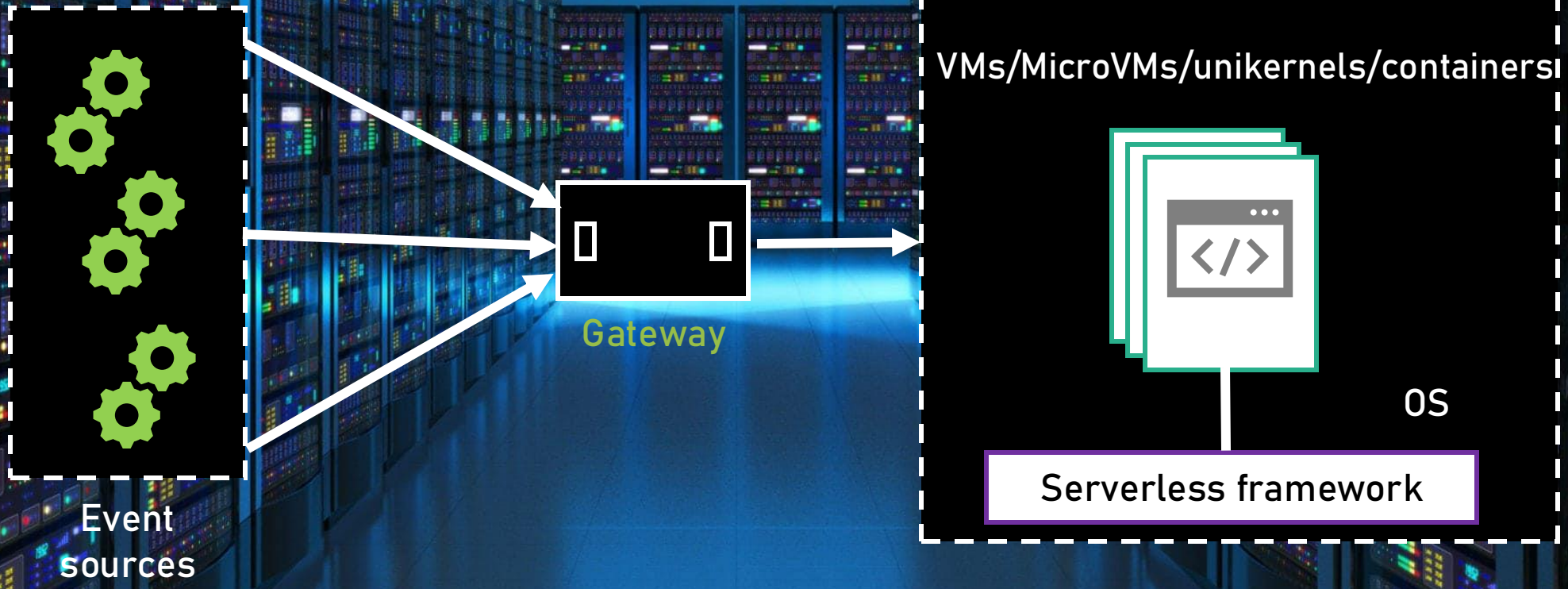
Performance challenges for FaaS workloads

Djob Mvondo

Associate Professor – University of Rennes

Context: Function As A Service

Developers send the code and configure the events/trigger



- Billed based on execution time and memory used.
- Focus on your code and leave the rest to the provider

Context: Function As A Service

Serverless cloud model is gaining a lot of traction.

~ 22 Billion $ estimated by 2025[1]

Amazon Lambda (microVMs)

Alibaba Function Compute (containers in VMs)

Azure Functions (not clear ???)

[1] https://www.alliedmarketresearch.com/serverless-architecture-market

Context: Function As A Service

Functions execute rapidly: hundreds of milliseconds (<1s) thus cold starts should be avoided

Mohammad Shahrad et al. Serverless in the Wild - Usenix ATC'20

Ao Wang et al. FaaSNet: Scalable and Fast Provisioning of Custom Serverless Container Runtimes at Alibaba Cloud Function Compute - Usenix ATC'21

Slides attacks – Containers vs VMs for FaaS

- CVE1…..1000
  - ❑ Escalation privilege
  - ❑ DOS
  - ❑ Crash system
- $exec_{func} \leq 500ms$ ➔ attack realisable ? Sous quelles contraintes ?
- Bonnes pratiques containers + Cloud

Attack 1	Yes / No	
Attack 2	Yes / No	How much time ? 10s ?

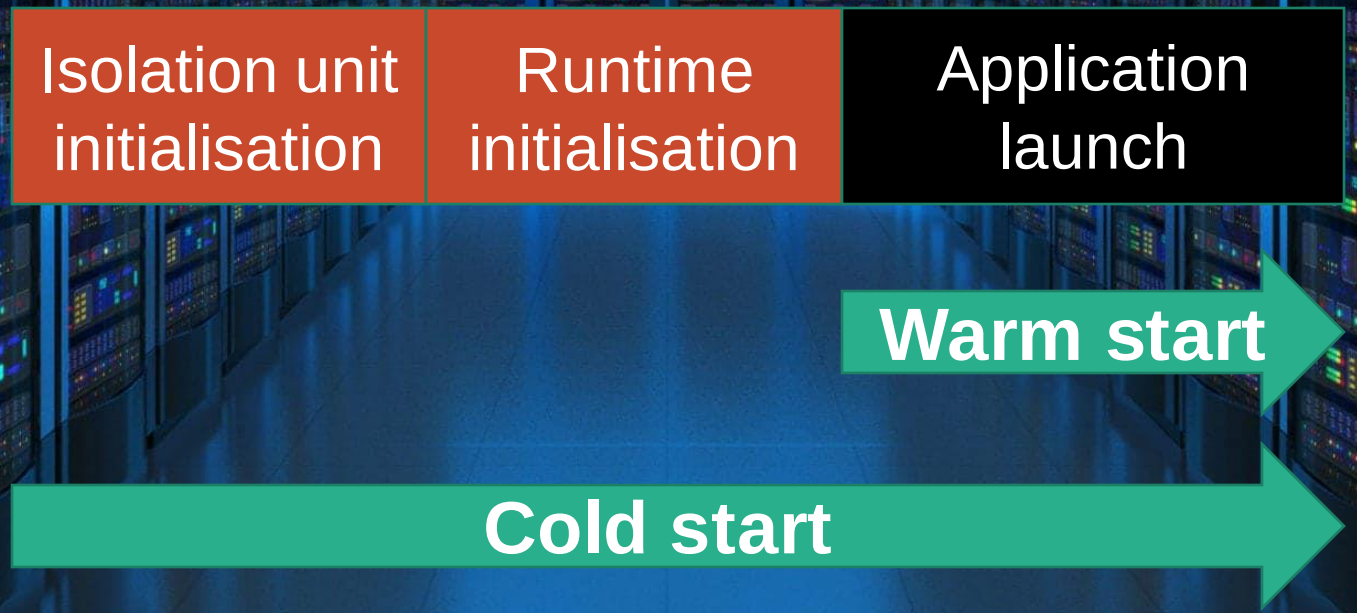
- Benchmark of attacks on containers (description, category, reproducable)
- Notation system
- Run/test container security in the context of FaaS (function to the control of the user, container/framework management, OS ➔ Cloud) > **30s**

