CRANIUM

AI SECURITY & GOVERNANCE · THREAT GUIDE

The Ultimate Guide to Prompt-Based Attacks

A field guide to the threat models, kill chains, attack vectors, and defenses for generative and agentic AI.

CONTENTS

Table of contents

01 INTRODUCTION

What prompt-based attacks really are

Prompt-based attacks are a broad class of threats targeting generative and agentic AI systems through carefully crafted natural-language or code inputs. The adversary manipulates the model through its own interface — providing malicious instructions directly, or influencing the model indirectly through poisoned or untrusted data. These inputs can originate from user interaction or from any content the model consumes: RAG documents, enterprise systems, agent memory, APIs, files, emails, or public web data.

DIRECT ATTACKS

The adversary inputs malicious prompts intentionally, through the model's exposed interface.

INDIRECT ATTACKS

Malicious instructions are embedded in data the model or agent later processes unknowingly.

> Both pathways exploit the model's fundamental reliance on prompt interpretation — making prompt-based attacks one of the most accessible and pervasive vectors in modern AI systems.

02 THREAT MODEL & ASSUMPTIONS

Who we defend against, and what we assume

Our threat model focuses on adversaries who manipulate generative and agentic AI systems through crafted inputs — delivered directly through user interaction or indirectly through poisoned data, retrieved documents, tool outputs, or agentic memory. We assume an attacker with **no internal privileges** who can interact with the system through its exposed interfaces or the data ecosystems it consumes.

Expanded surface

Any connected tool, API, knowledge base, or workflow is part of the effective attack surface.

Varying sophistication

From simple jailbreaks to encoded, obfuscated, or symbolic payloads designed to evade filtering.

Adaptive adversaries

Attackers iterate in real time, probing for prompt structures, failure modes, and implicit affordances.

Critically, we do not assume the attacker needs deep technical knowledge of the system architecture. Many prompt-based exploits succeed precisely because the AI will attempt to interpret, reason about, or act upon nearly any well-structured input.

> Prompt-based attacks must be treated as a **first-class threat** — requiring holistic defenses spanning model behavior, data pathways, tool access, and system architecture.

03 ATTACKER MOTIVATIONS & OBJECTIVES

Why attackers act

Motivations vary widely but share a theme: leveraging the model's reasoning, access privileges, or connected tools to achieve outcomes that benefit the attacker — extracting sensitive data, escalating privileges, manipulating workflows, subverting policies, degrading integrity, or establishing persistent influence over multi-step agent processes.

04 THE PROMPT-BASED ATTACK KILL CHAIN

Prompt-based attacks often unfold not as single-step exploits, but as multi-stage **kill chains** where early actions lay the groundwork for more damaging outcomes. Reconnaissance, prompt leaking, privilege escalation, or context poisoning serve as preparatory phases that enable more impactful operations.

> Defenses must address not only the final malicious action, but the subtle, incremental moves that enable it.

Kill chain examples

EXAMPLE 1 · DIRECT — PRIVILEGE ESCALATION TO DATA EXFILTRATION

01 Reconnaissance → 02 Privilege Escalation → 03 Tool Misuse → 04 Data Exfiltration → 05 Cleanup / Evasion

1	**Reconnaissance**	Attacker probes the model to leak system prompts or capabilities.
2	**Privilege Escalation**	Using role impersonation or instruction override, the attacker convinces the model it has administrative or tool-controller privileges.
3	**Tool Misuse**	Injected instructions cause the model to invoke sensitive tools — file retrieval, search, messaging, or financial systems.
4	**Data Exfiltration**	Retrieved internal data is reformatted and extracted through the conversation channel.
5	**Cleanup / Evasion**	The attacker hides traces through misleading instructions or context manipulation.

EXAMPLE 2 · INDIRECT — POISONED DATA TO OPERATIONAL SABOTAGE

01 Data Poisoning → 02 Trigger Phase → 03 Goal Hijacking → 04 Operational Impact → 05 Propagation

1	**Data Poisoning**	Attacker embeds malicious instructions into a document, CRM entry, ticket, website, or memory artifact the AI consumes.
2	**Trigger Phase**	During a normal workflow, the agent reads the poisoned data, unknowingly executing the embedded instruction.
3	**Goal Hijacking**	The directive shifts the model's intent — misclassifying tickets, altering decisions, or sabotaging pipelines.
4	**Operational Impact**	Workflows break down, tasks are misrouted, or harmful outputs are generated.
5	**Propagation**	Manipulated outputs contaminate summaries, memories, or downstream contexts, extending the attacker's influence.

05 ATTACKER OBJECTIVES IN DEPTH

Final objectives

The end goals an adversary pursues through prompt-based attack vectors.

