Disclaimer

• This talk vastly over-simplifies things.
• See notes for full details and resources.

https://github.com/gregkh/presentation-mds
• Same “family” of bugs as Spectre/Meltdown
• Hardware bugs
• Exploits the speculative execution model of modern CPUs.
• Many different variants.
• Is going to be with us for a very long time!
MDS

- MDS == “RIDL”, “Fallout”, “Zombieload”, and others
- CPU Hardware bugs
- Variants of the same basic problem
- Exploits the speculative execution model of Intel CPUs.
- Discovered by many different research teams
- Kernel and BIOS fixes required to fully solve
• One program can read another program’s data
• Can cross the virtual machine boundary
• Exploits “hyper threading” (SMT) issues
• SMT are CPUs that usually share TLBs and L1 cache
OpenBSD was right

- Guessed more problems would be in this area
- Disabled SMT for Intel chips in June 2018
- Repeated the plea to disable this in August 2018
- Prevented almost all MDS issues automatically
- Security over performance
- Huge respect!
• Rouge-Inflight-Data-Load
• Exploits CPU Line-fill buffers and Load ports
• Steal data across applications, virtual machines, secure enclaves
• Kernel fix by flushing CPU buffers/ports on context switch
Fallout

- Exploits CPU Store Buffers
- Read kernel data from userspace
- Breaks ASLR (random kernel addresses)
- “Meltdown” mitigation made this easier to exploit
- Kernel fix by flushing CPU buffers on context switch
Zombieland

- Exploits CPU Line-Fill buffers
- Much like RIDL
- Steal data across applications, virtual machines, secure enclaves
- Cool logo/name and demo
- Kernel fix by flushing CPU buffers on context switch
“Store-to-Leak forwarding”
“Meltdown UC”
All allow data to be stolen across security boundaries
Kernel fix by flushing CPU buffers/ports on context switch
Flushing CPU buffers is slow

- All of these mitigations slow down the system
- No way yet to schedule different security domains on different physical processors (gang scheduling)
- Disabling SMT mitigates most problems (not ALL!)
- Must disable SMT and enable mitigations to solve completely.
Flush CPU buffers is slow

- Performance numbers depend on your workload
  - Kernel build
    - 2% smt=on
    - 15% smt=off
    - Heavily multi-threaded, CPU bound
  - Kernel creation, no decrease
    - Single threaded, I/O bound
- Syscalls are now expensive
- Test your workload!
Do you feel lucky?

- Users must now choose between performance and security
- What choice did your cloud provider choose?
- https://make-linux-fast-again.com/
  - Kernel builds faster by 15%!
Linux’s response

• Kernel fixes available on announcement date
• Intel notified some kernel developers in advance
• Worked together across OS vendors to solve
• Much better than Spectre/Meltdown
• Process still needs to improve, Debian notified 48 hours before release.
• More fixes came after announcement
• Update your kernel and BIOS!
Linux security fixes

• Happen at least once a week
• Look like any other bugfix
• Rarely called out as security fix
• Many bugfixes not known to be security related until years later
• No differentiation between bug types
  • A bug is a bug is a bug
• Very few CVEs ever get assigned for kernel security issues
Linux security fixes ≠ CVEs

• Small fraction of kernel security fixes get CVEs
• If you only cherry-pick CVEs, you have an insecure system
• Some CVEs have follow-on fixes not documented anywhere

How the Linux Kernel Security team works
Linux security fixes ≠ CVEs

- Small fraction of kernel security fixes get CVEs
- 2006-2018 had 1005 CVEs assigned to the kernel
  - 41% (414) had a negative “fix date”
  - 12 never fixed
  - Average fix date, -100 days
  - Longest fix dates, -3897 and 2348 days
  - 88 fixed within 1 week
  - Standard deviation 405
Linux Longterm Kernels Fix Problems

• Bugs are fixed before you realize it is a issue.
• Google security team requests for Pixel phones in 2018:
  • 92% (201/218) problems were already fixed in LTS kernel
  • No need for cherry-picking or backporting
  • Remaining issues were due to out-of-tree code
If you are not using a supported Linux distribution kernel, or a stable / longterm kernel, you have an insecure system.
Disclaimer

- This talk vastly over-simplifies things.
- See notes for full details and resources.

https://github.com/gregkh/presentation-mds

In order to keep this talk within the time limit, I am vastly over simplifying things.