- Eurosys 2024 | 19 octobre
- Usenix Security 2024 | 17 octobre
- Asplos 2024 | 30 Novembre

FaaS: speeding isolation units boot time

Reduce the boot time of isolation units + runtime initialization ???

- Checkpointing/restore techniques
- Specialization of isolation units



FaaS: speeding isolation units boot time

Reduce the boot time of isolation units + runtime initialization ???

- Checkpointing/restore techniques
- Specialization of isolation units

Potkemin-SOSP'05, SnowFlock-EUROSYS'09, ShadowReboot-VEE'13/DSN'11, Halite-ATC'13, Agamotto-USENIX SECURITY'20, Catalyzer-ASPLOS'20, FaasM-ATC'20, REAP-ASPLOS'21, FaaSnap-EUROSYS'22 AWS SnapStart

FaaS: speeding isolation units boot time

Reduce the boot time of isolation units + runtime initialization ???

- Checkpointing/restore techniques
- Specialization of isolation units

ClickOS-NSDI'14, OsV-ATC'14, LightVM-SOSP'17, Yolo-EUROPAR'19, Firecracker-EUROSYS'20, Unikraft-EUROSYS'21, FlexOS-HotOS'21/ASPLOS'22

Optimiser le démarrage des VMs

Utiliser des noyau optimisé pour chaque fonction
➔➔ **Unikernels**

Optimiser le démarrage des VMs

Technologie assez récente mais plusieurs projets par domaine

Mini-OS

IncludeOS

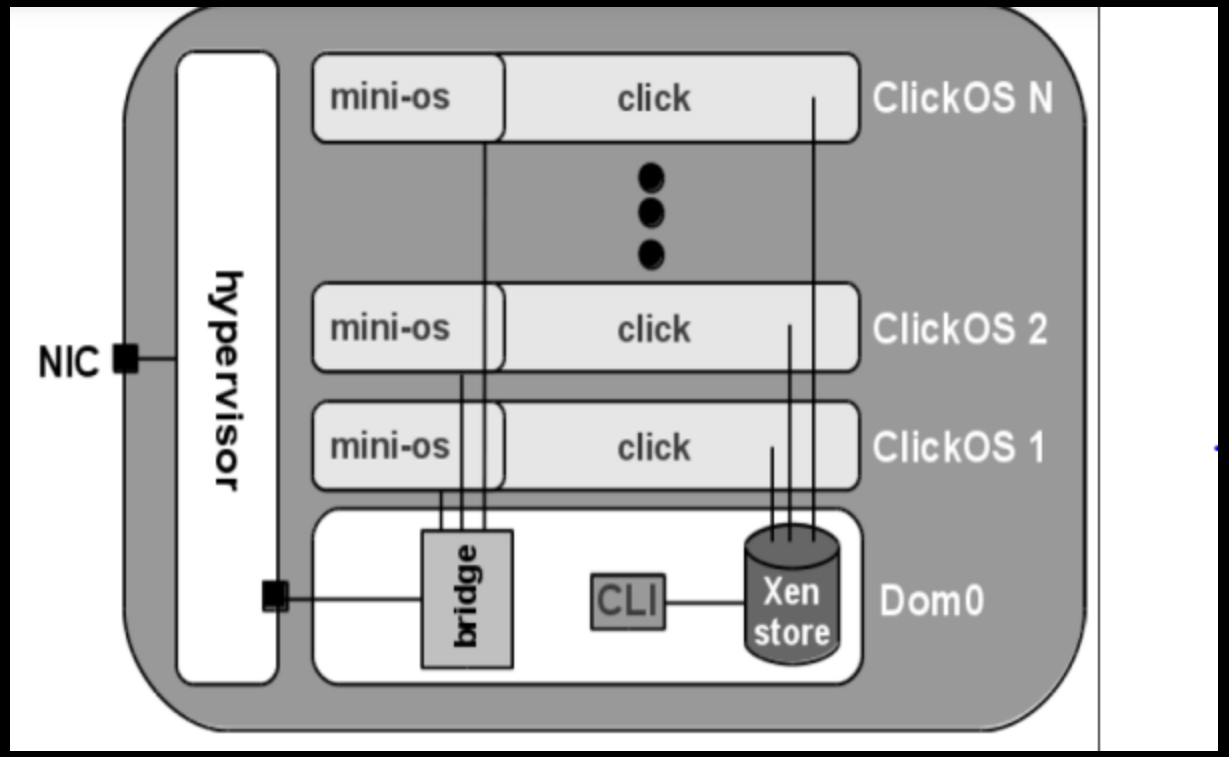
Unikraft (Toolchain)

Rumprun

ClickOS

Optimiser le démarrage des VMs

Cas pratique: ClickOS



Se base sur miniOS (5MB de taille)

Propose une optimisation des couches "bridge" et l'interface d'administration pour retirer les fonctionnalités inutiles

Près de 30x le temps de démarrage comparé à un OS classique.

