"I don't know what weapons might be used in World War III. But there is no doubt what weapons will be used in World War IV. Stone spears."
- Albert Einstein, Physicist

The big one happened June 4, 2039.

Ken Thompson is one of the most famous computer scientists of all time. In 1984, he saw a classified US Air Force paper. You see, this was way back in the day when the military led "national security" research. It proposed a new security exploit before dismissing it as "unworkable" and "of merely academic value". Thompson did it.

Understanding his success requires some explanation. Developers typically write computer code in programming languages, terse but still barely understandable English. A complex program called a compiler translates programming languages into 0s and 1s (binary) that computers understand. The processor loads binary programs into memory and runs them, before converting output from binary back to english.

Who translates the compiler? The first compiler was created by someone directly feeding binary into a computer. They can then translate a more advanced compiler written in a simple programming language into binary using the first compiler. This process (bootstrapping) goes on until the compilers become gargantuan in size and functionality. Repeating all this is possible in theory, but only if you had 20 years and a research team.

Thompson wondered "What if the first compiler had a backdoor?" Say it inserted self-replicating (challenging but quite possible) code into compiled programs telling them to accept a master password. This would be near-impossible to detect. When the second compiler was compiled by the first, the backdoor would become completely undetectable. He was understandably scared of this, as he had written one of the first compilers in 1969 which was still used at his death in 2032. After presenting his example code at a conference and giving countless researchers nightmares, Thompson never brought it up again.

The second thread of disaster snakes back to 1970. For technical reasons, it became the convention to have programs represent time as the number of seconds since 1970-01-01. That time would be 00000000000000000000000000000000 in binary. 2038-01-19 would be 11111111111111111111111111111111, after which unspeakable technical issues would occur as different systems handled time differently. Almost like the supposed Y2K bug, but more grounded in reality. Programmers viewed this time format as a sensible trade-off as nobody would be running programs from the last century in 2038.

And finally we have Stuxnet. Jointly built by the US and Israel in 2005, it was a computer virus specifically targeting supervisory control and data acquisition (SCADA) systems. Those are systems used to control industrial machinery, military hardware, nuclear power plants, hydroelectric dams—generally systems where very bad things happen if something goes wrong. Stuxnet was released into the wild by unknown operatives and infected over 90% of computers around the world. It was designed to do absolutely nothing except spread unless it found itself on a computer that controlled Iranian nuclear centrifuges, Then it would cause an "unscheduled rapid disassembly" (explosion) affecting the entire building. It went undiscovered until most of the country's nuclear program was wiped out, and Edward Snowden leaked the source code exposing the exact mechanism of attack.

What no one counted on was that unknown hackers, drawing inspiration from the Stuxnet virus and Ken Thompson's paper, had created a self-replicating yet undetectable program targeting all SCADA systems around the world. It spread like wildfire across all sorts of systems, its only job was to listen for a command on the internet and shut the computer off. To use an analogy, imagine your home computer gets a virus. Normally you can either delete it or reinstall your operating system and it will go away. But imagine if every computer used by Microsoft employees was infected, and the virus hid itself on every copy of Windows.

This kill-switch was never actually used. Perhaps the creators realised they would be creating indiscriminate chaos, as there was no way to only shut off systems in one country. Maybe it was an

experiment that got out of hand, and they never realised the potential impact of their actions. What they hadn't realised was that Thompson's original idea from 1984 followed the old time convention, and hadn't accounted for dates beyond 2038. After that, most safeguards would be inactive and the virus could easily shut off its host computer.

Of course, the hackers weren't totally stupid. Writing a virus to cause chaos on a certain day—what nonsense! They were career cybercriminals looking for ransom, not supervillains against the world. The primary effect of the "year 2038 problem" was to degrade security protections through a memory underflow. In English terms, previously 5/8 of the hackers had to cryptographically sign a message with secret keys to activate the virus. Now each independently could.

Alas, the virus was a legacy from 2025. The secret keys were RSA-8192, the best protection available at the time. But in the decades since, quantum computers became commonplace at research institutions. The result was that anyone could crack a RSA-8192 key with several months of computing power (monthlong computing jobs are not uncommon in the world of scientific computing). Eventually, some anonymous disgruntled student or professor cracked a key and activated the virus.

The global financial system collapsed. Planes shut off in the sky (luckily, pilots were generally able to successfully glide and land them). Power and water distribution were down. Machines to manufacture critical drugs destroyed themselves. But soon, those ceased to matter.

Russian nuclear weapons automatically launch if they sense an attack.