

# DAMON Updates and Future Plans:

Automation of DAMON tuning, tiering, and VM guest scaling

SeongJae Park <sj@kernel.org>

<https://damonitor.github.io>

# Notices

- The views expressed herein are those of the speaker; they do not reflect the views of his employers
- Slides are available at <https://github.com/damonitor/talks> or below QR code



Scan to  
download these  
slides

QR code is generated by <https://qr.io/>

From: SeongJae Park <[sj@kernel.org](mailto:sj@kernel.org)>

- Just call me “SJ” (easier to be consistently pronounced)
- Kernel Development Engineer at AWS
- Interested in the memory management and the parallel programming
- Maintaining [DAMON](#) ([mm/daemon/](#))

# Overview

- DAMON in a Nutshell (2 min)
- Updates since LSFMM+BPF 2023 (5 mins)
  - Misc Things: Documentation, selftests, filters
  - DAMOS Auto-tuning
- Major Future Plans
  - Tiered Memory Management (5 mins)
  - Access/Contiguity-aware Memory Auto-scaling (5 mins)
  - Misc Things: LRU\_SORT auto-tuning, THP, monitoring improvement (3 mins)
- Discussions (10 mins)

# DAMON in a Nutshell

# DAMON: Access Pattern Snapshot Generator

- Inform which *address range* is how *frequently* accessed for how *long* time

```
|000000000000000000000000000000000000000000000| size 31.219 MiB access rate 0 % age 2 m 46.500 s
|000000000000000000000000000000000000000000000| size 31.426 MiB access rate 0 % age 3 m 47.200 s
|000000000000000000000000000000000000000000000| size 31.422 MiB access rate 0 % age 3 m 49.300 s
|000000000000000000000000000000000000000000000| size 31.316 MiB access rate 0 % age 3 m 49.600 s
|000000000000000000000000000000000000000000000| size 31.273 MiB access rate 0 % age 3 m 47.400 s
|000000000000000000000000000000000000000000000| size 31.379 MiB access rate 0 % age 3 m 34.700 s
|000000000000000000000000000000000000000000000| size 31.449 MiB access rate 0 % age 45.800 s
|000000000000000000000000000000000000000000000| size 31.438 MiB access rate 0 % age 27.300 s
|000000000000000000000000000000000000000000000| size 31.391 MiB access rate 0 % age 9.300 s
|000000000000000000000000000000000000000000000| size 6.000 MiB access rate 0 % age 2.400 s
|000000000000000000000000000000000000000000000| size 8.000 KiB access rate 55 % age 0 ns
|999999999999999999999999999999999999999999999| size 9.531 MiB access rate 100 % age 1.900 s
|444444444444444444444444444444444444444444444| size 8.000 KiB access rate 45 % age 300 ms
|000000000000000000000000000000000000000000000| size 9.660 MiB access rate 0 % age 2.300 s
|000000000000000000000000000000000000000000000| size 6.949 MiB access rate 0 % age 3 m 21.300 s
|000000000000000000000000000000000000000000000| size 120.000 KiB access rate 0 % age 3 m 50 s
|444444444444444444444444444444444444444444444| size 8.000 KiB access rate 55 % age 300 ms
|000000000000000000000000000000000000000000000| size 4.000 KiB access rate 0 % age 3 m 49.700 s
total size: 314.598 MiB
```



# DAMOS: DAMON-based Operation Scheme

- Apply memory operation actions to regions of interesting access pattern

```
# # pageout memory regions that not accessed for >=5 seconds
```

```
# damo start --damos_action pageout --damos_access_rate 0% 0% --damos_age 5s max
```

Pageout!

Size	Access Rate	Age
31.219 MiB	0 %	2 m 46.500 s
31.426 MiB	0 %	3 m 47.200 s
31.422 MiB	0 %	3 m 49.300 s
31.316 MiB	0 %	3 m 49.600 s
31.273 MiB	0 %	3 m 47.400 s
31.379 MiB	0 %	3 m 34.700 s
31.449 MiB	0 %	45.800 s
31.438 MiB	0 %	27.300 s
31.391 MiB	0 %	9.300 s
6.000 MiB	0 %	2.400 s
8.000 KiB	55 %	0 ns
9.531 MiB	100 %	1.900 s
8.000 KiB	45 %	300 ms
9.660 MiB	0 %	2.300 s
6.949 MiB	0 %	3 m 21.300 s
120.000 KiB	0 %	3 m 50 s
8.000 KiB	55 %	300 ms
4.000 KiB	0 %	3 m 49.700 s

total size: 314.598 MiB



## Features for Product Quality DAMOS Control

- *“One does not simply control DAMOS with only access pattern”*
- Quota: set aggressiveness of DAMOS
  - e.g., pageout cold pages up to 100 MiB per second (coldest 100 MiB pages)
- Filters: define target regions with non-access-pattern information
  - e.g., pageout cold pages of NUMA node 1 that associated with cgroup “A” and file-backed

