クラウド環境を指向するライブラリ OS の分類のための起動にかかる時間の計測

金津穂 † 田崎創 ‡ 宇夫陽次朗 ‡ 山田浩史 †

† 東京農工大学大学院

‡IIJ 技術研究所

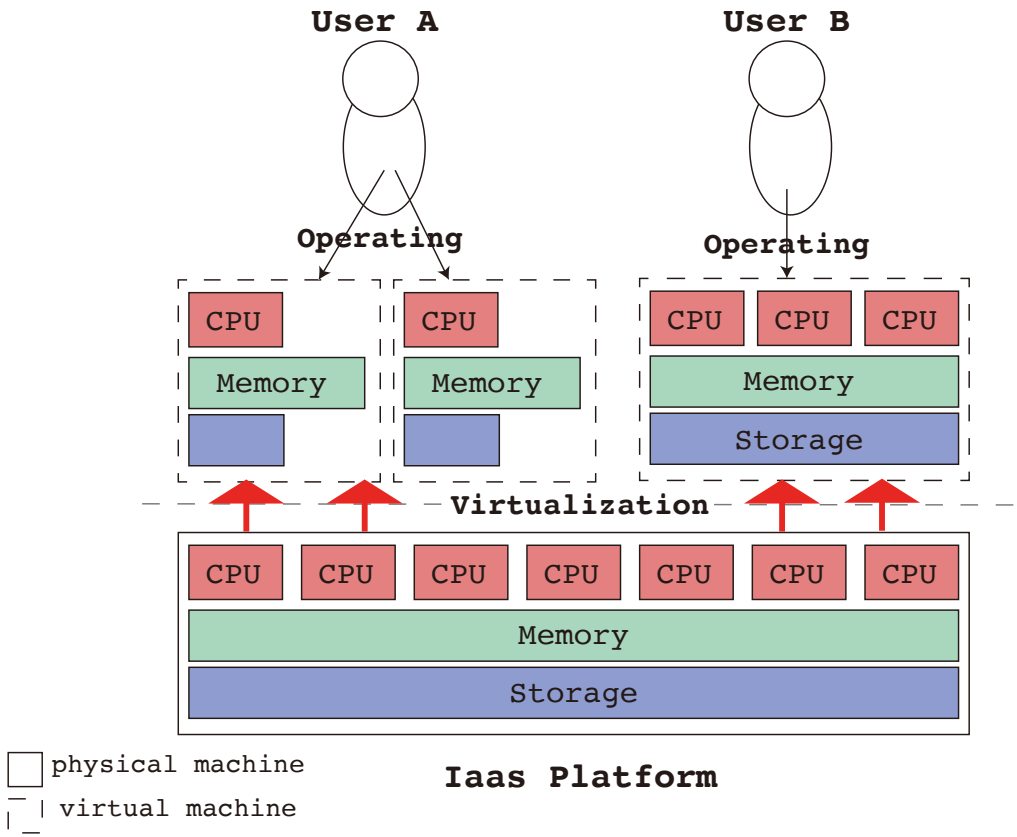
Oct 11th, 2016

あらまし

- クラウド環境向けに複数の LibOS が出現している
- 既存の OS と LibOS の比較は多いが LibOS 同士の比較が少ない
- LibOS を起動にかかる時間によって比較した
  - クラウド環境において起動にかかる時間は重要なファクターのひとつ
  - クラウド向けの LibOS のうちいずれが良いのか尺度の導入
  - 手軽に測定できる
- 計測により次の事実が判明した
  - 軽量のインスタンス，LibOS，Linux の起動時間にそれぞれ違いが見られた.
  - それぞれの起動時間について，メモリの消費量との相関が見られた.
  - LibOS の実装の違いについては起動時間から断定できるほどのデータが見られなかった.

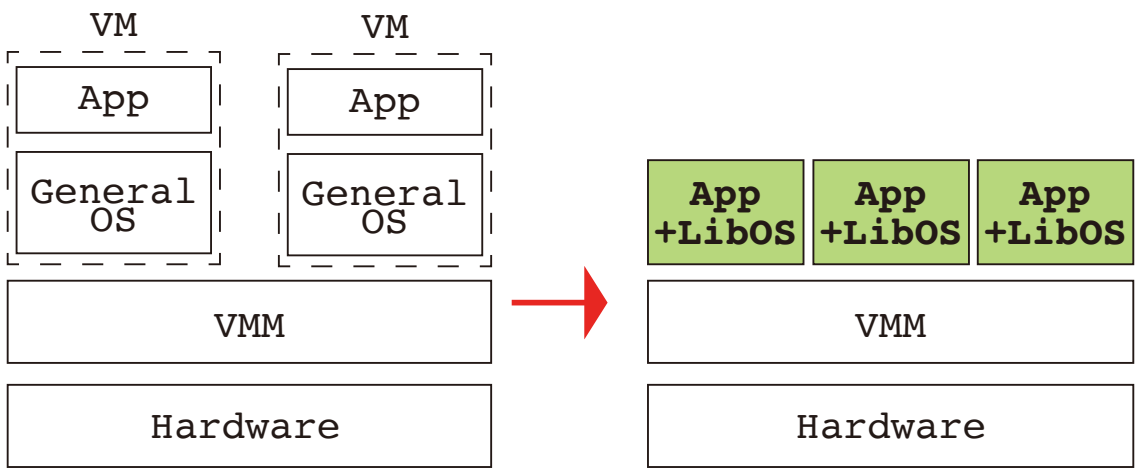
クラウド環境（IaaS プラットフォーム）

- IaaS プラットフォーム（Amazon EC2，Google Compute Engine）
  - Type 1 ハイパーバイザによるマルチテナント
  - オンデマンドに計算資源を増減可能
  - アプリケーションインスタンスの作成／削除が容易



