mrubyを利用した軽量コンテナクラウド基盤の研究開発を介したmrubyの大規模・高負荷テスト

開発助成金成果報告

---

1. プロジェクト概要

ウェブホスティングにおける突発的なアクセス集中などの高負荷への対応について、容易に利用できるレンタルサーバ（共用サーバ）では利用者側がリソースの制御を行うことが難しい一方、柔軟性の高い専用サーバーやVPSはサーバーの拡張作業や監視といった運用管理が継続的に発生するという課題がある。我々はこの課題に対して、軽量コンテナを基盤としたホストの起動・複製・終了・資源割り当て変更処理などをリクエスト単位程度の粒度で高速に実現するシステムの開発と、IaaSを用いた実運用に近い環境でのホスティングサービスの実証実験を通じて、次世代的ホスティング環境基盤技術の研究開発を行う。これにより、高負荷時のスケーリングやコンピューティングインフラの増強・移行や、システムの再起動が必要なために滞りがちなセキュリティ更新や基盤ソフトウェアの入れ替え等が柔軟かつ自動的に行えるようになる。

2. 採択者

- 近藤宇智朗（GMOペパボ、報告書作成者）
- 小田知央（GMOペパボ）
- 松本亮介（GMOペパボ）
- 笠原義晃（九州大学）
- 嶋吉隆夫（九州大学）
- 金子晃介（九州大学）
- 岡村耕二（九州大学）

3. プロジェクトのゴール

ウェブホスティングのリソースの増強・セキュリティ更新などを柔軟かつ自動的に行うために、IaaSを用いた実運用に近い環境を用い、以下のような基盤ソフトウェアを開発・改善する。その過程で、主にmrubyに関連するライブラリやツール、ミドルウェアをOSSにする。

合わせて、このような大規模・高負荷環境でmruby自身に起こる問題などについても改善を提案する。

- コンテナを基盤としたホストの起動・複製・終了・資源割り当て変更処理などをリクエスト単位程度の粒度で高速に実現するシステムに関連するソフトウェア
- 大規模なメール送受信基盤に関係するmruby拡張ミドルウェアなどのソフトウェア
- それらのログなどの情報を解析するライブラリ・ツール

4. 今回開発あるいは改善したシステム・ソフトウェア

4.1. Haconiwa

概要

Haconiwaは、mrubyを組み込んだコンテナランタイムである。特徴としてはコンテナに関するカーネルの機能を自由に組み合わせが出来る点、コンテナのライフサイクルや起動後一定時間経過に応じたフック処理をmrubyでプログラミング可能な点がある。

今回の開発箇所

大規模なホスティング基盤に利用するため、今回以下のような機能追加を実施済みである。

- 2017末段階でのmruby最新化 (https://github.com/haconiwa/haconiwa/pull/126)
- LXCFSとの連携 (https://github.com/haconiwa/haconiwa/pull/127)
- ファイルマウントオプションの詳細化 (https://github.com/haconiwa/haconiwa/pull/131)
- ネットワーク機能の追加 (https://github.com/haconiwa/haconiwa/pull/132)

また、本助成による開発の成果を含んだHaconiwaに関する予稿を、情報処理学会第80回全国大会に提出している。

- 近藤 宇智朗, 松本 亮介, 栗林, 健太郎, Haconiwa: プログラムによる、組み立て可能性と拡張性を持つLinuxコンテナ, 情報処理学会第80回全国大会公演論文集, Mar 2018, https://rand.pepabo.com/papers/ipsj-nc80-udzura.pdf.