# Usages, To One's Best Knowledge

- Products
  - Proactive memory reclamation for memory overcommit systems
  - CXL-based tiered memory management software development [kit](#)
- Researches
  - DAMON paper got 20 [citations](#)
- [Distros](#) having CONFIG\_DAMON=y
  - Amazon Linux (>=5.4), Android (>=5.10), CentOS (>=4.18), Fedora (>=6.2), UEK (>=5.15)
- Package [repos](#) providing DAMON user-space [tool](#)
  - AUR, Debian, EPEL, Fedora, Kali, PyPI, Raspian, Ubuntu

## Community

- “*Strive to be Earth’s best community*”
- Public mailing [list](#) ([lore.kernel.org](#)-archived)
- Bi-weekly meetup [series](#) (on Google Meet in usual)
  - Occasional/regular private meetings on demand
- Project [website](#) (<https://damonitor.github.io>)
  - Starting point for DAMON users and developers
  - Daily performance test results [archive](#)

Scan to visit  
the project  
website



QR code is generated by  
<https://qr.io/>

# DAMON Updates

## Answer to LSFMM 2023 Feedbacks

- “Some good documentation would be appreciated”
  - Improving Documentation/.../damon/
- “Adding DAMON user-space tool in-tree sounds not a good idea”
  - The goal was test coverage and easy DAMON interface understanding
  - Implementing DAMON functionality selftests

```
v6.5-rc1
  Patch series "Docs/{mm,admin-guide}damon: update design and usage docs".
  Patch series "Docs/mm/damon: Minor fixes and design doc update".
v6.8-rc1
  Patch series "selftests/damon: add Python-written DAMON functionality
v6.9-rc1
  Patch series "Docs/mm/damon: misc readability improvements".
  Patch series "selftests/damon: add more tests for core functionalities and
```

## More Features

- New filter types
  - “address range”: Specific NUMA nodes or zones
  - “young page”: Per-page access double check
- Fast snapshot generation
  - Generate per “aggregation interval” -> “sampling interval”
  - Faster monitoring, faster DAMOS with large aggregation interval

```
v6.6-rc1
Patch series "Extend DAMOS filters for address ranges and DAMON monitoring
v6.7-rc1
Patch series "mm/daemon: implement DAMOS apply intervals".
Patch series "mm/daemon: provide pseudo-moving sum based access rate".
```

# Aim-oriented Feedback-driven DAMOS Aggressiveness Auto-tuning

- *“One does not manually control DAMOS”*
- Auto-tune effective DAMOS quota using a proportional feedback loop
- Arbitrary value can be fed by user
  - E.g., Main workload's latency as feedback score
- System metrics based target can be set so DAMOS self-feed/auto-tune
  - Supporting memory PSI-based target (will add more target metrics)
  - *“Reclaim cold pages aiming 0.1% memory pressure stall rate”*

$$f(n) = \max(f(n - 1) * ((\text{target\_score} - \text{current\_score}) / \text{target\_score} + 1), 1)$$

```
v6.8-rc1  
Patch series "mm/damon: let users feed and tame/auto-tune DAMOS".  
v6.9-rc1  
Patch series "mm/damon: let DAMOS feeds and tame/auto-tune itself".
```

# DAMON Future Plans



# DAMOS Auto-tuning Based Tiered Memory Management

<https://lore.kernel.org/damon/20231112195602.61525-1-sj@kernel.org/>

## Existing DAMON-based Tiered Memory Management Approaches

- Tiered memory demotion (Alibaba)
  - [Patchset](#) is available (not yet merged; no updates for last 2 years)
- Two-tier memory promotion/demotion (HMSDK [v2](#), SK hynix)
  - [Patchset](#) is available (actively working, merged in `damon/next` tree)
  - Motivated ‘young page’ type DAMOS filter
- [MTM](#): Multi-Tiered Memory Management (Jie Ren et al., Eurosys’24)
  - Proposing monitoring improvement and fast migration node decision
- Patches implement only mechanisms, not the policy
  - HMSDK v2 open-source their [policy](#)

# DAMOS-based Tiered Memory Management Policy Proposal

- For each CPU-independent NUMA node,
  - If the node has a lower node,
    - Demote cold pages of the current node to the lower node, aiming little fraction (e.g. 5%) of free memory of the current node
  - If the node has an upper node,
    - Promote hot pages of the current node to the upper node, aiming big fraction (e.g., 96%) of used memory of the `_upper_` node

```
node 0 (fast)  Demote cold pages in node 0 aiming 5% free memory of node 0
node 1 (slow)  Promote hot pages in node 1 aiming 96% used memory of node 0
                Demote cold pages in node 1 aiming 5% free memory of node 1
node 2 (slowoo)Promote hot pages in node 2 aiming 96% used memory of node 1
```

