Virtualization and Adaptive Provisioning of Non-volatile Memory

Bhavesh Singh
143059003

Guided by

Prof. Purushottam Kulkarni

Three Questions...

- What?
- Why?
- How?

What? Virtualization

- Creating virtual machines to consolidate multiple services on one physical server
- Efficient resource utilization through over-commitment
- Require isolation guarantees

What? Non-volatile Memory

- A class of storage devices that provide persistence
- Can be byte-addressable (memory bus) or block-addressable (I/O bus)
- Performance characteristics lie in between DRAM and magnetic hard disk

What? Performance Metrics of NVM [Mittal, TPDS 16]

	Access Granularity	Read Latency	Write Latency	Erase Latency	Endurance (# Writes)
HDD	512 B	5 ms	5 ms	N/A	$> 10^{15}$
SLC Flash	4 KB	25 ms	500 ms	2 ms	$10^4 – 10^5$
PCM	64 B	50 ns	500 ns	N/A	$10^8 – 10^9$
STT-RAM	64 B	10 ns	50 ns	N/A	$> 10^{15}$
ReRAM	64 B	10 ns	50 ns	N/A	10^{11}
DRAM	64 B	50 ns	50 ns	N/A	$> 10^{15}$

What? Two Important Questions

- Are there use-cases for a virtual non-volatile memory available to the guests?
- Can non-volatile memory be over-committed/multiplexed like other resources?

Why? Use-cases for Non-volatile Memory

- Two broad categories
  - Transparent to the application
  - Visible to the application

Why? Use-cases for Non-volatile Memory

- Transparent to the application
  - Block cache [Byan, MSST 12]
  - Second chance page cache[Venkatesan, HPCC 14]
  - Hybrid main-memory systems [Hirofuchi, SOCC 16]
  - Specialized file systems for non-volatile memory [Lee, FAST 15]

