Kubernetesクラスタ上でのeBPFを用いたサーバレス基盤の実装と評価

GitHubレポジトリ

コントローラ　デーモン

東京都立産業技高等専門学校3年
ニノ方 理仁

序論：サーバレス基盤運用の課題

サーバレスアーキテクチャ普及の背景

- **クラウドネイティブアーキテクチャ**の利用増大傾向により、サーバレス基盤の需要が高まった。
- サーバレス基盤をKubernetes上に実装すると、既存の**エコシステム**や**宣言的なシステム**を利用可能になる利点がある。

課題

- アクセス時のコールドブート時間の長さ
- 待機時間の増大によるUX(ユーザエクスペリエンス)の低下

→ **eBPF**を用いてリクエストの監視を低レイヤ化することで**コールドブートに要する時間を短縮**できる

eBPF(extended Berkeley Packet Filter)とは

Linuxカーネル内のサンドボックス環境下でプログラムを実行可能にすることで、動的にカーネルを拡張できる技術

目的

TCPレベルでリクエストの監視を行うサーバレス基盤をKubernetes上に実装し、コールドブートに要する時間の短縮率を評価した。

手法：サーバレス基盤の実装と仕様

本研究では、XDPを用いてクラスタ外からのTCPリクエストを検知し、独自プロトコルを用いて迅速に伝達するシステムを実装した。

XDP(eXpress Data Path)とは

Linuxカーネルに実装されたeBPFを用いたネットワークパケット高速処理基盤である。データリンク層レベルでのパケット操作を行う手段をユーザ空間に提供する。

仕様

- 本提案はコントローラ、デーモン、XDPプログラムの3構成とした。
  - **コントローラ** (Go言語で実装)
    - Kubernetes Operatorとして実装し、Kubernetesのマニフェストとしてサーバレスアプリケーションの**宣言的定義**を可能にした。
  - **デーモン** (Go言語で実装)
    - KubernetesのDaemonSetを用いて全ノードに配置され、ホストのインターフェースにXDPプログラムをアタッチする。
    - サーバレスアプリケーションの情報をコントローラと同期する。
  - **XDPプログラム** (C言語で実装)
    - カーネル空間でパケットを処理する。
- UDP上に独自プロトコルを実装し、コントローラとデーモン間での通信速度を**高速化**した。
  - Kubernetesのノードは複数になり得るため、クライアントサーバモデルを採用した。
  - パケットを定期的に送り合ってピア状態を維持し、サーバレスアプリケーションの情報を同期するようにした。
- リクエストは以下の手順で処理する。
  1. XDPプログラムでユーザからサーバレスアプリケーションへのSYNパケットを待ち受ける。
  2. デーモンから**独自プロトコル**でコントローラに宛先IPアドレス・ポート・シーケンス番号を通知する。
  3. コントローラからコンテナを起動する。
  4. コンテナにSYNを一定間隔で送り、ACKが帰ってくるまで待機する。
  5. 最初のリクエスト時のシーケンス番号を用いてSYNを送信し、ユーザとの通信を確立する。

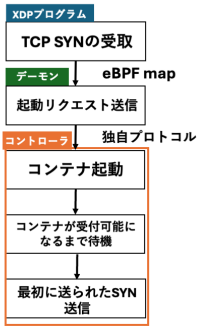

図1. 処理フロー

図2. マニフェストの定義

評価：コールドブート所要時間の評価

対象

- 提案システムとKnative上にサーバレスアプリケーションをデプロイし、コールドブートに要した時間を評価した。KnativeはKubernetes上で動作する既存のサーバレス基盤である。
- デプロイしたサーバレスアプリケーションは、Go言語で実装した文字列を返すHTTPサーバである。

動作環境

Kubernetesのバージョンはv1.30.2、CNIとしてFlannel、ロードバランサとしてMetalLB、Knativeはv0.43.0、L7ロードバランサとしてContourを使用した。

方法

コールド状態のサーバレスアプリケーションにリクエストを行いレスポンスまでの所要時間を10回計測し平均した。

結果と考察

結果

提案システムと既存システムのコールドブート所要時間を比較した結果は以下のとおりである。

- コールドブートに要した時間は、既存システムが平均1307ミリ秒、提案システムが平均1038ミリ秒であった。

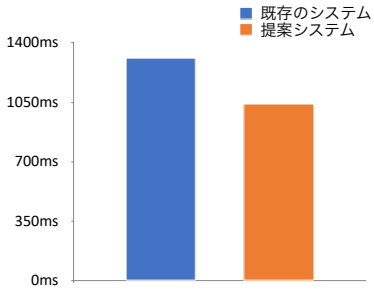

図3. コールドブート所要時間比較

- 既存システムに比べ提案システムのコールドブートに要した時間は**20.6%減少**した。
- コールドブートに要した時間の標準偏差は、既存システムが0.407、提案システムが0.0160であった。
- 提案システムでは、既存システムに比べ、安定して**コールドブート所要時間の短縮**がみられた。

結論

結論

サーバレスアーキテクチャの利用が一般的になったことにより、サーバレス基盤も利用増大傾向にある。しかし、アクセス時のコールドブート時間の長さと待機時間の増大によるUXの低下は課題である。本研究では、XDPを用いてクラスタ外からのリクエストを検知し、独自プロトコルを用いて迅速に伝達するサーバレス基盤を実装した。
その結果、**コールドブート時間を短縮**できることが示された。既存のシステムと比べて**低レイヤで実装**した提案システムの特徴が、高速化や効率化に寄与したと言える。また、低レイヤで処理することによる高速化へのアプローチは、リソースの削減やスケーラビリティの向上を考える上でも意味深いと考える。

課題

本研究では、リクエストに応じてコールド状態からコンテナを起動する仕組みを実装した。実際の運用においてはリクエスト量の増加に応じて複数のインスタンスへスケールアウトする仕組みも必要である。今後、負荷分散についても検討することが課題である。

参考文献

Lin, P. M., & Glikson, A. (2019). Mitigating cold starts in serverless platforms: A pool-based approach. *arXiv preprint arXiv:1903.12221*.