SPECTRE Vulnerability

What is this vulnerability?

SPECTRE is a vulnerability in modern processors that takes advantage of speculative execution in order to gain access to sensitive data from a computer's memory. This vulnerability is essentially taking advantage of a technique that is intended to improve a CPU's performance.

What is its root cause?

In speculative execution, a processor essentially tries to predict what instructions will be needed next, executes them ahead of time, caches the results, and removes them from memory if they are not actually needed. However, even if the data that is cached is removed, there can still be traces of it remaining in the cache. Attackers can utilize timing attacks, where they monitor how long it takes to access memory, in order to infer what data was found by the pre-executed instructions. Essentially, this was caused by CPUs prioritizing performance over security.

What is the extent of its impact?

Essentially all modern CPUs at the time of SPECTRE's discovery (2018) were affected by this vulnerability. All devices that utilized these CPUs were vulnerable, including personal computers, mobile devices, cloud servers, etc. There are many risks associated with the vulnerability, such as data leaks, cross process attacks, or more specific vulnerabilities that may present themselves in a cloud environment.

How to patch it? (Or how it was patched by the vendor)

The patches to this vulnerability took both software and hardware forms. In terms of software, on the OS level, there were proper authorization levels added so that user applications can't access unauthorized memory, while on the compiler level, there were techniques added so that CPUs can't perform unsafe branch predictions. On the hardware side, Intel, AMD, and ARM redesigned processors to specifically address SPECTRE vulnerabilities. This includes restricting speculative execution and preventing one thread from influencing another thread's predictions.

How could it have been prevented?

This would have been a difficult vulnerability to prevent, as it was an exploit in the fundamental design of processors at the time, but, with this being said, the design could have changed. For example, processors could have had stronger isolation between processes or stronger cache protections in order for processes to not be able to access memory utilized by other processes.