Martins, J. et al. Enabling Fast, Dynamic Network Processing with ClickOS. HotSDN 2013.

Optimiser le démarrage des VMs

Cas pratique: ClickOS

Plusieurs alternatives ont emergé au fil des années, mais la logique reste la même est insiste sur la spécialisation.

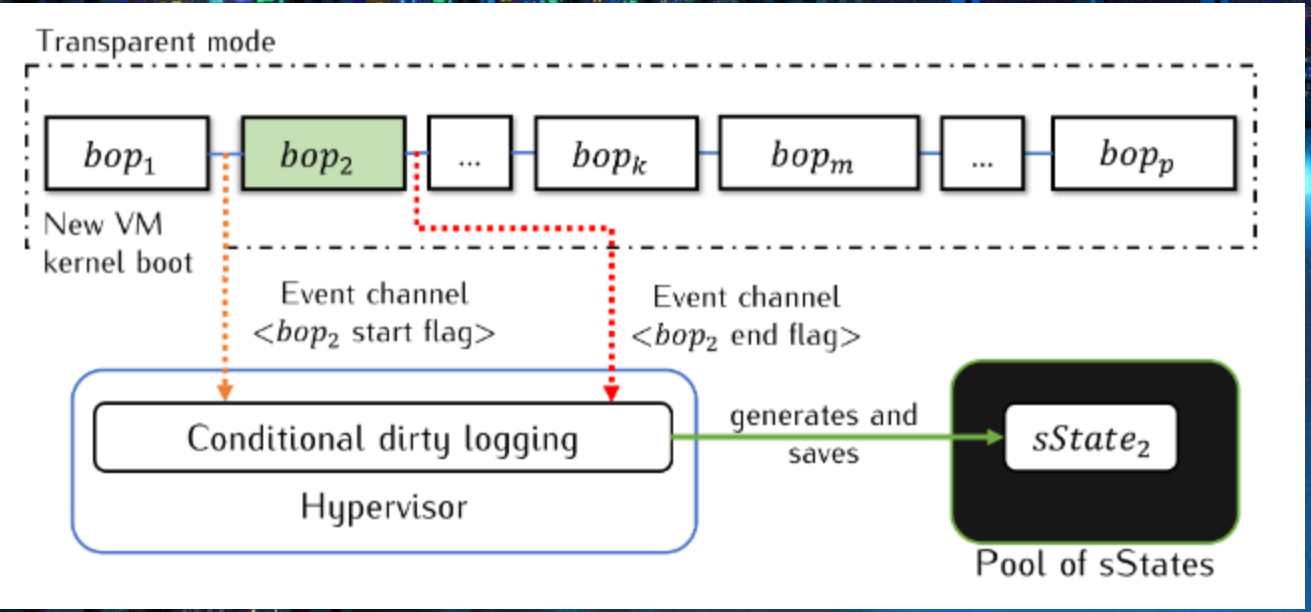
D'autres approches plus complexe insistent sur la généricité et peuvent aussi être exploité pour generer des images spécialisés mais facilement reconfigurable en fonction des besoins.

Unikraft: fast, specialized unikernels the easy way. Simon Kuenzer et al. EUROSYS'21
FlexOS: towards flexible OS isolation. Hugo Lefeuvre et al. ASPLOS'22

FaaS: speeding isolation units boot time

RockStar (Rocket Start):

- Leverage all the boots performed in the Cloud
- Construct a pool of sub-boot phases to reuse while ensuring coherence




```
Input: vmConf, sStatePool
/* input configuration of the VM and the pool of
   sStates                                          */
/* The gray steps are performed by the hypervisor,
   otherwise the booting VM.                        */
2  reUsables = findSimilar(sStatePool, vmConf)
13 updateStartInfoPage(reUsables)
4  for each bop during boot do
5      if !reUsable(bop) then
           /* nothing, the VM continues its boot as
              normal                                */
6      else
7          notifyHypervisor(bop,LOAD_PHASE)
9          pauseVM()
10         loadsState(reUsables,bop)
81         resumeVM()
           /* Resume VM and setting the resume
              instruction                           */
```