## Expectations, or Hopes

- High utilization of upper nodes, with more frequently accessed pages
- Low utilization of lower nodes, with less frequently accessed pages
- Keep slow but continuous promotion/demotion
  - Overlapping memory util/free goals
- Easy to be extended for multiple tiers

## Progress

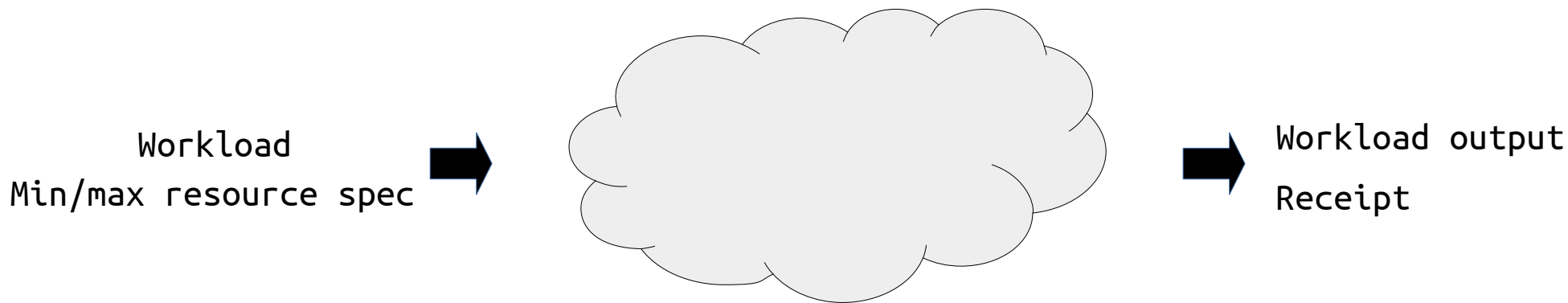
- Detailed RFC [idea](#) is sent to the mailing list
- No test setup, no implementation

# Access/Contiguity-aware Memory Auto-scaling (ACMA)

<https://lore.kernel.org/damon/20231112195114.61474-1-sj@kernel.org/>

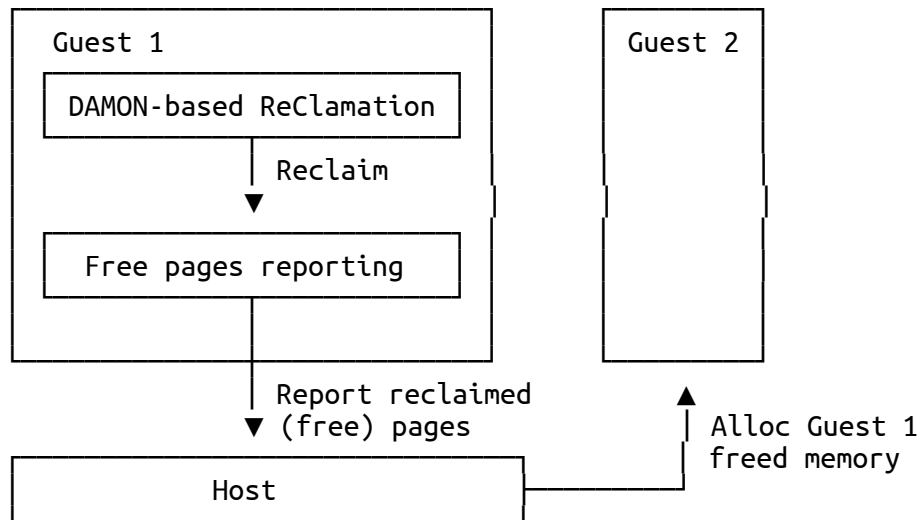
## Motive Business Model

- User: specify workload and min/max memory requirements
- Service Provider: run it somewhere, charge as they gone
  - Achieving high performance and low price is the provider's duty, and benefits both



## An Existing Approach: Orchestration of Four Kernel Features

- Collaborative overcommit (Free pages reporting)
- DAMON\_RECLAIM for reporting more pages without performance degradation
- Periodic compaction for reporting level contiguity
- Memory hot-[un]plugging for hard limit and 'struct page' reduction
- Works well in real world