Please see the presentation notes at the link here for more details and a full list of resources on how to find out more information about this topic.
• Same “family” of bugs as Spectre/Meltdown
• Hardware bugs
• Exploits the speculative execution model of modern CPUs.
• Many different variants.
• Is going to be with us for a very long time!
MDS

• MDS == “RIDL”, “Fallout”, “Zombieload”, and others
• CPU Hardware bugs
• Variants of the same basic problem
• Exploits the speculative execution model of Intel CPUs.
• Discovered by many different research teams
• Kernel and BIOS fixes required to fully solve
MDS

- One program can read another program’s data
- Can cross the virtual machine boundary
- Exploits “hyper threading” (SMT) issues
- SMT are CPUs that usually share TLBs and L1 cache
OpenBSD was right

- Guessed more problems would be in this area
- Disabled SMT for Intel chips in June 2018
- Repeated the plea to disable this in August 2018
- Prevented almost all MDS issues automatically
- Security over performance
- Huge respect!
RIDL

- Rouge-Inflight-Data-Load
- Exploits CPU Line-fill buffers and Load ports
- Steal data across applications, virtual machines, secure enclaves
- Kernel fix by flushing CPU buffers/ports on context switch
Fallout

• Exploits CPU Store Buffers
• Read kernel data from userspace
• Breaks ASLR (random kernel addresses)
• “Meltdown” mitigation made this easier to exploit
• Kernel fix by flushing CPU buffers on context switch
Zombieland

- Exploits CPU Line-Fill buffers
- Much like RIDL
- Steal data across applications, virtual machines, secure enclaves
- Cool logo/name and demo
- Kernel fix by flushing CPU buffers on context switch
Other variants

- “Store-to-Leak forwarding”
- “Meltdown UC”
- All allow data to be stolen across security boundaries
- Kernel fix by flushing CPU buffers/ports on context switch
Flushes CPU buffers is slow

- All of these mitigations slow down the system
- No way *yet* to schedule different security domains on different physical processors (gang scheduling)
- Disabling SMT mitigates most problems (not ALL!)
- Must disable SMT and enable mitigations to solve completely.
Flushing CPU buffers is slow

- Performance numbers depend on your workload
- Kernel build
  - -2% smt=on
  - -15% smt=off
  - Heavily multi-threaded, CPU bound
- Kernel creation, no decrease
  - Single threaded, I/O bound
- syscalls are now expensive
- Test your workload!
Users must now choose between performance and security
What choice did your cloud provider choose?
https://make-linux-fast-again.com/
Kernel builds faster by 15%!
<table>
<thead>
<tr>
<th>Linux’s response</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Kernel fixes available on announcement date</td>
</tr>
<tr>
<td>• Intel notified some kernel developers in advance</td>
</tr>
<tr>
<td>• Worked together across OS vendors to solve</td>
</tr>
<tr>
<td>• Much better than Spectre/Meltdown</td>
</tr>
<tr>
<td>• Process still needs to improve, Debian notified 48 hours before release.</td>
</tr>
<tr>
<td>• More fixes came after announcement</td>
</tr>
<tr>
<td>• Update your kernel and BIOS!</td>
</tr>
</tbody>
</table>
Linux security fixes

• Happen at least once a week
• Look like any other bugfix
• Rarely called out as security fix
• Many bugfixes not known to be security related until years later
• No differentiation between bug types
  • A bug is a bug is a bug
• Very few CVEs ever get assigned for kernel security issues
Linux security fixes ≠ CVEs

- Small fraction of kernel security fixes get CVEs
- If you only cherry-pick CVEs, you have an insecure system
- Some CVEs have follow-on fixes not documented anywhere

How the Linux Kernel Security team works
Linux security fixes != CVEs

- Small fraction of kernel security fixes get CVEs
- 2006-2018 had 1005 CVEs assigned to the kernel
  - 41% (414) had a negative “fix date”
  - 12 never fixed
  - Average fix date, -100 days
  - Longest fix dates, -3897 and 2348 days
  - 88 fixed within 1 week
  - Standard deviation 405
Linux Longterm Kernels Fix Problems

- Bugs are fixed before you realize it is an issue.
- Google security team requests for Pixel phones in 2018:
  - 92% (201/218) problems were already fixed in LTS kernel
  - No need for cherry-picking or backporting
  - Remaining issues were due to out-of-tree code
If you are not using a supported Linux distribution kernel, or a stable/longterm kernel, you have an insecure system.