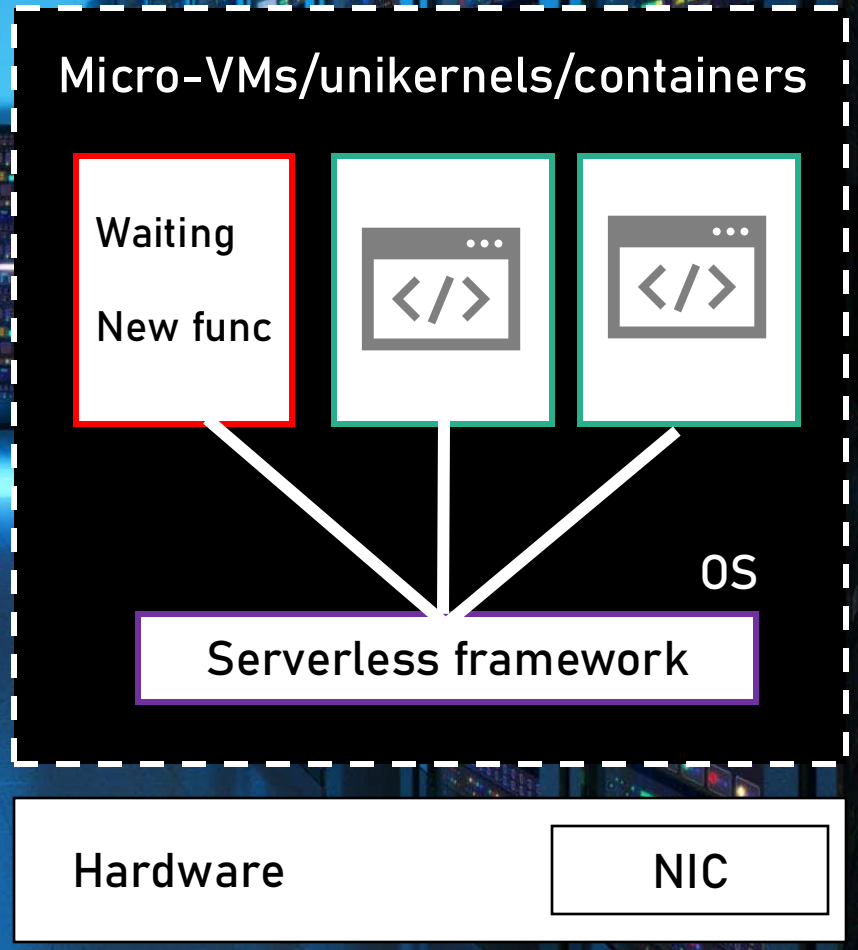
FaaS: the wrath of the idle

Reducing boot time is not enough !!!

Maximizing warm starts is a must

Keep-alive (10s-20s)

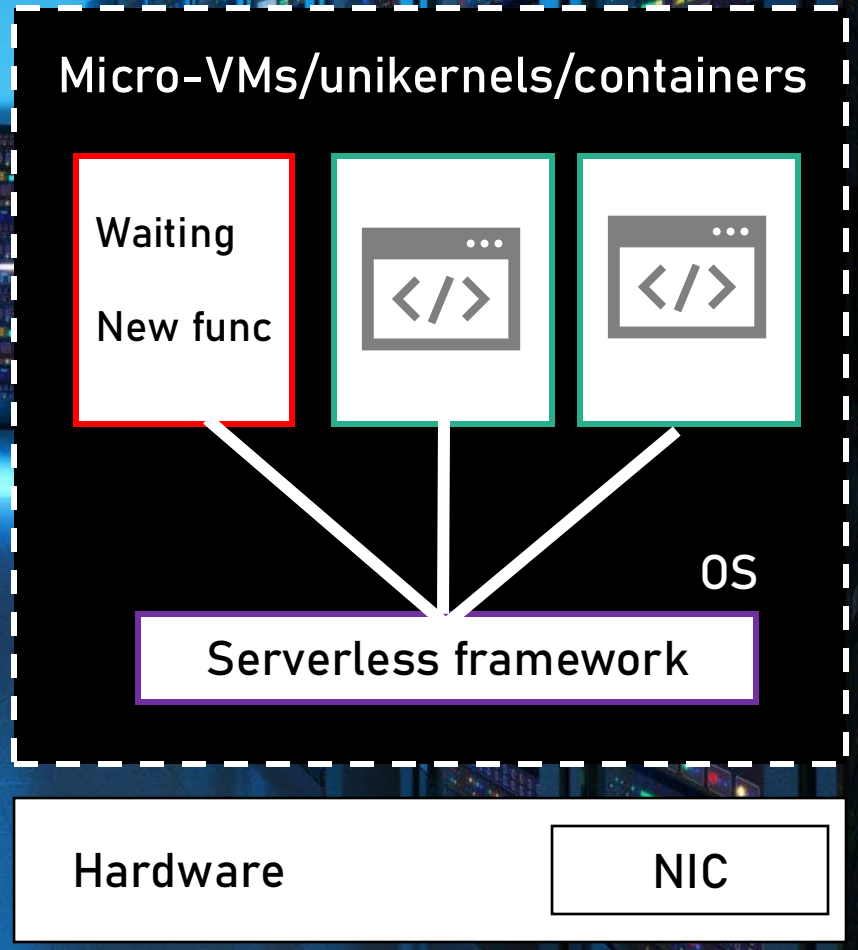
Premium offerings e.g., AWS provisioned concurrency or Azure Premium Functions



FaaS: the wrath of the idle

These strategies to maximize warm starts lead to several idle isolation units alongside active ones

