1. RIDL and Fallout

1

Speculative execution is a CPU feature that allows for instructions that may or may not be executed in the future to be executed earlier in the timeline, leading to an improvement in performance but also an increase in the number of unnecessary instructions executed. This technique typically leads to the execution of incorrect code, that is ultimately reverted when the real instruction does indeed reach the present moment in the execution timeline. This technique was implemented in most Intel processors with some flaws like not having to check that the data being accessed from cache is private or not. Additionally, speculatively executed instructions can leave temporary data in cache.

A speculative execution attack takes advantage of these flaws somehow and can therefore access data that should typically not be accessed by the executing program/user.

RIDL and Fallout (RaF) share some similarities with Meltdown and Spectre (MaS), namely they all make use of flaws in the speculative execution implementation of some processors. However, RaF focus on leaking arbitrary data from CPU-internal buffers, including data that is not stored in the CPU cache - (Line-Fill Buffers and Load Ports for RIDL and Store Buffers for Fallout) - whereas Meltdown uses another speculative execution flaw combined with a race condition flaw at the CPU level to access unauthorized data from any address that is mapped to the current process's memory space (which can be controlled in the case of using speculative execution of branched code) and Spectre also focuses on branch-prediction techniques (special case of speculative execution) and it is a more general type of vulnerability than Meltdown itself, with multiple exploits possible.

2

RIDL has the following CVEs:

- [MFBDS] CVE-2018-12130
- [MLPDS] CVE-2018-12127
- [MDSUM] CVE-2019-11091

Fallout has the following CVE:

- [MSBDS] CVE-2018-12126

3

RIDL and Fallout fall under the CWE-200: Information Exposure.

Under the CIA (Confidentiality, Integrity and Availability) they fall under the confidentiality umbrella. Nevertheless, they can also work as starting points for attacks that compromise integrity or even affect availability.

4

This has partially been answered in the previous question, but other consequences involve using access to unauthorized information and files to:

- introduce backdoors
- exfiltrate sensitive data
- ransomware
- delete data
- and more

These are consequences that can eventually compromise the complete CIA-triad.

5

Hardware vulnerabilities are closely related to the low-level components of the computer, their architecture and their logic. Whereas software vulnerabilities are related to all software developed by computer users rather than computer makers. For these facts, hardware vulnerabilities are much less common but can also be much more serious in cases where they are widespread (as the ones previously analysed).

For example CVE-2018-18674 describes a software vulnerability - cross site scripting - that results from poor implementation of the code that will display user input to other users (namely in the `adm/board_form_update.php bo_content_tail` variable). On the other hand, CVE-2019-11163 describes a vulnerability in a driver for an Intel processor that is related to a lack of access control - although this is a flaw in the driver code (software) the fact that it is in a component that is essentially responsible from iterating directly with the computer hardware makes the case that this is a hardware vulnerability plausible.

6

Let us dissect Mr. Super Secure's words:

1. `"full anti-virus scan"` - this action would cover only the cases where the anti-virus companies had already deployed means of identifying vulnerabilities that were a direct consequence of RaF attacks. However, these attacks are of a nature (Information Exposure) that does not prompt viruses to exist as a first consequence, rather the exfiltration of data which sometimes is undetectable by anti-viruses.
2. `"installed all application updates"` - this action makes sense as the role of application updates is to patch recently discovered vulnerabilities, but these should always be kept up to date and not just when a "big" new vulnerability is found
3. `"personal firewall"` - this approach seems rather out-of-the-blue since anything Mr. Super Secure does by having his own personal firewall will most likely be less effective than those that come out-of-the box with operating systems or even anti-virus programs.
4. `"my computer should be secure enough"` - this assumption is always tricky to make, even just for the case of RIDL and Fallout, but besides making sure that the **relevant** patches for RaF are installed, there is little more to do to guarantee that a machine is out of reach from them.

Of course Mr. Super Secure took the extra steps of "closing" some network access to his computer, but he might just have missed a port or a protocol that would allow an attacker to use RIDL or Fallout on his machine.

All in all, I agree with some of Mr. Secure's actions (installed patches) and even grant some merit to others (anti-virus scan) but typically I would not go as far as assume I am capable of designing a firewall that would deter any attacks that the companies that produce the software I own can.

Indeed, the amount of complexity in these systems is so big that not even Mr. Super Secure might be capable of handling it all, that is why PC security software such as anti-virus are so important and also why patches should always be done and not only when a new attack is discovered or made public (which might even be later than the discovery moment).

7

The first measure would definitely make sure that the correct patches for those vulnerabilities are available and installed.

The second measure would be to follow official guidelines to ensure protection such as Windows client guidance for IT Pros to protect against speculative execution side-channel vulnerabilities. If any such guide would not be available for my machine/Os I would consider finding the configuration that would allow me to turn speculative execution off, at the cost of less processor efficiency, but more security.