



# Classifying and Mitigating Side-Channel Vulnerabilities between VMs

Jinpeng Miao\* Dwight Browne\* Abdulrahman Alaraj\* Tamara Lehman\* Daniel Massey\*  
\* University of Colorado Boulder

## Objectives

- Classify VM side-channel vulnerabilities novelly
- Typical corresponding defenses discussion
- Propose an effective mitigation direction
- Minimize performance loss and resource consumption
- Comprehensively suite and protect against possible future attacks
- Encouragement purpose

## Status Quo

- Popularity of large-scale cloud services and virtualization technology breeds more powerful remote side-channel attacks.
- The fact that side-channel attacks exploit different properties requires distinct defenses, hence making it difficult to defend against.
- Existing defenses for these vulnerabilities are either limited to a single vulnerability or causing a lot of performance loss.
- Hard to completely solve these problems with small software and hardware patches.
- Failure to consider future related potential attacks and prevent them in advance, which will once again make us vulnerable.

## References

Figure1: <https://blog.trailofbits.com/2015/07/21/hardware-side-channels-in-the-cloud/>  
 Figure2: <https://www.starwindsoftware.com/blog/how-transparent-page-sharing-memory-deduplication-technology-works-in-vmware-vsphere-6-0>

[1] Kocher, Paul, et al. IEEE S&P. 2019.  
 [2] Weisse, Ofir, et al. (2018).  
 [3] Yan, Mengjia, et al. IEEE MICRO. 2018.  
 [4] Khasawneh, Khaled N., et al. IEEE DAC. 2019.  
 [5] Sakalis, Christos, et al. ACM ISCA. 2019.  
 [6] Turner, Paul. URL: <https://support.google.com/faqs/answer/7625886> (2018).  
 [7] Canella, Claudio, et al. USENIX Security. 2019.  
 [8] Naghibijouybari, Hoda, et al. ACM CCS. 2018.  
 [9] Lindemann, J., et al. ACM SIGAPP Applied Computing Review 18.4 (2019): 31-46.  
 [10] Gruss, Daniel, et al. ESORICS. Springer, Cham, 2015.  
 [11] Barresi, Antonio, et al. USENIX WOOT 15. 2015.  
 [12] Gras, Ben, et al. USENIX Security. 2018.  
 [13] Thomas, Claburn. [https://www.theregister.co.uk/2018/11/02/portsmash\\_intel\\_security\\_attack/](https://www.theregister.co.uk/2018/11/02/portsmash_intel_security_attack/) (2018-11-02)