Djob et al. Tell me when you are sleepy and what may wake you up! SOCC'21



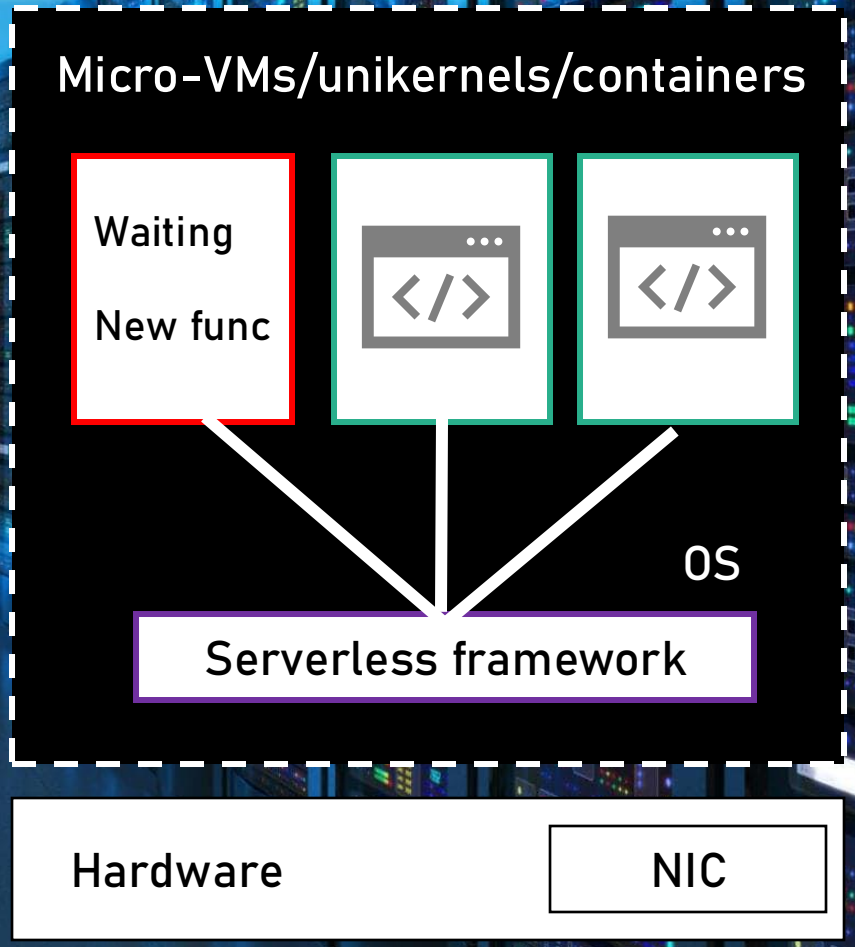
FaaS: the wrath of the idle

42%

Power usage of idle **100 isolation units** compared to **100 active isolation units –** schedutil governor

Can we consider idle functions to achieve energy savings without impacting performance?

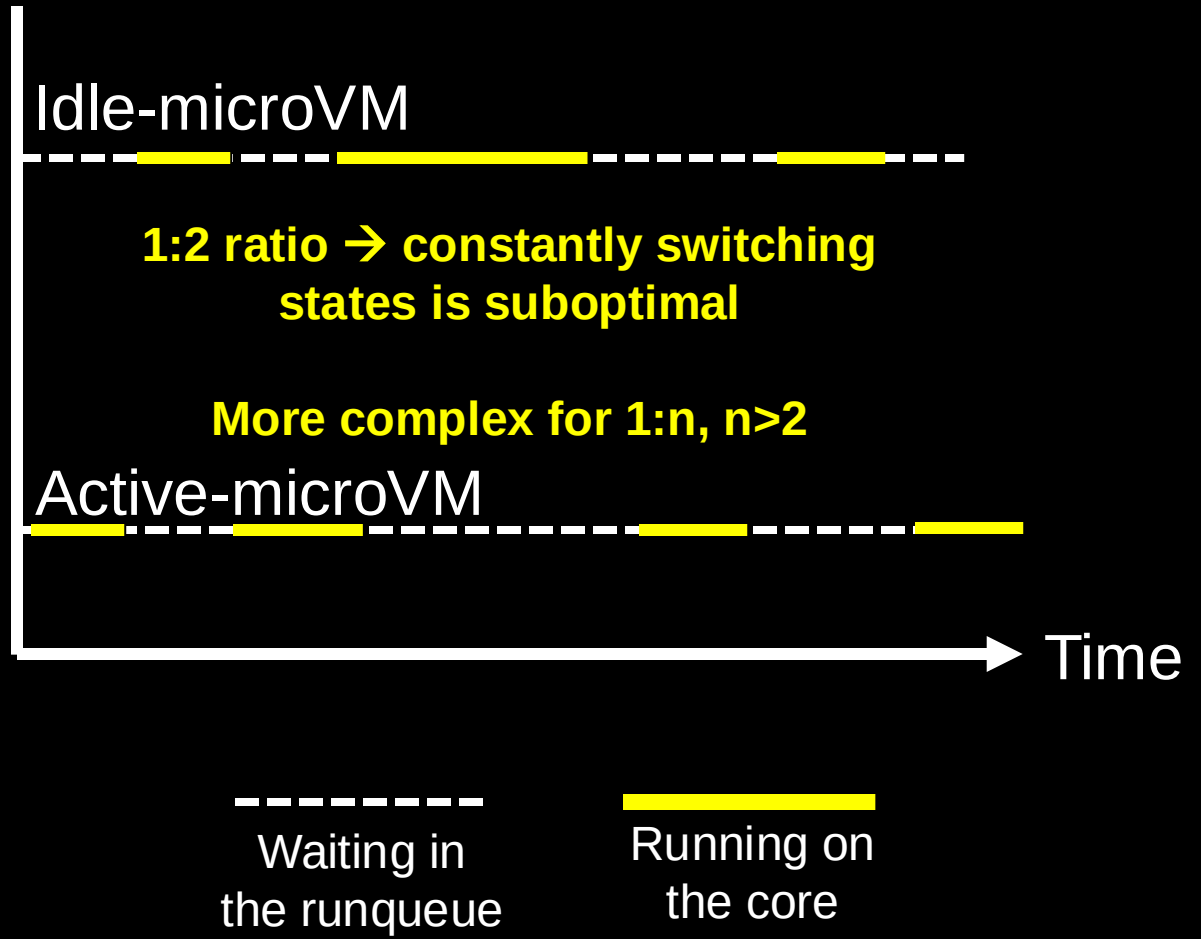
Google Study (Barroso)-ISCA'07, Quasar-ASPLOS'14, Pegasus-ISCA'14, Rubik-MICRO'15, E3-ATC'19, FaaSNet-ATC'21, Alibaba Trace Analysis-SOCC'21



FaaS: the wrath of the idle

Issues with DVFS for FaaS:

- Several isolation units vCPU on one core
- Latency of switching frequency states and energy drawn (Turbo-ATC'14,AgilePkg-MICRO'23) ~ 100th of μs
- Functions' run in milliseconds' scale (should run at the highest frequency if possible)



Scheduling for improved energy usage ratio

Research directions (not tuned for FaaS):

- Faster frequency states switching?