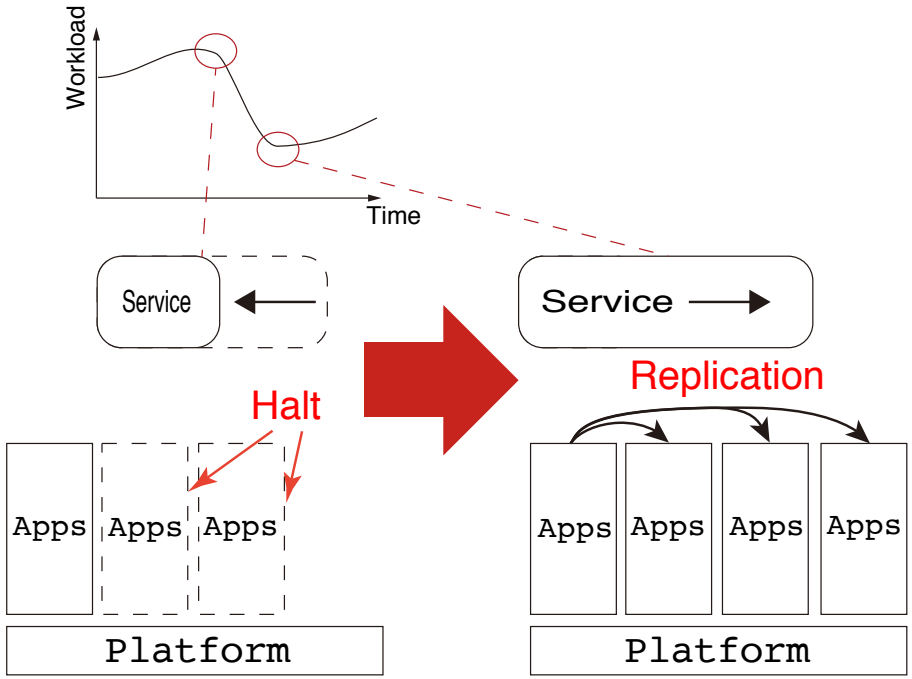
ライブラリ OS（LibOS）

- OS の機能をライブラリに持たせアプリケーションにリンクして使用するもの
- IaaS では特定のサービスや単一のアプリケーションのホストをする事が多い
- LibOS をリンクしたアプリケーションならハイパーバイザ上に直接デプロイ可能
  - 既存の汎用 OS とアプリケーションの構成とも同居可能
  - 複数のアプリケーションの動作を目的とした汎用 OS の機能が不要
  - 特定のサービスの起動のために実行されるコードが Linux 等に比べ少ない
- OSv[Kivity et al., 2014] や Unikernel[Madhavapeddy et al., 2013] など



クラウドでの起動にかかる時間—Elasticity

- IaaS では
  - ホストするサービスのアクセスが集中にインスタンスの増加による対応が考えられる
  - 負荷に応じてサービスの稼動規模を増減させることが出来る
- ユーザのリクエストに対しインスタンスの作成・起動が遅い場合
  - その時点で起こっている過負荷への対応が遅くなる
  - つまりサービスの品質低下
  - 異常発生時のサービスの再起動も遅くなる



クラウドでの起動にかかる時間—障害復旧

- Amazon EC2 のような巨大なシステムでホストされるサービスは無数である
- 障害発生時に復旧をするために，なるべく早く復旧する必要がある
  - 数千もインスタンスがある場合，通常の Linux を一気に起動しようとすると詰まってしまう
  - このような場合にいかにインスタンスを早く立ち上げ通常状態に戻すのか

クラウドと LibOS

- 同じサービスを稼動させる場合，Linux より LibOS のほうが起動にかかる時間が小さくなる可能性がある
  - 最低限の機能のみをリンクさせる事が可能
  - メモリフットプリントと実行される命令数の両方で有利