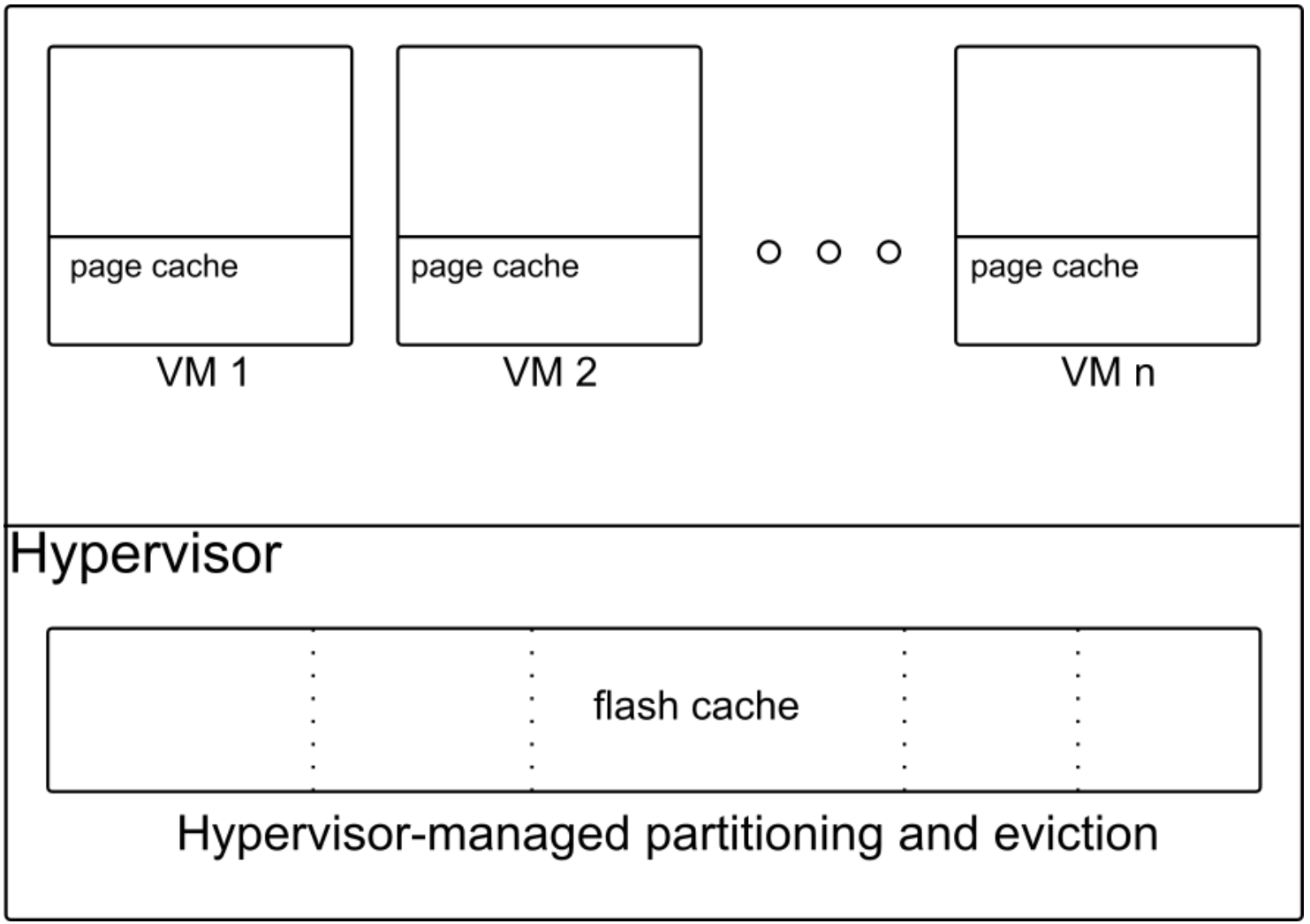
Why? Use-cases for Non-volatile Memory

- Visible to the application
  - Libraries facilitating applications to place data in persistent memory [Dulloor, EuroSys 16]
  - Applications taking advantage of open-channel SSDs [Wang, EuroSys14]

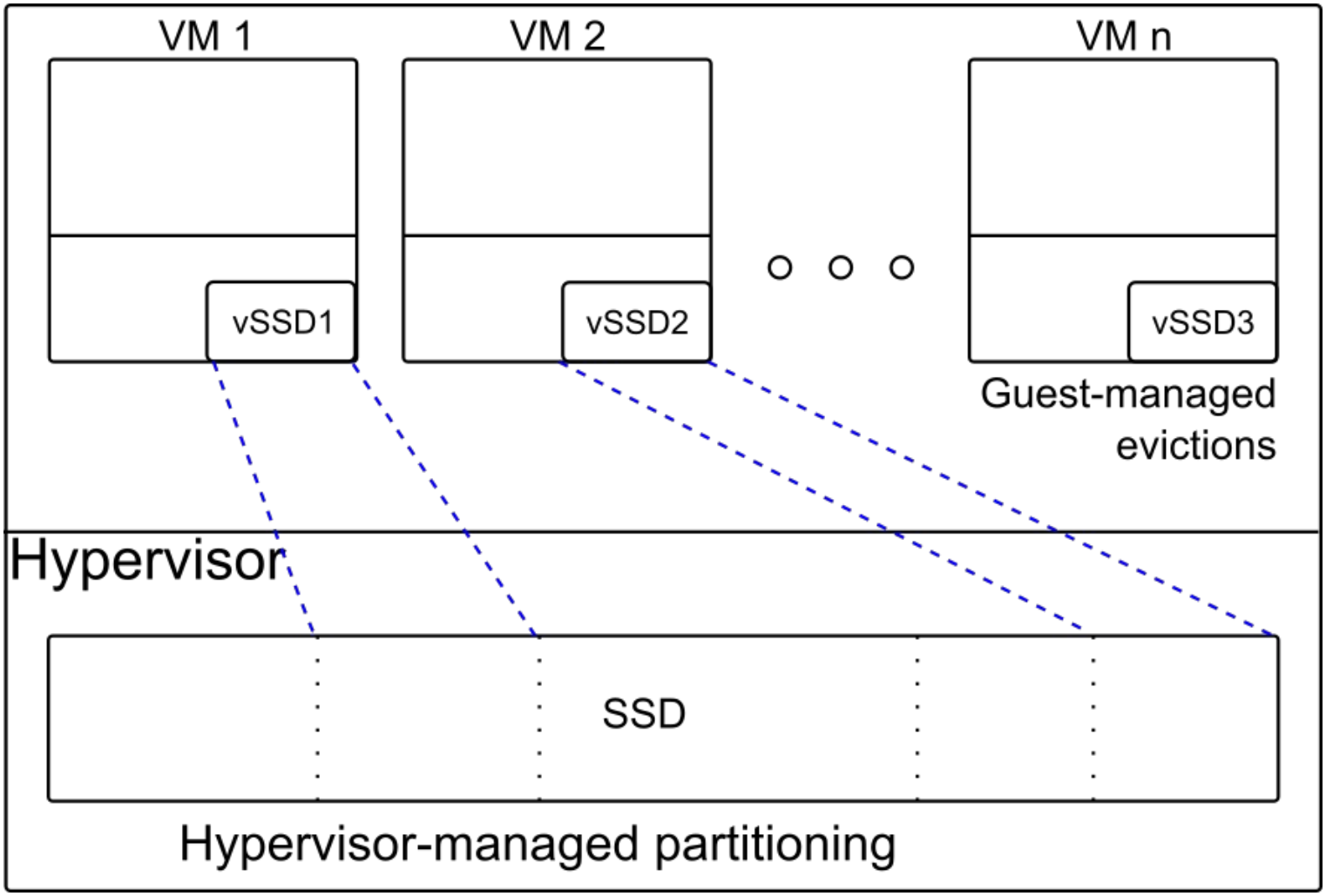
Why? Over-committing Non-volatile Memory

