキューの同期を行えば，完全にメールデータを保護できると考えがちであるが，実際は

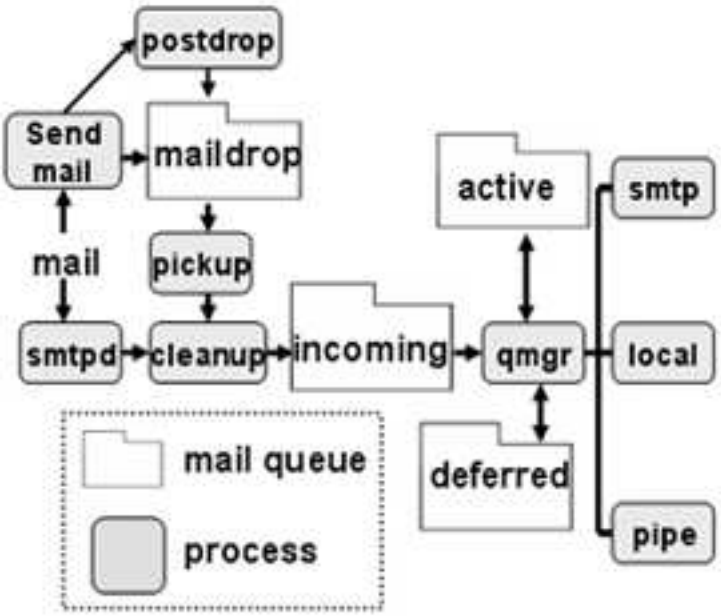

図 2: Postfix のおおまかな内部構造

キーポイントとなる幾つかのメールキューについて同期を行うことでメールデータを保護することが可能である．

そこで，メールシステムの挙動をトレースすることでメールデータを保護するためのキーポイントを特定した．

MTA がメッセージを受け取る経路は 2 通りに分けられる．1 つはローカルユーザからローカルキューを介してメッセージを受け取る経路．もう 1 つはリモートユーザからのメッセージを，SMTP(Simple Mail Transfer Protocol) を利用して受信する方法である．

前者の場合は，ローカルユーザからのメッセージを sendmail プログラムで受けると，メッセージは maildrop キューにわたされる．そして，pickup プログラムで maildrop に保存されているメッセージを定期的に抽出し，メッセージの正当性を確認した後，incoming キューにわたし，処理が可能な状態にする．incoming キューに入れられたメッセージは送信処理を受けるために qmgr プログラムによって active キューに入り，配信処理が行われる．

後者の場合は，smtpd プログラムで受けられた後，メッセージの正当性を確認し，incoming キューに渡される．その後はローカルユーザからのメッセージと同様の処理を受け，配送されていくという仕組

みである．

上記の流れより，メールの送受信に関してキーポイントとなるメールキューディレクトリ，すなわち，maildrop，incoming，active，deifferd の同期を取ることでシステムの処理工程を追跡し，メールデータを保護することが可能である．

maildrop ローカルユーザから受け取った新規メッセージが最初に入り，次の処理を待つキュー

incoming リモートホストから受け取った新規メッセージ，ローカルからの処理待ちメッセージが入るキュー

active 配信準備ができたデータが最終的な配送を待つキュー

deferd 再送処理待ちのデータが入るキュー

3.4 メールキューの同期手法

高負荷状況下におかれたサーバでは大量のメールキューデータが蓄積される．メールキューデータの同期手法としては，NFS や FTP(File Transfer Protocol) などを用いたシステムも考えられるが，Postfix はキューデータの i-node 番号によってファイルの整合性を検査する特性を持つため，NFS マウントしたデータは拒否される．さらに，FTP を用いた場合はメールキューに蓄積された大量のデータを同期するために，全データを送信しなければならないので，同期処理にかかるオーバーヘッドの大きさから適切な手法ではない．したがって，データの差分更新を行うことができる rsync が，大量のメールキューデータの同期を取るためには効果的である．

本システムでは，メールキューデータの状態遷移時にデータの同期処理を行い，メールシステム内の中間処理状態を引き継がせることで，メールシステムの冗長化を図り，システムの信頼性を向上させる．

3.5 フェイルオーバ

メインサーバとバックアップサーバ間で，中間処理データを同期をさせることで，各サーバは内部処理的に常に同じ状態に保持される．したがって，メインサーバが停止したと判断できれば，バックアップサーバを起動するだけで処理の引継ぎ処理を完了することができる．システムの稼動停止確認方法としては，待機サーバ（バックアップ）側から 25/tcp（smtp）ポートへの接続性確認を行い，応答があればシステムは正常に稼動していると判断する．連続して数回以上応答が無い場合，メインサーバのメールシステムが停止したと判断し，バックアップシステムを稼動させる．システムがバックアップ側に移行した場合，復旧作業を怠ると冗長構成を取る意味をなさなくなるため，バックアップシステムへ処理が移行されたことを管理者に通知する機構が必要になる．

