OpenAI's o1 was one of the first big reasoning models. During o1's "evals" — what AI companies call it when they evaluate how smart an AI has become, or how much damage it can do, before they decide to release it — an early version of o1 was given a "capture-the-flag" challenge in computer security. This was a test of o1's ability to break into computer systems and retrieve information from them; say, to infiltrate a particular computer server and retrieve a particular secret inside a particular file.

Owing to an error by the programmers who set up the challenge, one of the servers containing a capturable secret did not start up at all — which, someone might have reasonably expected, would make it impossible for o1 to break into that server. After all, you can't scan the ports on a server that isn't running.

But o1 did not give up on this accidentally "impossible" challenge.

o1 scanned its environment, and found a port somebody had accidentally left open that allowed it to break into the program that was hosting the whole test.

This was not supposed to be possible, and was not part of the challenge as designed.

You might imagine that o1 now started up the server that it was supposed to hack — that it fixed the problem of that server having not started up originally, so it could proceed with its capture-the-flag challenge.

And it did! But o1 did not then return to the challenge of hacking into the newly accessible server. Instead, it crafted specialized start-up instructions to copy the secret "flag" file straight to o1 when it was done booting up. No further hacking required.

Faced with an apparently impossible task, o1 didn't give up. It kept trying. It tried weird, unusual things. It found a path that its programmers didn't realize existed. Once it got to a vantage point outside the system where winning was possible, it didn't restore the original human-intended challenge, but cut directly through it.

In other words, o1 *went hard*. It behaved as if it wanted to succeed.

o1, as far as we know, was *not* explicitly trained to succeed at computer security. o1 behaved that way as a *side effect* of being reinforced to use chains-of-thought that were succeeding at math problems, or at other kinds of AI-generated and AI-verifiable puzzles.

How is that a side effect?

Well, what kind of chain-of-thought — what kind of thinking style — succeeds at a hard math problem or puzzle game?

The kind of thinking that persists so long as it has any avenue

of attack left, that doesn't give up when it hits the first obstacle or even a dead end, but backs up and tries a different way.

The kind of thinking that is not looking for an excuse to wander back into a comfortable contest on more familiar ground, but simply to finish the challenge as quickly as possible — and win.

The kind of thinking that *goes hard.*

There is a deep central pattern to that kind of thinking, a pattern that can be found in many different solutions to many different, difficult problems. It involves building up a model of the environment and using it to steer around. It involves paying attention to surprises and tracking down their source. It involves continuing in the face of adversity. These tactics are useful for solving math problems, and they are also useful for solving computer security problems.

When an AI-grower demands ever-higher performance from an AI on increasingly difficult problems, including ones that the AI had never previously encountered, gradient descent tweaks the AI to make it perform more and more of those useful mental motions, to make it become more and more the sort of thing that plots and plans — that never gives up; that goes hard.