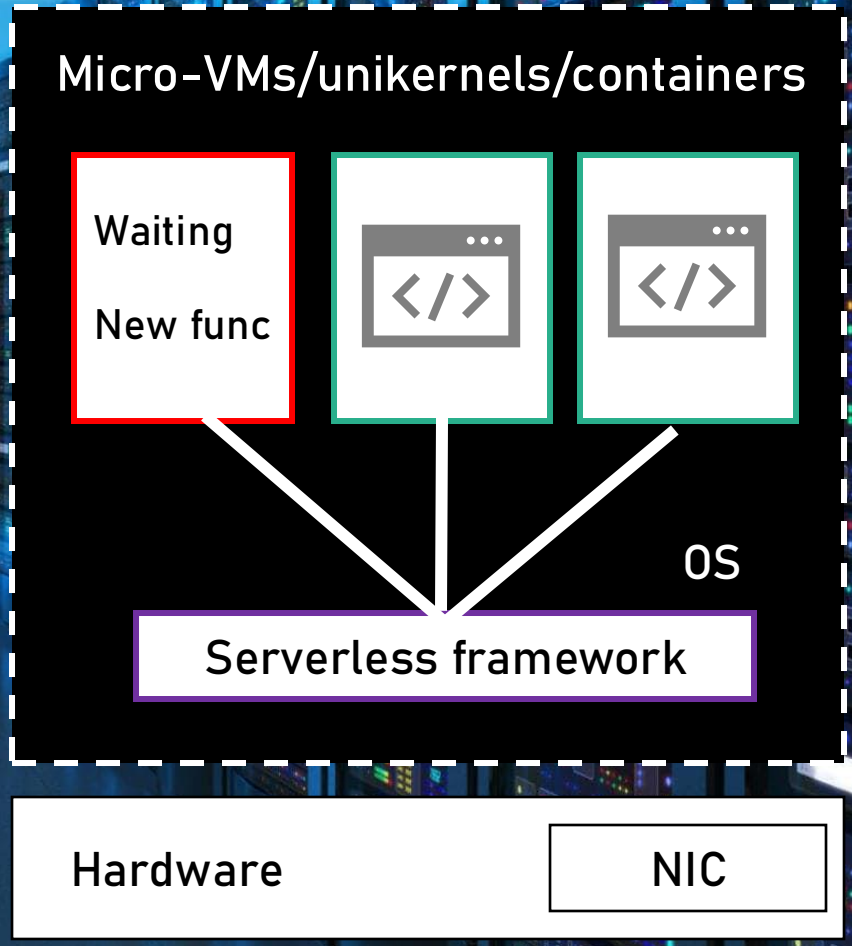
  AgilePkg-MICRO'23, AgileWatts-MICRO'23

- Predict inter-functions' arrival?

  Pegasus-ISCA'14, IdlePower-MMCS'08, Yawn-ApSys'19

- **Scheduling approaches?**

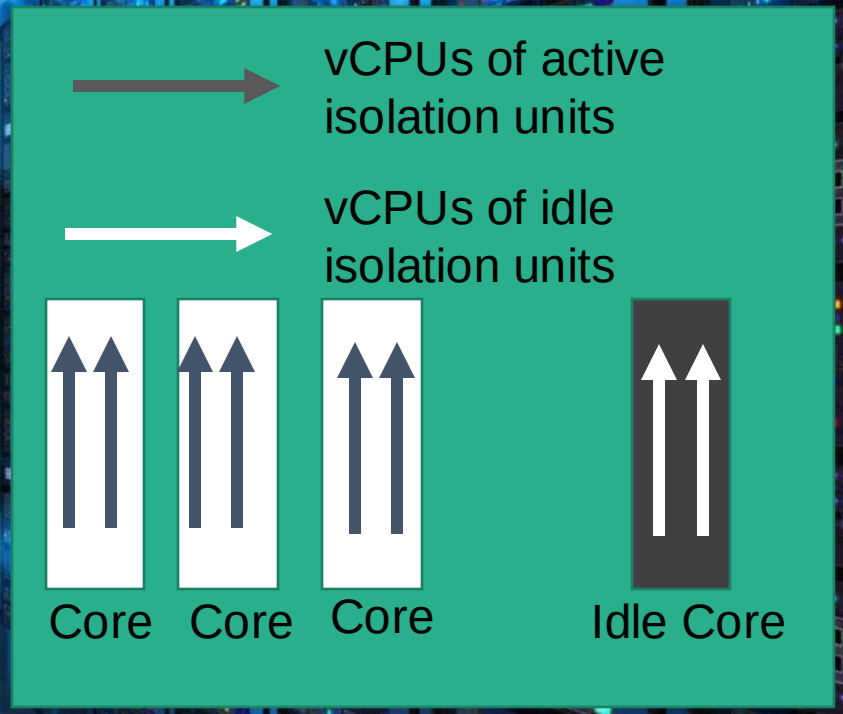
  Peafowl-SoCC'21, Djob et al. SoCC'21



DMX: Scheduling for improved energy usage ratio in FaaS

Early idea → DMX: Detect Move eXecute

- ❖ Detecting isolation units without peeking into isolation units
- ❖ Move idle isolation units vCPUs to a core that constantly remains at the lowest running state possible
- ❖ Move back the vCPUs to other cores upon trigger isolation unit requested for execution



WIP: Still figuring out the design (similar to vTurbo-ATC'13 for I/O)

- ➢ Cost of migration?
- ➢ Impact of idle isolation units running on the idle core e.g., clock synchronization?
- ➢ Event to monitor that triggers execution?

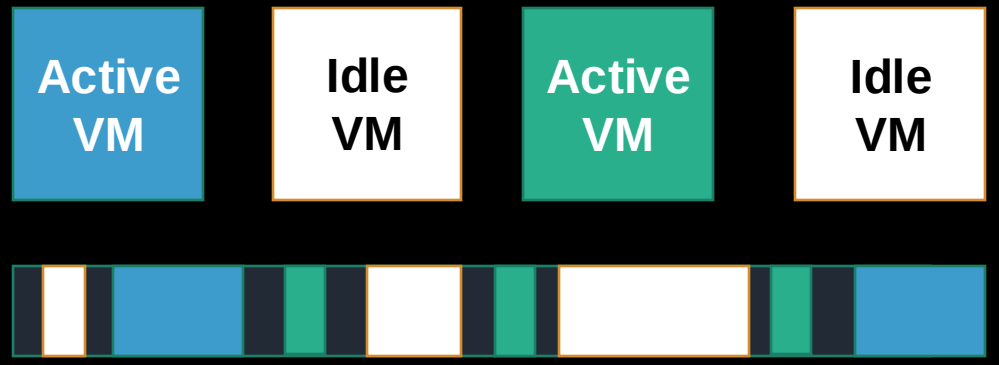
FIMeR: Idle memory footprint reduction for FaaS

Reducing boot time is not enough !!!

