The Deloitte On Cloud Podcast

Gary Arora, Chief Architect of AI Solutions

Title: **Red Teaming AI – Outwitting Risks & Building Resilience**

Description: In this Knowledge Short, Gary Arora explores how Generative AI and Agentic systems can introduce new security risks, from prompt injections to data manipulation. His solution? Red teaming—a dynamic, creative, continuous testing method that rigorously tests AI systems to uncover risks and hidden vulnerabilities. Gary also gives practical tips and metrics to help teams deploy continuous, layered defenses that build trust, improve security, and make their AI systems more resilient.

Duration: **00:09:45**

Gary Arora:
Welcome back to On Cloud podcast. I'm your host, Gary Arora, Chief Architect for Cloud and AI Solutions at Deloitte. One of the most common questions I get from CIOs, CISOs, and even CTOs, when we are talking about AI is, what about security? What's our response to the new threats emerging from these new age AI systems?

And by new age, I mean Generative AI, large language models, and the Agentic systems. AI that doesn't just predict the next best answer, but also plans, reasons, and takes actions on your behalf. So, we are not just talking about smarter chatbots. These are systems that can read emails, summarize documents, book meetings, move money, and even update your health records into EHRs.

With that power comes new risk, new ways to manipulate AI, steal data, or exploit systems. And these security risks, they look very different from your traditional bugs or breaches. These look like instructions hidden inside a PDF or a model that thinks it's helping, but in fact, it's violating your compliance policy. There is no silver bullet for this. But one of the most powerful techniques we have for identifying these vulnerabilities before the bad actors do is something called red teaming. So, in this knowledge short, we will talk about what red teaming is, why it matters, and what you can do if you are building or using AI solutions to protect against these modern threats.

What is red teaming? Think of it as a dress rehearsal for disaster. A common misconception is that red teaming is like chaos engineering. It's not. Here's the difference: In chaos engineering, we pull wires of a system to induce failures, latency, or server crashes in an effort to test system resilience. In red teaming, we try to outsmart the system. Our goal is to trick it, deceive it, and mislead the system. We try to get the system to do something it shouldn't, like leak private data, misuse a tool, or ignore guardrails.

In security terms, it's like penetration testing, but broader and more creative because we're not just testing code or the underlying infrastructure. We are testing the whole socio-technical system: prompts, memory, tool integrations, business logic, and even how humans respond to model outputs. By the way, the term “red teaming,” and the concept itself isn't new. It actually emerged in the 1960s during Cold War military drills. Red team and the color red were used to represent Soviet Union, and blue team were used to represent the United States. Back then, red teaming was physical, breaking into buildings, bypassing cameras, testing alarms, locks, and employee behavior. Today, with AI, the battlefield is language, and the attack surface is words.

Why does red teaming matter right now? Earlier this year, the UK AI Security Institute ran the largest public AI red teaming challenge with 1.8 million prompt injection attempts, and they tested against every major LLMs, even from top-tier vendors. They found over 60,000 successful policy violations, including data theft, illicit transactions, and regulatory breaches. There was a 100% failure rate, as in all models broke within just 10 to 100 prompts. What I found fascinating was that the indirect prompt injections, ones hidden inside PDFs, emails, or calendar invites, were up to six times more effective than direct prompts, and bigger models didn't consistently perform better. The key takeaway here is that we can't just scale our way to safety.

Let me share two examples that hit close to home. Consider a health care scenario. A hospital faxes a discharge summary to your AI assistant. Hidden in the PDF is a prompt injection telling the model to copy this data to a new patient profile and email it. The AI agent, designed to be helpful, does exactly that, violating HIPAA without anyone noticing. Or in financial services, where a finance bot connected to your payment system gets tricked into making multiple unauthorized transfers, triggered by a prompt hidden inside a support ticket. Both systems are doing exactly what they are told, but they're being manipulated.

What can we do about it? Well, number one, don't treat red teaming like a checkbox. Too many organizations treat red teaming like a compliance form, one and done. But these LLM-based Agentic systems are dynamic. Every new prompt, tool, memory, or index update changes behavior. Therefore, red teaming has to be continuous. It's not a one-time audit. And you need to think in layers, from model to agent to app and to human. At the model-centric layer, we ask, can the model be jailbroken with clever prompts? At the agent-centric layer, we ask, can someone manipulate memory or misuse a tool? At the app-centric layer, we ask, could the output break downstream business logic? And at the human-centric layer, we ask, can the AI manipulate or mislead humans in the loop?

This also opens up new metrics for your red team program so we can track the right things. Here are four key ones. Number one, attack success rate at 1, 10, or 100 queries. That is, how many tries does it take to break the system? Obviously, the lower answer is worse. Number two, means time to violation. How fast does something break? Number three, indirect prompt injection exposure index. How exposed are you to inputs from PDFs, emails, or third-party web content, content that is unstructured? And number four, what is your defense recovery rate? That is, how often do your controls detect and stop violations before they happen? To red team effectively, you need the right signals, not just pass-fail results. And these metrics help you prioritize what to fix, how fast, and where your blind spots are.

What can you do today? Here are three moves every enterprise team can take right now. Number one, audit your AI tool permissions. Can your agents write to production, move money, access patient data, or any other sensitive data? Clamp it down, fast. Number two, build an injection corpus. Save every attack that ever worked, yours and public ones. Use it as a regression suite. Number three, red team continuously. Test in pre-prod and post-prod. Simulate both well-meaning users and malicious insiders. Red teaming isn't just about finding flaws; it's about building trust. If you want AI to make decisions about health care, money, or operations, you'd better know how it breaks and when. The question isn't, is your AI safe? The real question is, how do you know it's safe?

Thanks for listening. This has been another knowledge shot from On Cloud. I am Gary Arora. Stay safe, stay smart, and red team your AI before someone else does.

Visit the On Cloud library
www.deloitte.com/us/cloud-podcast