- [Liang, SOCC 16] is the only paper that makes a case for virtualizing persistent memory
- Virtualize byte-addressable memory
- Try to optimize placement of guest pages (in DRAM vs PM)

Why? A Motivating Use-case - Host-side Caching



What? Guest-aware Host-side Caching



How? Design

- A new type of dynamically resizable block device in the guest
- VirtIO paravirtualized drivers for communication of special events
- VirtIO backend can notify the frontend of a resize event
- Frontend notifies the block caching layer to release *N* blocks.

How? Challenges

- An efficient on-disk data structure to keep block mappings
- Finding a means of communicating to the guest caching layer via the frontend
- Guest caching layer should react appropriately to a resize event. Current caching solutions work with a fixed size block cache device and make mappings accordingly

How? Current Work

- Just beginning implementation by looking VirtIO block driver code
  - Traced and understood the flow of a read/write from VFS to a block device driver
  - VirtIO backend uses the new multiqueue block IO introduced in linux 4.0. Need to understand that well before proceeding.
- This topic was chosen only two months ago! (August 11)
  - Abandoned the earlier topic because we could not zero in on a system model to go ahead
  - Even though the problem seemed interesting enough
  - Listed the failure modes in a data center and possible modes of operation

How? Future Plan

- Will need to compare the proposed solution with hypervisor managed solutions vis-a-vis performance
- Come up with more use-cases for a generic resizable persistent memory
  - for example, how to handle cases where a guest installs file system on it

Thank you!

Comments/Questions?

References

- Byan, S., Lentini, J., Madan, A., and Pabón, L. **Mercury: Host-side flash caching for the data center**. In IEEE 28th Symposium on Mass Storage Systems and Technologies (MSST), 2012 (apr 2012)
- Dulloor, S. R., Roy, A., Zhao, Z., Sundaram, N., Satish, N., Sankaran, R., Jackson, J., and Schwan, K. **Data Tiering in Heterogeneous Memory Systems**. In Proceedings of the Eleventh European Conference on Computer Systems (New York, NY, USA, 2016), EuroSys '16, ACM
- Hirofuchi, T., and Takano, R. **RAMinate: Hypervisor-based Virtualization for Hybrid Main Memory Systems**. In Proceedings of the Seventh ACM Symposium on Cloud Computing (New York, NY, USA, 2016), SoCC '16, ACM.
- Koller, R., Mashtizadeh, A. J., and Rangaswami, R. **Centaur: Host-Side SSD Caching for Storage Performance Control**. In IEEE International Conference on Autonomic Computing (ICAC), 2015 (jul 2015).

References

- Lee, C., Sim, D., Hwang, J., and Cho, S. **F2FS: A New File System for Flash Storage**. In Proceedings of the 13th USENIX Conference on File and Storage Technologies (FAST 15) (Santa Clara, CA, 2015), USENIX Association
- Liang, L., Chen, R., Chen, H., Xia, Y., Park, K., Zang, B., and Guan, H. **A Case for Virtualizing Persistent Memory**. In Proceedings of the Seventh ACM Symposium on Cloud Computing (New York, NY, USA, 2016), SoCC '16, ACM.
- Mittal, S., and Vetter, J. S. **A Survey of Software Techniques for Using Non-Volatile Memories for Storage and Main Memory Systems**. IEEE Transactions on Parallel and Distributed Systems 27, 5 (may 2016)
- Venkatesan, V., Qingsong, W., and Tay, Y. C. **Ex-Tmem: Extending Transcendent Memory with Non-volatile Memory for Virtual Machines**. In IEEE International Conferance on High Performance Computing and Communications, 2014 (aug 2014)

References

- Wang, P., Sun, G., Jiang, S., Ouyang, J., Lin, S., Zhang, C., and Cong, J. **An Efficient Design and Implementation of LSM-tree Based Key-value Store on Open-channel SSD**. In Proceedings of the Ninth European Conference on Computer Systems (New York, NY, USA, 2014), EuroSys ’14, ACM