# eBPF を用いた低遅延電力・温度モニタリング基盤の設計と実装

2022TC065 對馬秀悟
指導教員：宮澤元

## 1　はじめに

本研究では，eBPF と拡張スケジューラである SCX を用いたカーネル空間での温度, 電力抑制スケジューリングの試作を行い, 応答時間およびスループットの観点から評価を行う.
近年カーネル内に小型プログラムを安全に差し込むことで機能を拡張できる eBPF が注目されている [1].eBPF を用いたメトリクスのリアルタイムの監視ツールやスケジューラを動的に差し替えることができる SCX など多くのものが登場しているが, それらを統合したものは少ない. 本研究では温度と電力の監視から制御までカーネル空間で実施できるエッジデバイスの基盤を提案する.
本研究の目的は，エッジ環境において消費電力を抑えながら体感を損なわない応答時間で運用可能であることを示す.
カーネル内に観測ポイントを設置して温度をカーネル内で取得し，BPF マップに保持する．SCX 側の BPF スケジューラは，この情報と各タスクの指標を用いてキュー割付けを段階的に縮退させる．ユーザー空間への往復を避けることで，意思決定から制御の遅延とオーバーヘッドを最小化することが狙いである．本研究の貢献は以下の点である.

- カーネル内で完結する高温防止制御を提示し, 試作を行う
- 試作したシステムを用いて温度, 消費電力, 応答時間とスループットを評価する

## 2　背景

近年, エッジ環境や小規模クラスタでは, 限られた冷却・電力予算の下で応答性とスループットを両立する運用が求められている.
Linux には温度トリップポイントに応じて周波数制御や冷却デバイスを動かす Thermal　Framework ／ CPUfreq といった仕組みが用意されているが温度や使用率による判別であり, タスクの状況に応じた処理には対応しきれていない. Kepler は eBPF とセンサ情報等を活用して，コンテナ／ Pod 単位でエネルギー関連メトリクスを推定・輸出するプロジェクトである [2]. 一方，SCX は Linux 6.12 で公式ドキュメントに掲載された BPF でスケジューラを実装できる拡張可能なスケジューラクラスであり，スケジューリング挙動を BPF プログラムで定義可能にする [3].
本研究はその推定結果を用いた外部制御ではなく，SCX によりスケジューリングを直接カーネル内で制御する点に主眼がある.

## 3　設計

### 3.1　システム

本研究のシステムを図 1 に示す．本研究は以下の 2 つの

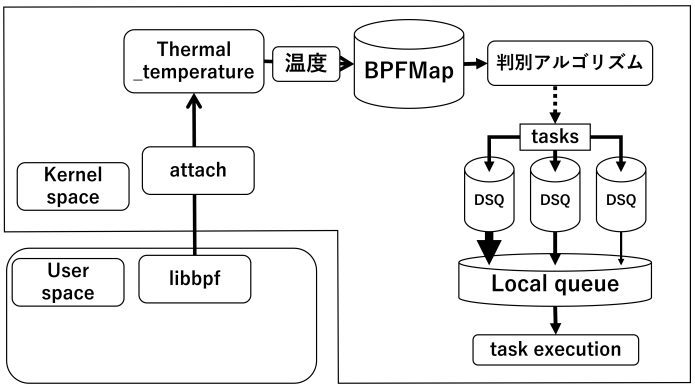


図 1　システムの構成

プログラムで構成されている.
メトリクス取得側:tracepoint/thermal/thermal_temperature をフックし，温度段階を決定して BPF マップに保存する.
スケジューリング側:SCX で BPF マップから温度を参照し，必要に応じ負荷軽減を実施する.

### 3.2　現状

現状の負荷軽減方法は BPFMap を参照し, 予め決めた閾値を跨いだら

- 参照する FIFO キュー本数の縮退
- タイムスライスの短縮
- 上位キューの優先的な参照

などを行う.
ただし, この策ははメトリクス取得と SCX の動作や連携の確認のためのものである.

### 3.3　理想

現状の設計の問題点としてはタスクをキューに割り振る際のスコアリングの負荷, 負荷対策がほぼ一意的に決まってしまう点, 高温時は下位キューを全く参照しないなどがある. 特にスコアリングはタスク毎に計算する必要があるので高負荷時にはかなり影響してしまう. 対処するためには

- ラベルの導入
- 優先処理をするキューの導入
- 高負荷時の負荷計算方法の見直し

が有効であると考えられる. 上 2 つはそれ単体での実装は済んでいる. そのため後は現状の方法との相性の確認と負荷の計算方法を詰めていく必要がある.

## 4 実装

以下表 1 に実験環境を示す.

表 1 実験環境

| | |
|---|---|
| 使用機器 | Raspberry Pi 5 Model(メモリ 8GB) |
| バージョン | Linux raspberrypi 6.12.34-v8-16k |
| eBPF バージョン | |
| SCX バージョン | |

また, 環境構築に当たりカーネルのオプションを変更した.Raspberry Pi ではデフォルトではオフになっているので変更する必要がある. 以下表 2 に変更を加えたオプションの一部を示す. 有効にする必要があるオプションの詳細

表 2 変更したオプション (一部抜粋)

| |
|---|
| CONFIG_BPF |
| CONFIG_BPF_SYSCALL |
| CONFIG_SCHED_CLASS_EXT |
| CONFIG_DEBUG_INFO_BTF |
| CONFIG_BPF_JIT_ALWAYS_ON |

については公式サイト [3] を参照されたい.

## 5 実験

温度ピーク抑制量とスループット低下のトレードオフ, および遅延のばらつきを評価観点とする.
今回は stress-ng コマンドを用いて負荷をかけた.
設計で説明したプログラムは期待通りの挙動をしない. 具体的には I/O 系の負荷をかけたときに実行時間も温度も通常時より高くなっている. これはタスク割り当て時の基準で I/O が後回しにされがちとなるためである.
今後はこれらの問題点に対処しつつ温度低下とスループット向上に努めていく.

## 6 おわりに

本稿では eBPF と SCX によりカーネル空間で温度指標に応じてスケジューリング挙動を制御する試作系の設計と実装要点を示した.
今後は電力測定機器の追加, プログラムの基本動作の検証を終えたため, 今後は設計を再構成し, 堅牢性・保守性・拡張性を備えた本実装へ移行する.

## 参考文献


[1] eBPF *eBPF Documentation* https://ebpf.io/what-is-ebpf/ (accessed September 29.)

[2] Marcelo Amara, Huamin Chen, Tatsuhiro Chiba, Rina Nakazawa, Sunyanan Choochotkaew, Eun Kyung Lee and Tamar Eilam: *Process-based Efficient Power Level Exporter*. 2024 IEEE 7th International Conference on Cloud Computing (CLOUD).

[3] The Linux Kernel *Extensible Scheduler Class*. https://docs.kernel.org/scheduler/sched-ext.html (accessed September 29.)