4.2. pmilter

概要

呼称はProgrammable Mail Filterの略。SMTPサーバ（送信）とmilterプロトコルで通信し、SMTPサーバの送受信の振る舞いをmrubyでコントロールできるサーバソフトウェアである。

今回の開発箇所

pmilterをコンセプトから一通り実装完了した。本ソフトウェアは後述するメール基盤に必要である。

ブログによる紹介はURL(http://hb.matsumoto-r.jp/entry/2016/11/03/121517)
を参照のこと。

4.3. dovecot-mruby-plugin

概要

メール受信（POP/IMAP）サーバの振る舞いをmrubyにより制御するプラグイン型ミドルウェア。ベースの実装にはDovecotを用いている。例えばDovecotにMRUBYコマンドを送信することで、mrubyスクリプトを評価し実行することができる。

今回の開発箇所

こちらもコンセプトから一通り実装完了した。本ソフトウェアは後述するメール基盤に必要である。

ブログによる紹介はURL(http://hb.matsumoto-r.jp/entry/2017/09/14/143030)を参照のこと。

4.4. ngx_mruby

概要

ngx_mrubyはNginxのフェーズごとの振る舞い、変数への値の設定などをmrubyにより記述することができるようにした拡張である。従来はそのような拡張はC言語で書き、コンパイルが必要であったが、ngx_mrubyによりスクリプト言語で振る舞いを記述できるようになった。他の言語組み込み実装よりも高速である点、mrbgemsを用いて自由に拡張が可能な点も特徴である。

今回の開発箇所

今回、ホスティング基盤のサービスメッシュ層にngx_mrubyを利用するにあたり、ngx_mrubyが実行するmrubyのコードの内部で実行がブロックするようなパターンが生じたときに、ワーカーのスケジュールング、個数の状況により他のリクエストも合わせてブロックしてしまうという問題が発生した。

ワーカ数を増やすなどの緩和策はあるが、根本解決のため、ngx_mrubyをmrubyのFiberクラスをベースとした設計に修正し、nginxの元々の設計思想に合わせてリクエストのハンドリングを非同期化する改修を行った。現在全ての既存のテストが通過する状態で、バージョン2としてブランチを切っており、利用可能な状態である。

4.5. FastContainer

概要

FastContainerとはWebアプリケーションコンテナの状態をリアクティブに決定するコンテナ管理アーキテクチャである。具体的には、アクセスの急増、リソースの枯渇、ライブラリの危殆化に伴う再起動の必要性などのホストや環境の状況の変化に素早く対応（リアクティブ）し、コンテナの起動、停止などの状態をコントロールする。この目的のため、FastCGIを参考にコンテナの一定期間の起動と廃棄を繰り返す管理モデルを取っている。また起動の粒度は原則アクセス単位としており充分に細かい。

今回の開発箇所

もともとFastContainerはWebを基盤として設計が議論されており、最初はWebホスティング基盤に導入される形となった。したがって、プロトコルとしてHTTPが前提となる戦略であった。

今回、FastContainerをメール基盤に拡張するにあたり、TCP/UDP一般を前提としたアーキテクチャの検証を行った。結果、ngx_mrubyを利用すれば問題がないことを示すことができた。

その上で、その成果とHaconiwa、先述したようなメール関連のmruby組み込みミドルウェアを活用し、大規模な配信・受信に耐えうるコンテナ基盤前提のメールアーキテクチャの設計が完了している。

また、FastContainerアーキテクチャをmrubyを活用して実装していく中で、幾つかの課題が整理できたため、それらについて以下の研究会で発表を行った。

- 笠原 義晃, 松本 亮介, 近藤 宇智朗, 小田 知央, 嶋吉 隆夫, 金子晃介, 岡村 耕二, 軽量コンテナに基づく柔軟なホスティング・クラウド基盤の研究開発と大規模・高負荷テスト環境の構築, 研究報告インターネットと運用技術（IOT）, Vol.2018-IOT-40(2), pp.1-8, Mar 2018.
- 松本亮介, 近藤宇智朗, 三宅悠介, 力武健次, 栗林健太郎, FastContainer: 実行環境の変化に素早く適応できる恒常性を持つシステムアーキテクチャ, インターネットと運用技術シンポジウム2017論文集，2017，89-97（2017-11-30）, 2017年12月.