Maximizing warm starts is a must

Keep-alive (10s-20s)

Premium offerings e.g., AWS provisioned concurrency or Azure Premium Functions



Memory

Idle instances are paused, thus they still hold memory.

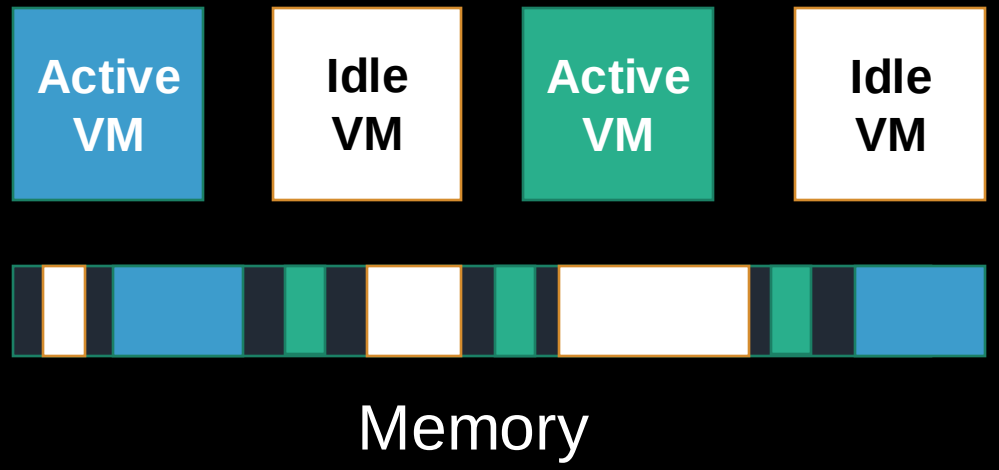
Up to **25%** of instances remain idle 50% of their lifetime. **Owl-SoCC'22**

FIMeR: Idle memory footprint reduction for FaaS

Idle instances are paused, thus they still hold memory

Existing: Predicting functions' memory usage to detect overprovisioning – **Owl-SoCC'22, OFC-Eurosys'22, Sizeless-Middleware'21**

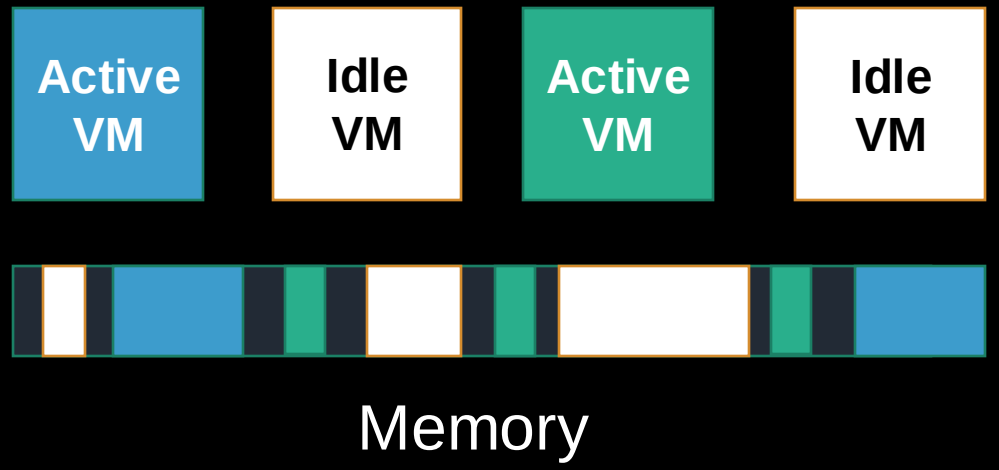
- **No profiling can be done on idle instances since they are not running**
- **Incorrect shrinking can increase resume times**



FIMeR: Idle memory footprint reduction for FaaS

Ideas to reduce idle memory footprint

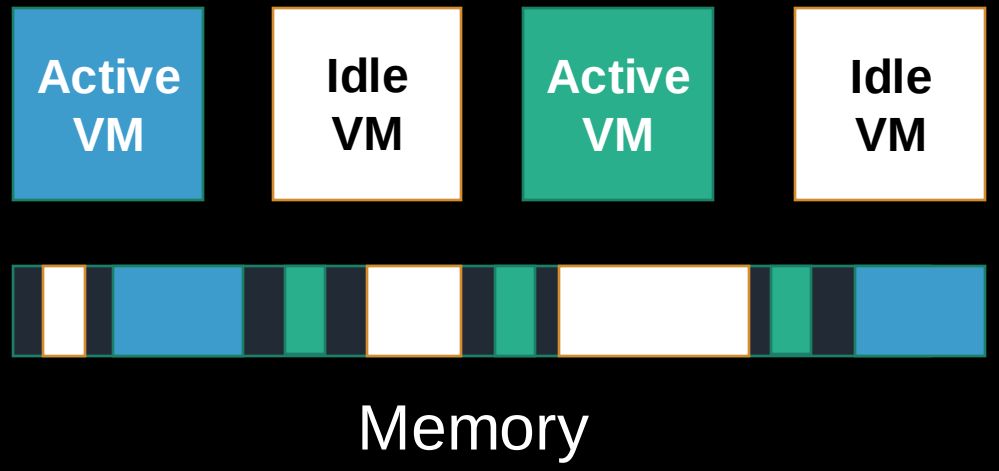
- **Fragmenting memory** of idle instances**?**
- **Detecting unused and dirty pages** (memory pages with no content)?
- **Merging similar memory regions** of idle instances?