⇒高い Elasticity，障害復旧性が実現可能

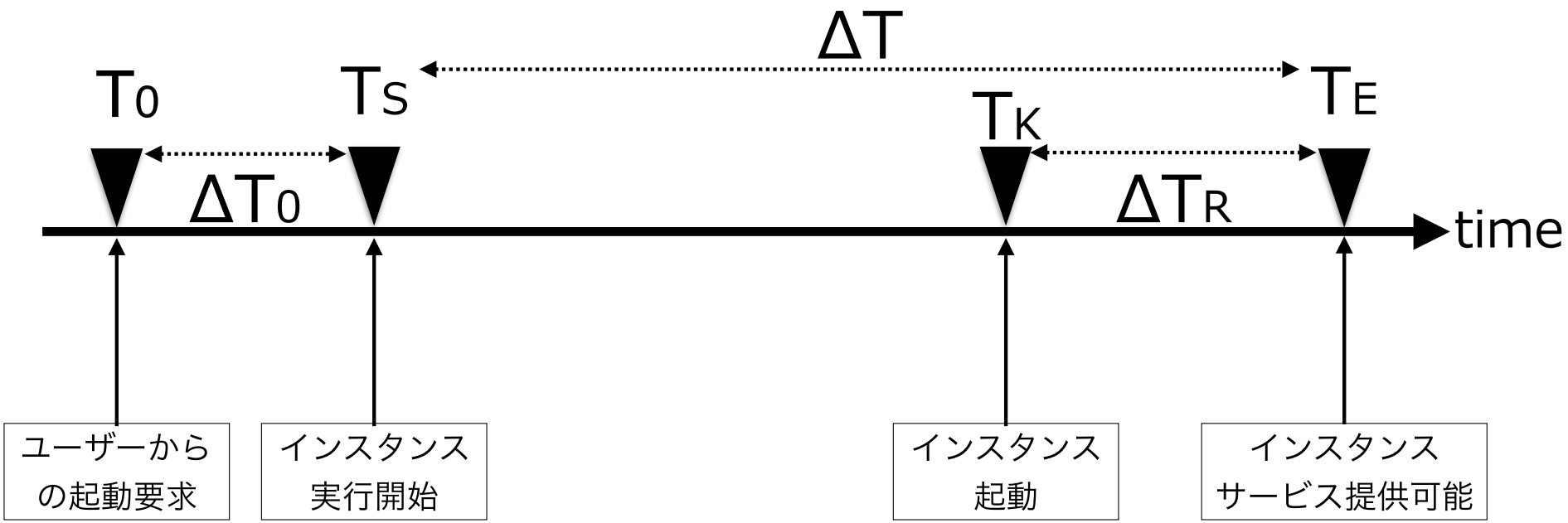
- 負荷上昇時のアプリケーションの起動
- 大量の代替インスタンスを一度に起動

本研究

- クラウドを指向する LibOS の起動にかかる時間の計測・比較
- クラウド環境向けの LibOS は複数存在する
  - どれが一番運用する上で好ましいか判断のための尺度が必要
- 起動にかかる時間をひとつの尺度として導入する試み
  - 3 種類の LibOS について計測を行なった
- 複数の条件で実験
  - 同時起動数やインスタンスを起動する間隔

起動にかかる時間の計測

- クラウド環境での起動にかかる速度をひとつの基準として考える
- 起動にかかる時間について LibOS がサービスをサーブできる状態になった時と考える
  - 図の ΔT について計測する事でその時間を割り出す



計測内容

- 同時起動数を変化させた場合の起動にかかる時間を計測
  - 障害復旧，Elasticity
- 64，128，256，512，1024，1280
- 各インスタンスの起動にかかる時間とシステム全体のメモリ消費量の遷移を記録
- インスタンスの起動時，各インスタンスの起動を 5 msec 間隔を開けた場合
  - *minkernel*，LibOS，Linux ではそれぞれ違いが出たが，LibOS 同士での違いはあまり見られなかった
- インスタンスの起動時，各インスタンスの起動を間隔を開けずに行なった場合
  - 測定時の，既に同時に N インスタンス起動している場合に起動にかかった時間を集計

起動にかかる時間のベースラインについて

- 計測環境における起動にかかる時間のベースラインとは
  - どの OS の起動にも仮想化環境の初期化やブートローダの実行が必要
- QEMU+KVM の初期化の直後が起動終了時刻になるモノがベースラインと考えられる
  - *minkernel* と名付けた右のコードを実行するだけのものを作成
  - QEMU のカーネルダイレクトロードによって読み込んだ

```
movl $stack_top, %esp
mov $0x03f8,%dx
mov $0x48,%al
out %al,%dx
mov $0x65,%al
out %al,%dx
mov $0x6c,%al
out %al,%dx
mov $0x6c,%al
out %al,%dx
mov $0x6f,%al
out %al,%dx
mov $0x0a,%al
out %al,%dx
cli
int $0x80
hlt
```

https://orum.in/gitbucket/orumin/libos_measurement/blob/master/minkernel_i386/srcs/boot.s

実験環境

- 今回は QEMU+KVM 上での動作を前提とした

実験環境

ベンダー	QCT D51U
CPU	Intel®Xeon®CPU E5-2640 v3 @ 2.60GHz （8 コア 2way）
主記憶	DD4 256GB
ストレージ	Intel S3610 SSD （128GB）
ホスト OS	CentOS 7 （Linux kernel 3.10.0-327.el7.x86_64）
VMM	QEMU（version 2.0.0）+ KVM