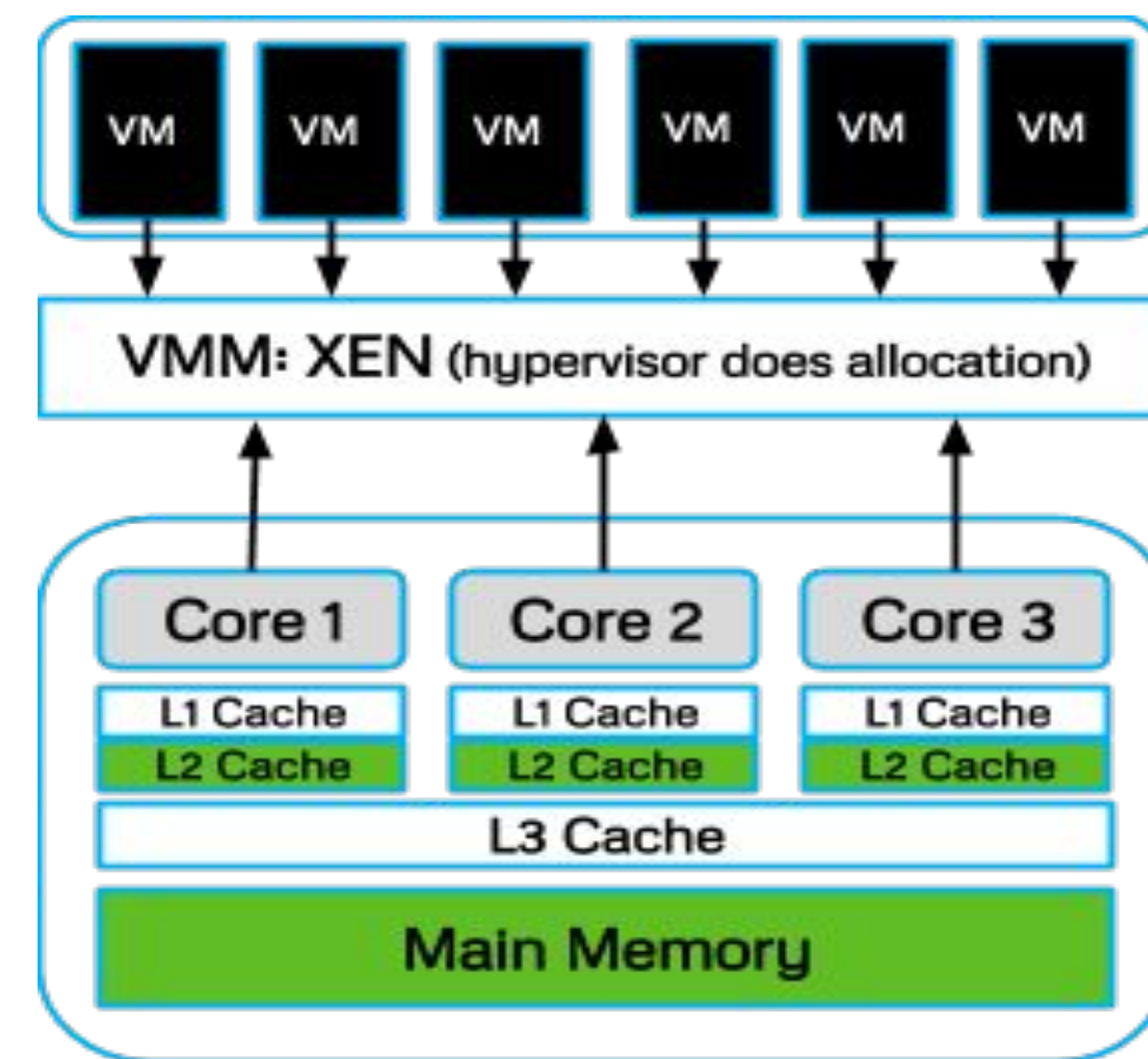


Figure1. VM Architecture