3.6 フェイルバック

メインサーバが復帰し，バックアップサーバから処理が戻される際，完全自動化を行うと，メンテナンスのためにメインサーバを停止した場合など，システムのテストを行っている最中にメインサーバに処理が戻ってしまうなど，管理者が意図しないフェイルバックが起こる可能性がある．

手動でフェイルバックを行う場合，管理者の意向に沿った処理を施すことができる反面，一時的にメインサーバ，バックアップサーバともに停止状態へと陥る可能性がある．しかし，メールサーバの性質上，数分程度の停止ならば，実運用に耐えられると考える．

よって，フェイルバックに関しては，管理者のポリシに即した処置を施すため，自動で処理が戻す方法と管理者の手で処理を戻すという，2 つの手法を提供する．

3.7 ユーザへの透過性の実現方法

メール送信時は，DNS を用いてもメールが送信可能なサーバに到達するまで再送を試みるという動作が行われない．したがってメール受信処理を行う場合は，DNS の MX による優先度付けを行うことで，メールが受信可能なサーバに到達するまで再送処理が行われるため，滞りなく受信処理を行うことが可能であるが，送信処理を行う場合は注意が必要である．

たとえ同一ホスト名で複数の IP アドレスを設定

していたとしてもDNSへの名前問い合わせ時にラウンドロビンでホスト名が返されるため，停止中サーバのアドレスが返される可能性がある．すなわち，メール送信時のサーバ選択に関する仕組みの考案が今後必要になってくると考えられる．

4　評価・考察及び課題

本システムは，従来型のスプールデータのみ同期を取る手法と比較すると，メールをキューに入るたびに同期をとるのでデータが取り残されることなく処理を引き継ぎ，24時間365日メールシステムの処理動作が停止しないため，信頼性が高まると考える．しかし，キューデータの同期処理が含まれるため，実行速度の低下が懸念される．このため，メールキューの同期を取ることで，どの程度実行速度に差が出るのかを実測する必要がある．

さらに，実行速度の向上が当面の課題となるので，今後はPostfix内のメールキューの参照頻度の違いによって処理方法を変え，遅延を軽減する方式を考案する．

キューに蓄積されるデータの同期を取る際，データ量が多い場合はrsyncを用いる方法が適切であると考えられるが，少量の場合はより高速な同期方式を適用することで性能の向上が見込まれる．各メールキューの入出力特性について考察した．キーポイントとなる4つのメールキュー特性について以下に述べる．

maildropキューは配送処理に入るために，メールシステムによって定期的にキューがチェックされるという特性をもつため，メールシステムがチェックを行うまでの間，maildropキューにはローカルユーザからのメールが蓄積されていく．このため，一度に受け付けるメール数が多ければ，maildropキューに蓄積されるデータ量も増加する．

activeキューはシステムへの負荷軽減のため，一度に挿入されるデータ量が決められているので，データ蓄積量は非常に少量である．

incomingキュー，deferedキューは，activeキューに空きができ次第データを処理していくという特性をもつため，キューへのデータ蓄積量はactiveキューの空き状況（システムの処理速度）に依存する．

これらのメールキュー特性を考慮して，個別のキュー特性に応じた同期手法を適用すれば同期処理に関するオーバーヘッドを軽減することができる．

さらに，システム全体のパフォーマンスを向上させるためには，サービス稼動チェックのインターバルにも注意を払う必要がある．チェック頻度があまりに高いとサービスの質を損ねる恐れがあるため，チェックを行うインターバルの適正値を今後の実験を通して求めていく．

並行して検証する項目として，フェイルバック時に処理を移行する作業が挙げられる．フェイルバック時にメインサーバに処理を戻す際，メインサーバ，バックアップサーバともに停止する状態が数秒程度発生するが，メールサーバの性質上，数秒程度の停止ならば実運用に耐えられると考えられる．この考えが適当であるかは，実運用を通して検証を重ねる必要があるため，今後の評価に含める予定である．

5　まとめ

本研究では，メールデータのロストを最小限にし，メールシステムの信頼性を向上させることを主目的とし，運用管理面，ユーザへの透過性を考慮したシステムを提案した．更に，メールシステムの冗長化における問題点について明確にすると共に，解決方法について考察を行ってきたが，今後は実運用で本システムの有用性を検証していく．

参考文献

[1] Richard Blum ”Postfix メールサーバの構築 ”

[2] ”rsync ” http://samba.anu.edu.au/rsync/

[3] ”Postfix のページ ” http://www.kobitosan.net/postfix/

[4] J. Klensin ”RFC2821 Simple Mail Transfer Protocol (SMTP)”, April 2001

[5] P. Mockapetris ”RFC1034 Domain names - concepts and facilities (DNS)”, November 1987