- 次にあるプログラムを用いた

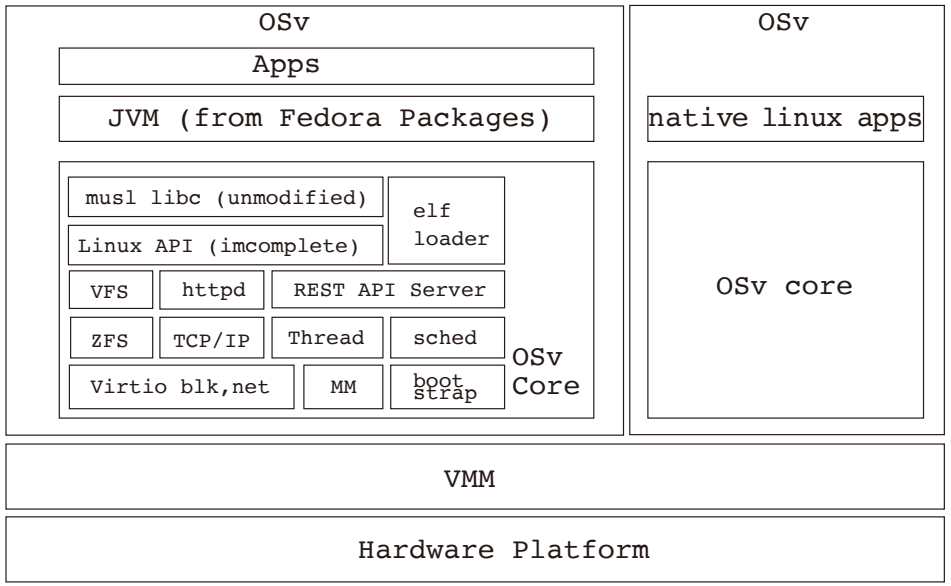
`https://orum.in/gitbucket/orumin/libos_measurement`

計測に用いた LibOS

- OSv[Kivity et al., 2014], IncludeOS[Bratterud et al., 2015], Rumprun[Kantee and Cormack, 2014]
  - フルスクラッチで作られた LibOS（OSv, IncludeOS）
  - 既存の OS コードを流用したもの（Rumprun）

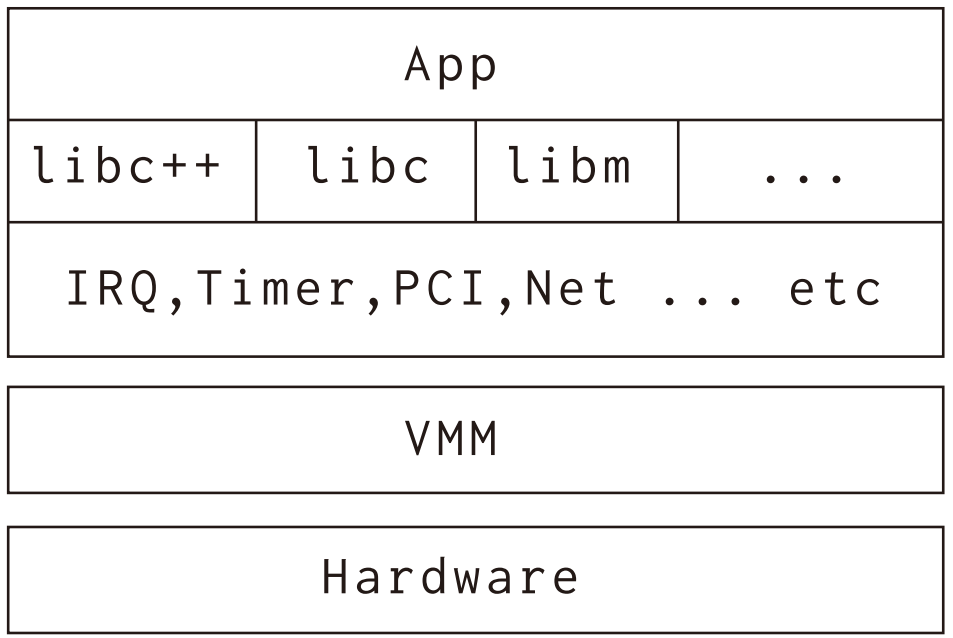
OSv

- 複数のハイパーバイザー向けにフルスクラッチで作成されている
- 単一のメモリ空間
  - システムコール呼び出しの際にメモリコピーが発生しない
- 既存の Linux バイナリの動作のために POSIX API を一部サポート