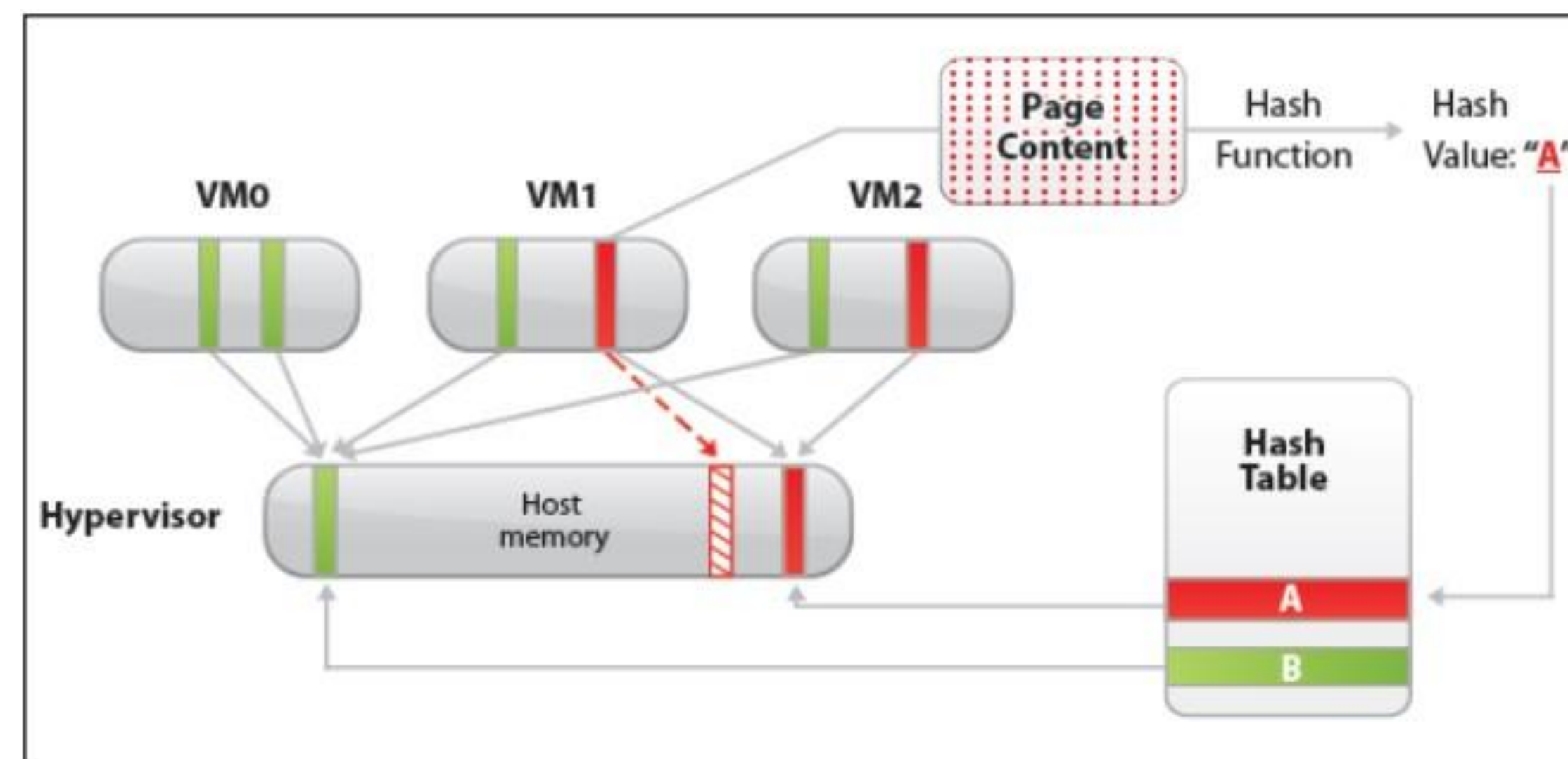
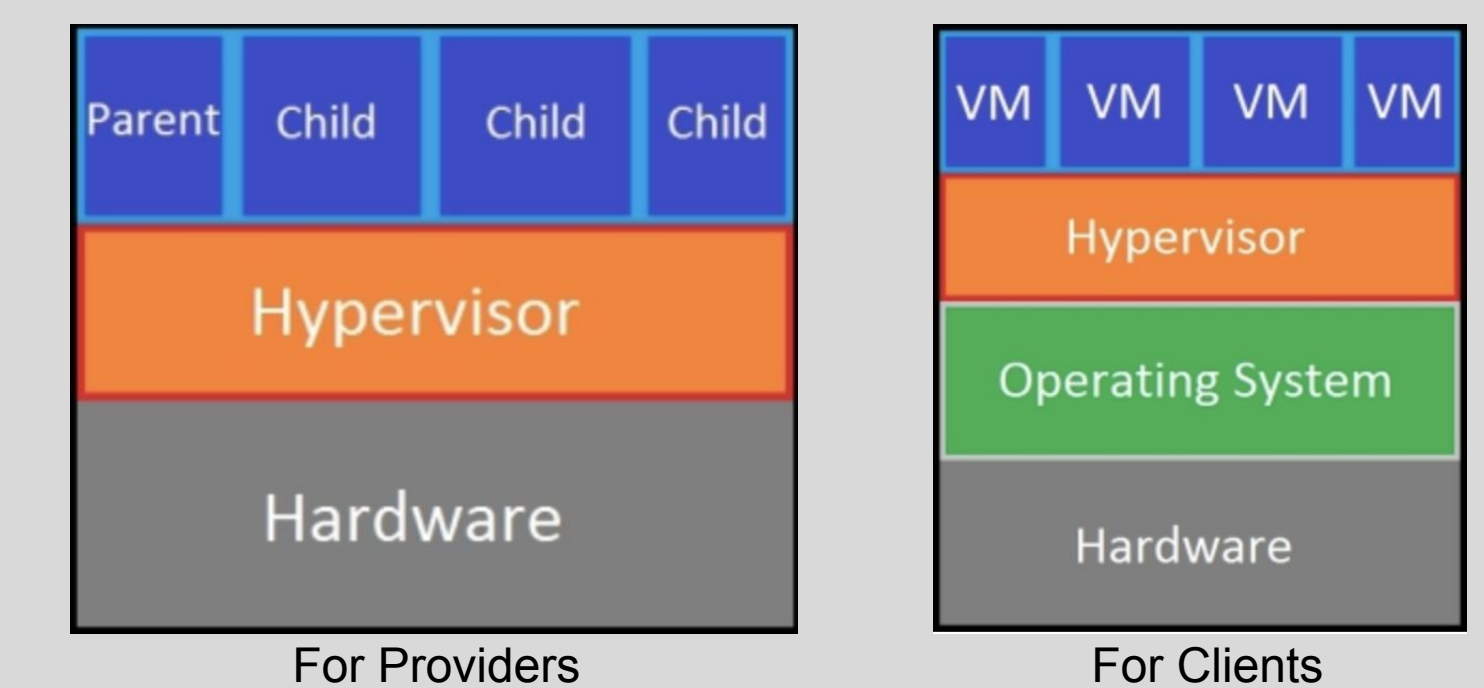