5. 当初の予定から見た未達成点・課題など

5.1. Haconiwa

概要

先述のとおりである。

課題と今後の予定

現在のHaconiwaに組み込まれたmrubyは1.3.0系統でありやや古い。mruby 1.4.0 に対応し、コアの改善の恩恵を受けるべきである。

しかしアップグレードを試みたところ、mruby-threadとそれを利用したmrubygems(mruby-timer-thread, mruby-signal-thread)のテストが通過しないケースが間々あった。その上、`MRB_GC_STRESS` を定義した状態でのビルドでは必ずテストが失敗することが確認できた。

さらに、スレッド作成時にmrubyのGCを強制停止するようなパッチ `a855e3e` (https://github.com/udzura/mruby-thread/commit/a855e3e94a3a4fd6d240266bcb04cfb6717ab5df) を作成し、同様のテストを実施したところ、スレッド関連に関しては全てのテストが正常に通過するようになった。

FastContainerのような場合にHaconiwaを利用するのであれば、コンテナの生存期間は有限かつ比較的短期であるためメモリ肥大などの問題は緩和されるが、一般的な用途としてGCの停止は望ましくなく、問題になった。

結論として、まつもとゆきひろメンターに相談した結果、ngx_mrubyと同様Fiberを用いて、mruby-threadで実現していたコンテナプロセスの`wait()`とは非同期に処理を実行する実装（タイムアウト・インターバルフック処理）を実現するよう検討している。

5.2. mrubyによるコンテナオーケストレーションスタック

概要

現在、商用環境においてmrubyと、他言語の実装によるミドルウェアスタックを利用したウェブホスティング環境（ロリポップ！マネージドクラウド）が存在するが、実証実験にあたり、その環境の可搬性を上げるべくOSSの参考実装を作成する。そのスタックを仮に、NHHM Stackと呼んでいる。

以下のような実装を含む。

- ロードバランシング
- CMDB管理APIデーモン
- ジョブ管理デーモン
- 監視基盤、オートスケール基盤

課題と今後の予定

Rubyによる実装（gem名: marfusha, https://github.com/udzura/marfusha）を検討しているが、先述したHaconiwaのmruby-threadの問題への対処などもあり、期間中に時間を取ることが

できなかった。今後OSSとして実装を進め、2018年のRubyKaigiをめどに何かしらの成果を公表する予定である。

5.3. FastContainer

概要

先述のとおりである。

課題と今後の予定

mrubyで実際に実装していく中で以下の課題が明らかとなった。

- リソース効率化の実証実験
- サービスレベルの適切な導出方法
- コンテナ起動時間の影響低減
- 要スケーリング状態の迅速な検知と誤検知の扱い

詳細は、論文「軽量コンテナに基づく柔軟なホスティング・クラウド基盤の研究開発と大規模・高負荷テスト環境の構築」にて言及している。上記の課題について、mrubyで実装するために、コンテナプロセスのイメージ化を行うためのmruby-criuを引き続き実装していく必要がある。

5.4. 精緻な解析と制御が可能な恒常性のあるメール基盤の設計

概要

FastContainer上にメール・ソフトウェアのmruby拡張を活用して、精緻な解析と制御が可能な恒常性のあるメール基盤の設計を行った。また、その実装には、mrubyでMilter処理を可能なpmilter、mrubyでdovecotを拡張可能なdovecot-mruby-plugin、mrubyでPostfixのデータのlookup処理を記述可能なpostfix-mrubyを利用して実験的実装を行い、動作することを確認した。また、研究会にて論文にまとめ報告し、一定の評価を得られた。

- 松本 亮介, 小田 知央, 笠原 義晃, 嶋吉 隆夫, 金子晃介, 栗林 健太郎, 岡村 耕二, 精緻に解析可能な恒常性のあるメール基盤の設計, 研究報告インターネットと運用技術（IOT）, Vol.2018-IOT-40(17), pp.1-8, Mar 2018.

今後の課題の予定

引き続き実験的環境から、構築環境を新たに用意して、よりプロダクション環境に近い品質の基盤を構築していく。

以上