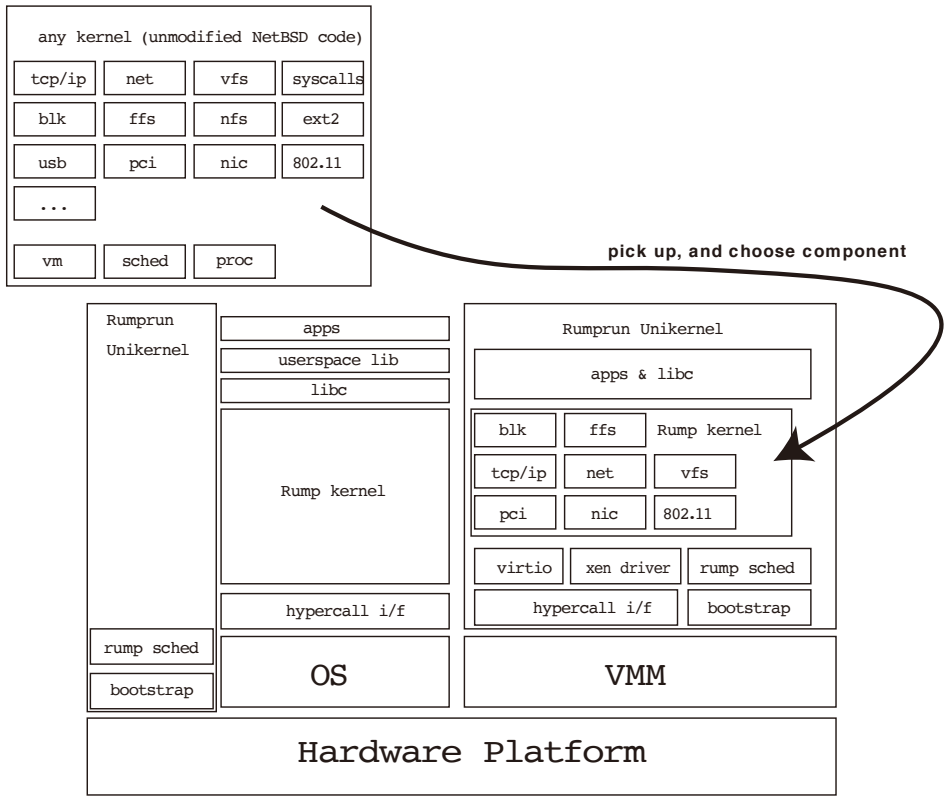
IncludeOS

- C++ と STL によるフルスクラッチ
- 全て静的バイナリとしてリンクされる
- ゼロオーバーヘッド原則を意識した設計
- 単一のメモリ空間



Rumprun

- NetBSD のコードを利用する Rump Kernel の派生
- Rump Kernel にブートストラップやハイパーコールの I/F を追加
- 独自のスケジューラをリンクしてベアメタルでも動作可能

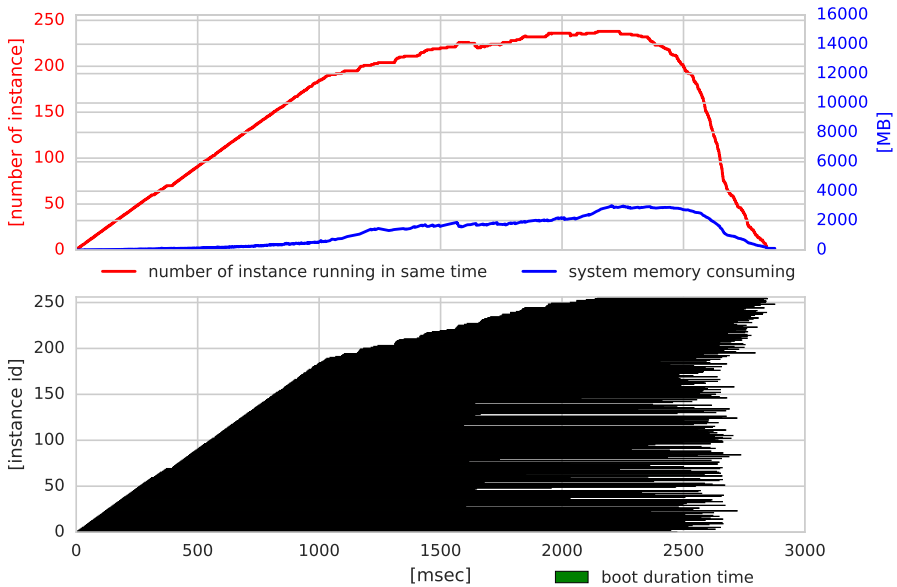


計測結果

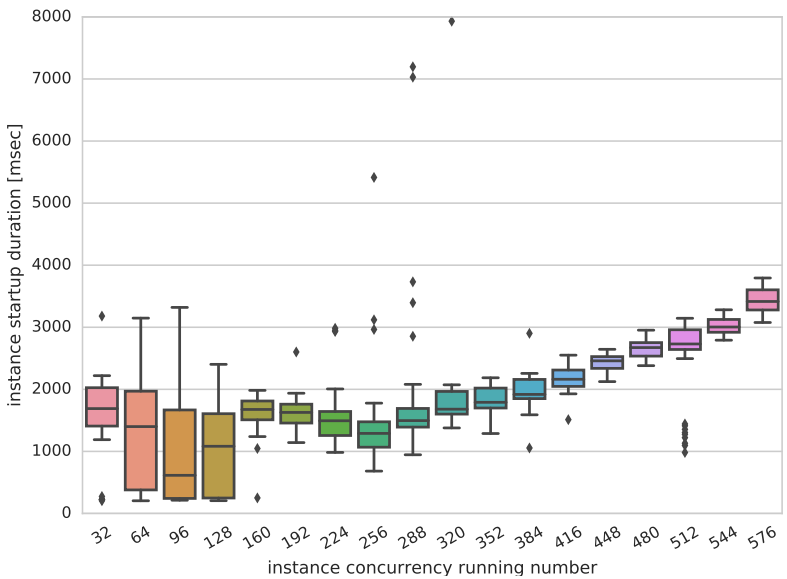
- インスタンスの起動にかかる時間，メモリの消費量はともに次の通りになった
  - Linux > LibOS > *minkernel*
- LibOS 同士の起動にかかる時間について大きな差が認められなかった
- 同時起動数に対しての起動にかかった時間の集計では LibOS ごとに特徴が表われていた