FIMeR: Idle memory footprint reduction for FaaS

Challenges regarding idle instances' memory footprint reduction

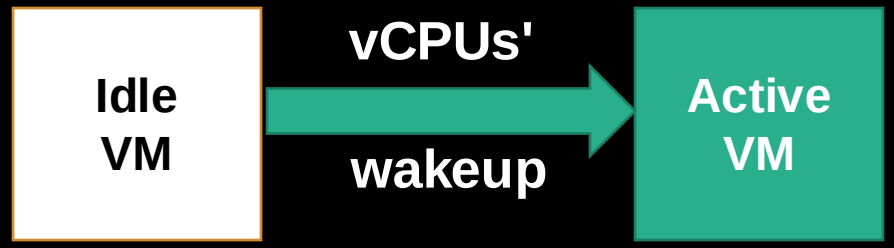
- Can we achieve that at the hypervisor level? Or should we **modify the runtime allocator?**
- What of **isolation/security for merging**?
- How to **shrink and expand memory** without impacting resume times?



PaR: Parallel resume for idle FaaS instances

Idle instances are paused, what of resume time ?

- Wakeup is **sequential** and depends on the **number of vCPUs**
  - To distribute the vCPUs on several cores
- FaaS instances can go up to 32 vCPUs *(Alibaba Function Compute and AWS Lambda)*
- Parallel resume? **Impact on other functions?**

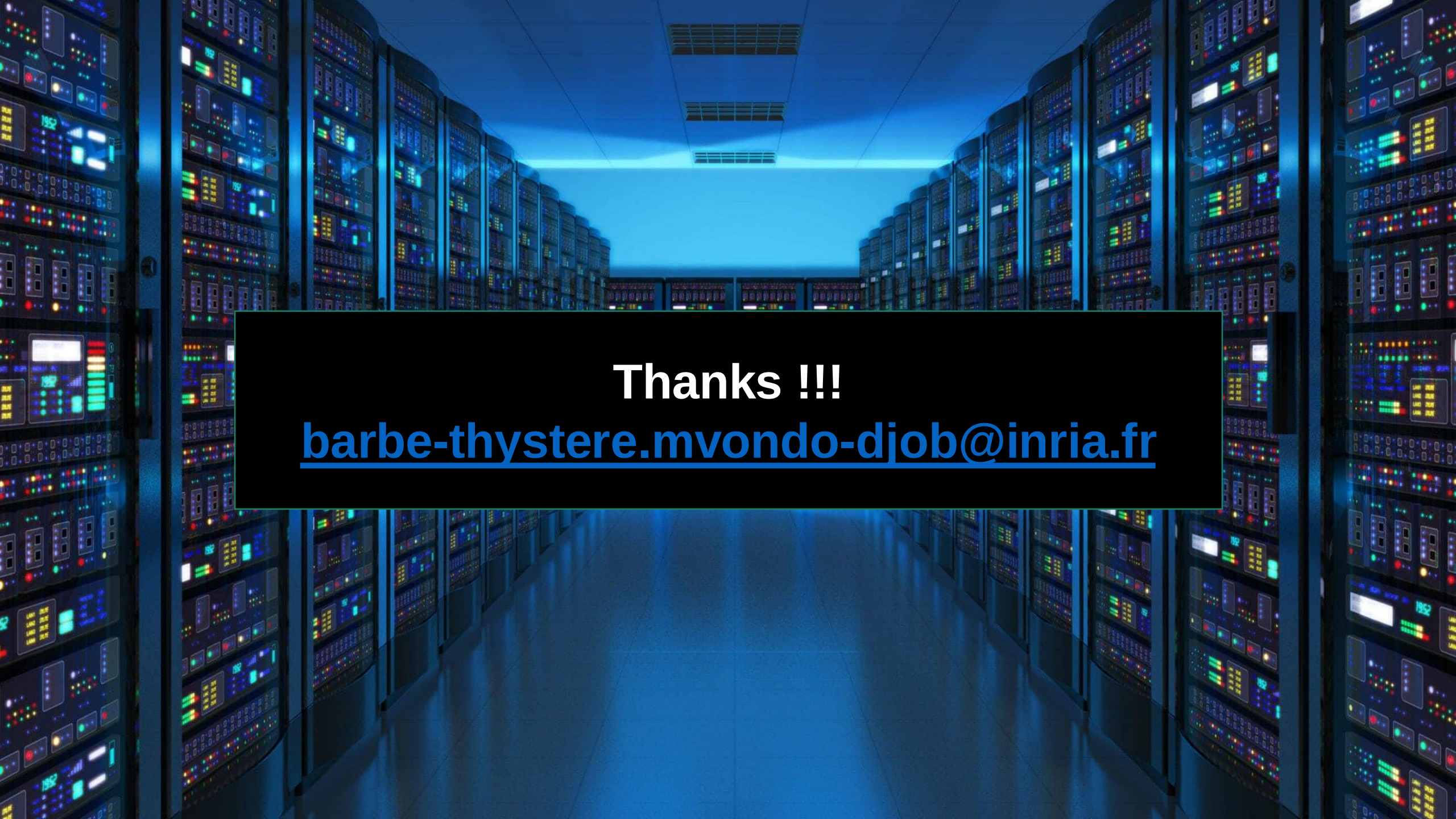



```
void domain_unpause(struct domain *d)
{
    struct vcpu *v;

    arch_domain_unpause(d);

    if ( atomic_dec_and_test(&d->pause_count) )
        for_each_vcpu( d, v )
            vcpu_wake(v);
}
```

common/xen/domain_unpause:1237


Thanks !!!
barbe-thystere.mvondo-djob@inria.fr