Figure2. Memory Deduplication

	Attacks	Description	Defenses	
CPU	Spectre-type Variants 1 [1]	"Bounds Check Bypass". V1 speculatively executes instructions inside conditional branches to perform an out-of-bounds memory access by bypassing the boundary check of memory accesses.	<ul style="list-style-type: none"> <li>• "InvisiSpec" [3]</li> <li>• "SafeSpec" [4]</li> <li>• "Efficient Invisible Speculative Execution Through Selective Delay and Value Prediction" [5]</li> <li>• Etc.</li> </ul>	
	Transient Execution Attacks	Spectre-type Variants 2 [1]	V2 mainly utilizes the mistrained <b>indirect branch predictor</b> of the CPU to speculatively execute specific code fragments the attacker specifies to complete the attack.	<ul style="list-style-type: none"> <li>• CPU microcode update</li> <li>• Software protection like "Retpoline" proposed by Google [6]</li> <li>• Etc.</li> </ul>
	Meltdown-type: Foreshadow-NG [2] (L1 Terminal Fault)	A malicious guest VM can fully <b>escape the virtual machine sandbox</b> to access memory which they do not have permission to access, such as hypervisor's memory, or even the memory belonging to another virtual machine.	<ul style="list-style-type: none"> <li>• Ensure that no hypervisor thread runs on a sibling core with an untrusted VM. [7]</li> </ul>	
Side Channel Attacks between VMS	GPU Contention Attacks [8]	<b>Description:</b> Multiprogramming on the GPUs can <b>create contention or even build threatened channels</b> between co-located VMs on the same cloud node.	<b>Defense:</b> <ul style="list-style-type: none"> <li>• Eliminate the shared APIs and the performance counters</li> <li>• Rate limiting and precision limiting</li> </ul>	
Resource Related Attacks	Application Detection [9,10]	<b>Description:</b> Observable <b>timing differences in write accesses</b> on deduplicated pages can be exploited to identify applications or user activities in another co-residency VMs.	<b>Defense:</b> <ul style="list-style-type: none"> <li>• Deactivate memory deduplication</li> <li>• Blur time differences</li> <li>• Obfuscate memory</li> <li>• Memory encryption</li> </ul>	
	ASLR Breaking [11]	<b>Description:</b> Memory deduplication feature can be utilized to <b>leak the address space layouts</b> of the co-residency VMs, and hence, defeats ASLR.	<b>Defense:</b> <ul style="list-style-type: none"> <li>• Deactivate memory deduplication</li> <li>• Increase ASLR entropy</li> <li>• increase entropy in sensitive memory pages.</li> </ul>	
	Shared TLB [12]	<b>Description:</b> Shared TLB can be abused to leak secret information in CPU with <b>hyperthreaded access to another hyperthreaded information</b> running on the same core, Like TLBleed and PortSmash.[13]	<b>Defense:</b> <ul style="list-style-type: none"> <li>• Disable hyperthreads</li> <li>• Partitioning the TLB between sensitive processes</li> <li>• Partitioning the TLB in hardware</li> </ul>	

## Our Proposal

- Hard to entirely solve these problems with small patches.
- The same component of both two mainstream architectures (shown below) is: hypervisor.
- Almost all the techniques exploited by attackers are on the hypervisor level.
- Therefore, hypervisor is the key to mitigate these vulnerabilities. Thus, we plan followings:
  - Develop a new hypervisor to completely conquer these challenges.
  - Small modifications on the hardware as a supplement, to improve overall resource utilization and system performance.



## Research Impact

- ❖ **Users**
  - Protect users' privacy.
- ❖ **Network Environment**
  - Solve co-resident attacks with minimal energy and performance loss.
  - Even if one VM is malicious, other VMs will not be affected.
- ❖ **Entire Society**
  - Cloud is ubiquitous. Protecting the cloud space is protecting our entire society.
- ❖ **Future**
  - Predict and prevent future related potential attacks.