計測結果 - ベースライン（*minekernel*）

- 起動にかかる時間は最長 2.626 秒，最短 0.696 秒
- 最大消費メモリは 3020 MB

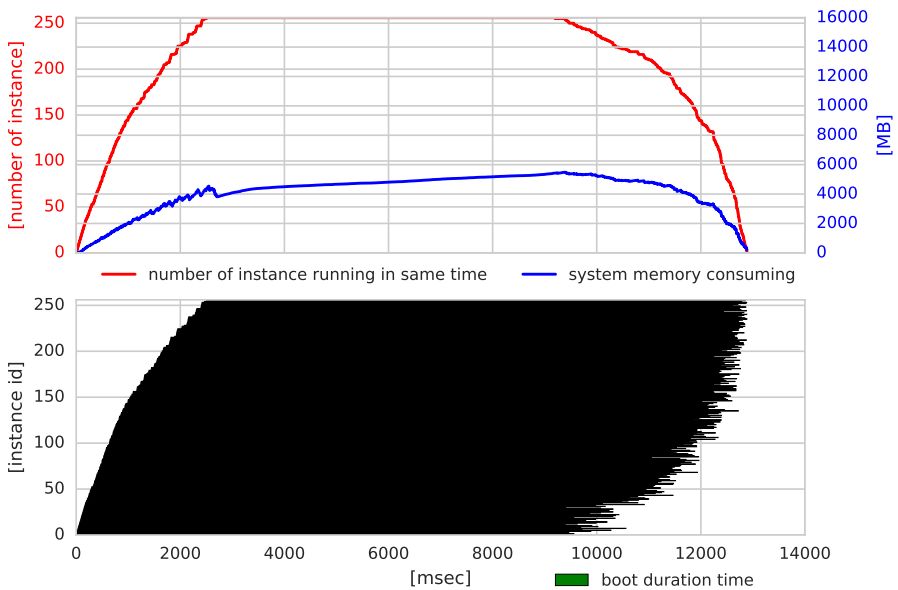


- 起動にかかる時間は小さく，同時起動台数に応じて起動にかかる時間が増えている

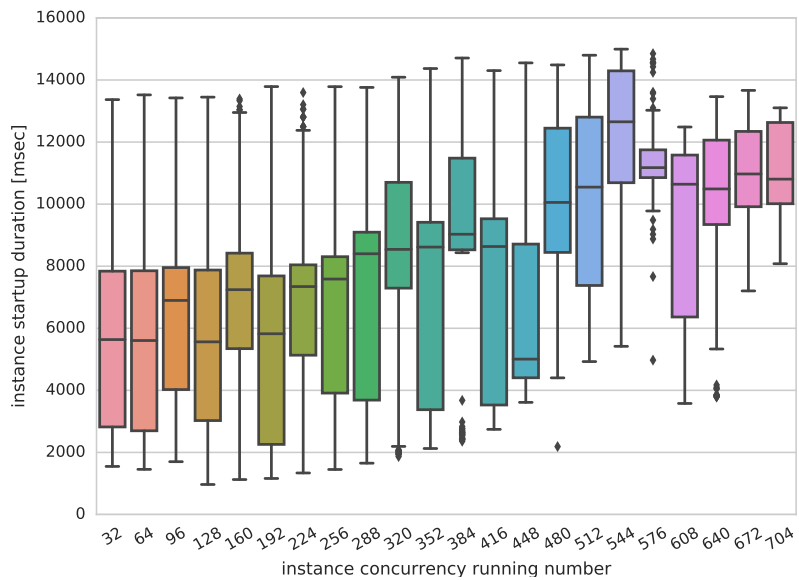


計測結果 - OSv

- 起動にかかる時間は最長 11.827 秒, 最短 9.119 秒
- 最大消費メモリは 5486 MB

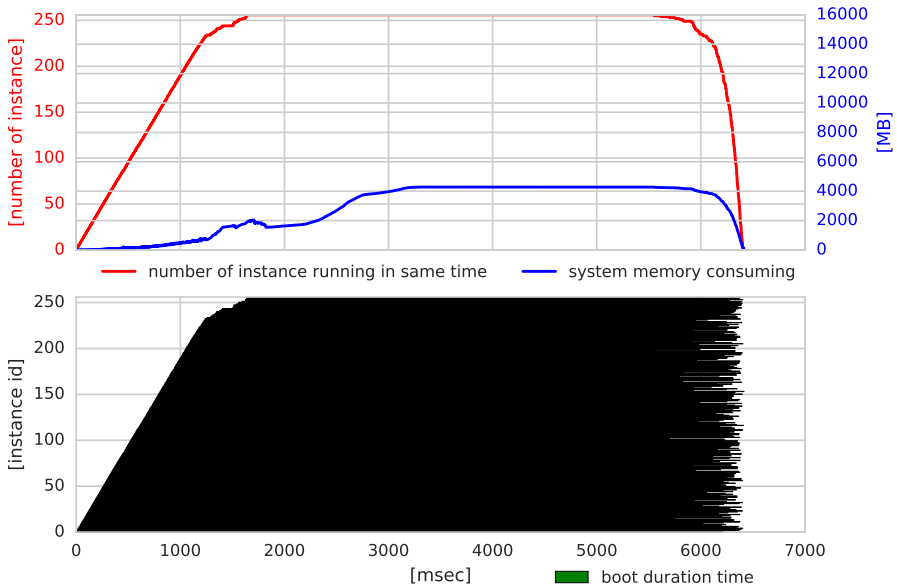


- 同時起動数に対して起動にかかる時間にバラつきがある
  - 同時起動数にあまり左右されていない

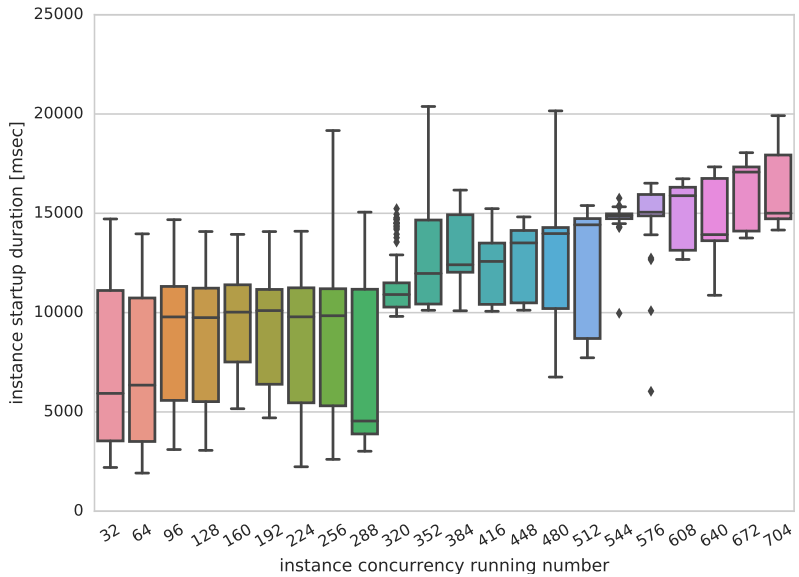


計測結果 - IncludeOS

- 起動にかかる時間は最長 6.380 秒, 最短 4.507 秒
- 最大消費メモリは 4281 MB

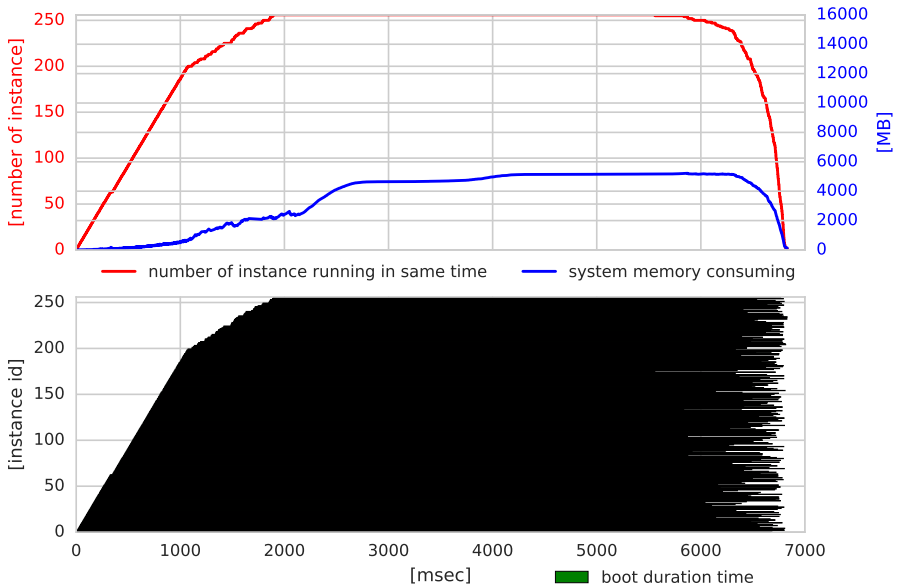


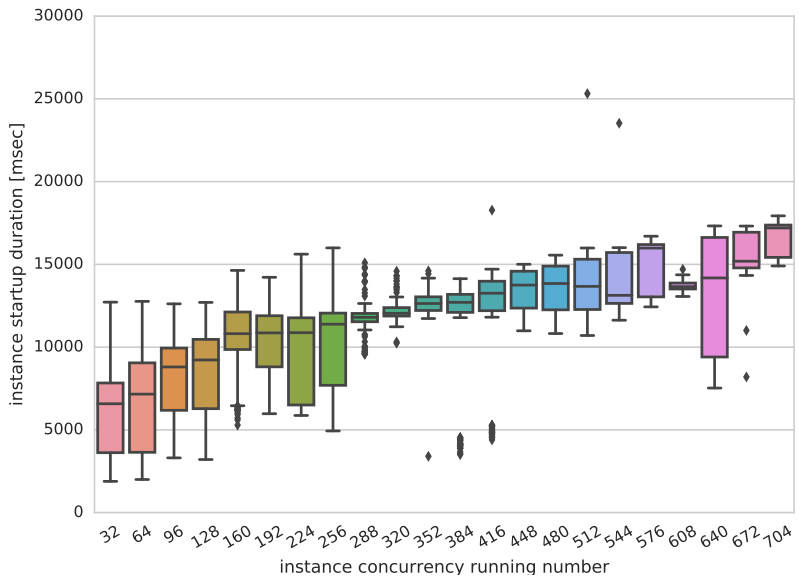
- OSvと違い同時起動数に対して起動にかかる時間が増えている



計測結果 - Rumprun

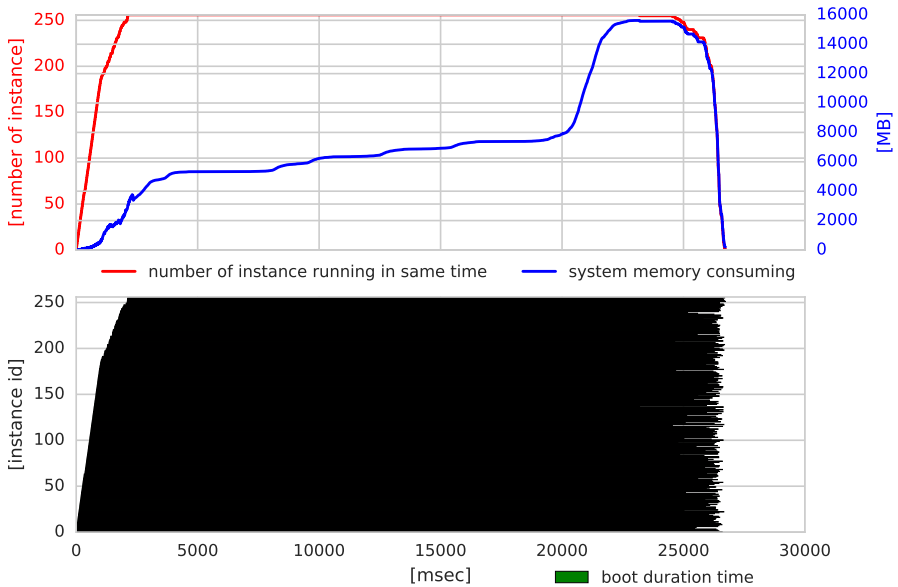
- 起動にかかる時間は最長 6.791 秒,
  最短 4.618 秒
- 最大消費メモリは 5220 MB



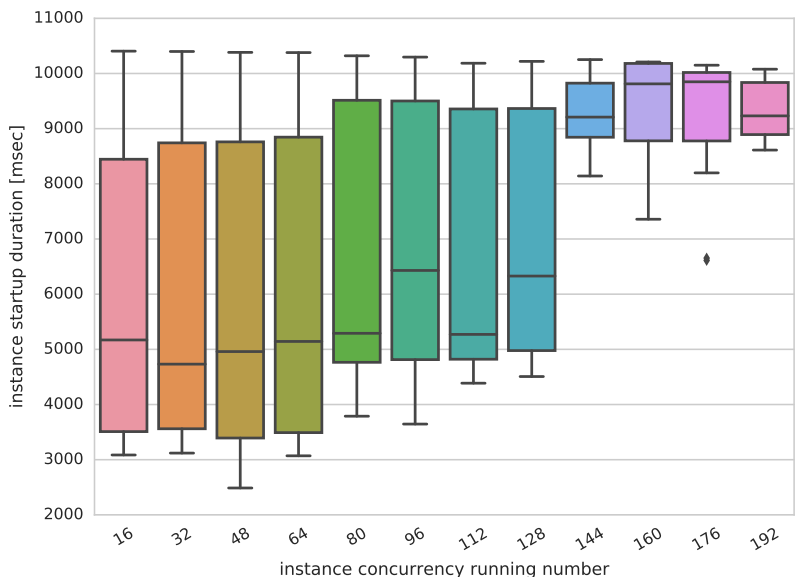


計測結果 - Linux

- 起動にかかる時間は最長 26.598 秒, 最短 22.441 秒