## Limitations

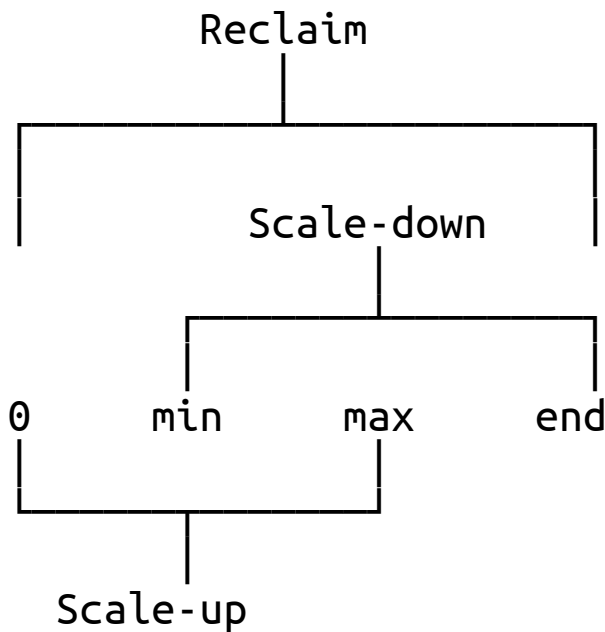
- Complexity of user-space driven multiple kernel features orchestration
- Memory hot-unplugging is slow and easy to fail
  - Due to coarse granularity and access obliviousness
- System-level compaction is wasteful and access oblivious
- Lack of after-report pages control
  - Any reported pages can be claimed again at any time
- Lack of non-collaborative guests control

# DAMOS Actions for Access-aware Contiguous Memory Allocation

- DAMOS\_ALLOC
  - Allocate given memory region with user-specified minimum contiguity
  - Notify (callback) the allocation to the user
  - “Repeatedly try to allocate cold memory regions, 2 MiB contig-regions at once”
- DAMOS\_FREE
  - De-allocate the region with user-specified minimum contiguity

## Access/Contiguity-aware Memory Auto-Scaling

- DAMON kernel module utilizing three DAMOS schemes
- Parameters: min-mem, max-mem, acceptable memory PSI
- Reclaim: Reclaim memory aiming “psi”
- Scale-down: ALLOC/report [min-mem, max) mem aiming “psi”
  - Auto-tune aggressiveness for higher PSI
  - Highest non-fully-DAMOS\_ALLOC-ed memory block only
  - Apply ‘struct page’ reduction in some level (like [HVO](#))
- Scale-up: FREE [0, max-mem) mem aiming “psi”
  - Auto-tune aggressiveness for lower PSI
  - Lowest partial-DAMOS\_ALLOC-ed memory block only



## Access-aware Ballooning: Control non-collaborative guests

- Adjust ACMA's max-mem parameter for balloon {in,de}flating
- Reuse virtio-balloon interface (no host-side change)

```
diff --git a/drivers/virtio/virtio_balloon.c b/drivers/virtio/virtio_balloon.c
[...]
@@ -472,6 +472,32 @@ static void virtballoon_changed(struct virtio_device *vdev)
    struct virtio_balloon *vb = vdev->priv;
    unsigned long flags;

+   #ifdef CONFIG_ACMA_BALLOON
+       s64 target;
+       u32 num_pages;
+
+       virtio_cread_le(vb->vdev, struct virtio_balloon_config, num_pages,
+                       &num_pages);
+       target = ALIGN(num_pages, VIRTIO_BALLOON_PAGES_PER_PAGE);
+       acma_set_max_mem_aggressive(totalram_pages() - target);
+       return;
+   #endif
+
    spin_lock_irqsave(&vb->stop_update_lock, flags);
    if (!vb->stop_update) {
        start_update_balloon_size(vb);
    }
}
```

## More Hopeful Usages of Access-aware Contiguous Memory Allocation

- Dynamic contiguous memory allocation pool allocation
- DRAM power saving
  - A variant of ACMA running on the bare metal
  - Do not report alloc-ed pages
  - Hot-unplug and power-off fully-alloc-ed memory blocks

## Progress

- Detailed design and partial pseudo-code level patchset will be available by the talk

## More Future Plans

- Monitoring improvements
  - Auto-tuning
  - higher accuracy
- Write-only monitoring
- LRU-sort auto-tuning
- Access-aware THP assistant
- CPU-aware monitoring and NUMA-balancing

## Discussion Time!

- ACMA
  - Is there existing alternatives for the motivation use case (memory over-commit VM systems)?
  - Ok to reuse pages reporting from ACMA?
  - Ok to reuse virtio-balloon's interface for Access-aware Ballooning?
  - Will access-aware migration make real improvement? Recommending test workloads?
  - Do DAMOS\_ALLOC-based dynamic CMA pool alloc and DRAM power saving make sense?
- Tiered-memory
  - Directly migrate to appropriate tier, instead of incremental bubbling up/down?
  - Any DAMON tuning failures from your tiering approach?
- Questions or comments on updated features and other future plans
- Don't forget [sj@kernel.org](mailto:sj@kernel.org), [damon@lists.linux.dev](mailto:damon@lists.linux.dev), and DAMON Beer/Coffee/Tea [Chat](#)



# Backup Slides

# DAMON Stack, In a Future