- 最大消費メモリは 15620 MB



- 256 台は起動したが 512 台起動した場合は計測プログラムが終了しなかった



計測結果

	起動最長時間	起動最小時間	システム最大消費メモリ
minkernel	2.626 sec	0.696 sec	3020 MB
OSv	11.827 sec	9.119 sec	5486 MB
IncludeOS	6.380 sec	4.507 sec	4281 MB
Rumprun	6.791 sec	4.618 sec	5220 MB
Alpine Linux	26.596 sec	22.441 sec	15620 MB

- また，300 台以上起動した場合，急激な性能の劣化が見られた

まとめと展望

- *minkernel*，LibOS，Linux での差は如実だが，LibOS 同士の違いの原因がまだ不明である
  - LibOS の何が起因して起動時間の違いがあるのかはっきりしていない
- さらなる調査が必要

参考文献 I

Bratterud, A., Walla, A.-A., Haugerud, H., Engelstad, P. E., Begnum, K., 2015. IncludeOS: A Minimal, Resource Efficient Unikernel for Cloud Services. In: 2015 IEEE 7th International Conference on Cloud Computing Technology and Science (CloudCom 2015). IEEE, Vancouver, BC, Canada, pp. 250–257.
URL http://dx.doi.org/10.1109/CloudCom.2015.89

Kantee, A., Cormack, J., 2014. Rump Kernels: No OS? No Problem! ;login: 39 (5), 11–17.
URL http://rumpkernel.org/misc/usenix-login-2014/login_1410_03_kantee.pdf

Kivity, A., Laor, D., Costa, G., Enberg, P., Har'El, N., Marti, D., Zolotarov, V., 2014. OSv—Optimizing the Operating System for Virtual Machines. In: 2014 USENIX Annual Technical Conference (ATC '14). USENIX Assocation, Philadelphia, CA, USA, pp. 61–72.
URL https://www.usenix.org/system/files/conference/atc14/atc14-paper-kivity.pdf

Madhavapeddy, A., Mortier, R., Rotsos, C., Scott, D., Singh, B., Gazagnaire, T., Smith, S., Hand, S., Crowcroft, J., 2013. Unikernels: Library Operating Systems for the Cloud. In: 18th ACM International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS '13). ACM, Houston, TX, USA, pp. 461–472.
URL http://dl.acm.org/citation.cfm?doid=2451116.2451167