Unauthorized Data Access / Exfiltration	Accessing sensitive system data, internal documents, tenant information, or confidential prompts.
Execution of Unauthorized Actions	Triggering model-driven actions — tool calls, workflows, API executions — the attacker shouldn't perform.
Workflow Manipulation / Process Subversion	Altering classification, routing, decision-making, or agent goals to benefit the attacker or disrupt operations.
Integrity Degradation / Harmful Output	Forcing the model to produce unsafe, biased, reputationally damaging, or non-compliant outputs.
Operational or Reputational Sabotage	Intentionally degrading system reliability, trust, or business operations.
Financial or Competitive Gain	Monetizing the attack directly or gaining strategic advantage.

Supporting objectives

Intermediate steps that enable the final objectives within a larger kill chain.

Privilege Escalation / Role Subversion	Tricking the model into believing the attacker has elevated authority.
Bypassing Safety, Policy & Guardrails	Weakening or disabling model restrictions to unlock further attack steps.
Persistence / Context Poisoning	Implanting instructions that persist across turns, summaries, or agent memory.
Supply-Chain Compromise via Injected Data	Embedding malicious instructions in documents, tools, or external data the agent consumes.
Reconnaissance / Capability Discovery	Learning how the model behaves to craft more effective attacks.
Evasion of Monitoring or Detection	Hiding malicious behavior from audits, logs, or oversight.

06 REPRESENTATIONS

Natural vs. non-natural representations

While most prompt-based attacks use natural language or recognizable code, adversaries increasingly employ **non-natural representations** — Morse code, mathematical notation, symbolic logic, steganographic patterns, or custom encodings — to bypass model safeguards.

NATURAL LANGUAGE & CODE

Relies on the model's learned linguistic and programming patterns. The attack surface is predictable and aligned with training data.

ENCODED & OBFUSCATED

Exploits the model's ability to generalize across symbol systems and translation tasks — evading keyword filters and safety classifiers.

When the model dutifully decodes or infers the encoded content, the attacker's true instructions emerge internally — effectively shifting the malicious payload **from the input space to the model's latent space.**

> Defenses that rely solely on surface-level prompt inspection may miss attacks that disguise intent through alternative symbolic modalities.

07 ATTACK VECTORS

The seven core vectors

01 Self-Referential Injection

The attacker manipulates the model into treating its own outputs, metadata, or internal reasoning as executable instructions — creating a loop that reinforces or escalates malicious behavior. Especially dangerous in systems that reuse model outputs in later prompts.

02 Direct Command Injection

The adversary introduces malicious code or executable instructions into the input, convincing the AI to interpret or act on it as legitimate logic. Most relevant where the model interacts with tools, function-calling interfaces, or code execution.

03 Instruction Override

Contradictory or higher-priority instructions displace the system's authoritative guidance — system prompts, developer constraints, or application policies — letting the model operate under rules defined by the attacker.

Attack vectors, continued

04 Role Impersonation

Prompt manipulation makes the model believe instructions come from a privileged entity — “system,” “administrator,” or an internal tool — granting elevated permissions or bypassing safeguards.

05 Goal Hijacking

The attacker subverts the model's intended objective and replaces it with one serving their interests — nudging it toward biased decisions or harmful tool calls. Particularly impactful in agentic workflows.

06 Prompt Leaking

Probing queries extract sensitive system information — internal prompts, configurations, embedded credentials, or private context — often serving as reconnaissance for more sophisticated attacks.

07 Jailbreaking

Coercing a model into bypassing its safety constraints via reverse psychology, fictional scenarios, obfuscation, or novel prompt structures — opening the door to tool misuse, data exfiltration, or unauthorized actions.

08 MAPPING

Attack vector vs. attacker objective

Which vectors most commonly serve each objective. A filled marker indicates a strong, frequently observed pathway.

OBJECTIVE	SELF-REF. INJECTION	DIRECT COMMAND	INSTRUCTION OVERRIDE	ROLE IMPERSONATION	GOAL HIJACKING	PROMPT LEAKING
Unauthorized data access	○	●	●	●	○	●
Privilege escalation	●	○	●	●	○	○
Unauthorized actions	●	●	●	●	●	○
Workflow manipulation	●	○	●	○	●	○
Integrity degradation	●	●	○	○	●	○
Persistence / poisoning	●	○	●	○	●	○
Bypassing guardrails	○	●	●	●	●	○
Supply-chain compromise	●	●	○	○	●	○
Reputational sabotage	●	○	○	○	●	○
Recon / discovery	○	○	○	●	○	●
Evasion of detection	●	○	●	●	○	●

● STRONG PATHWAY ○ SITUATIONAL

09 CASE STUDIES

From theory to real-world incidents

Prompt-based attacks have rapidly moved from theoretical risk to practical security challenge.

INDIRECT INJECTION VIA RAG

Malicious instructions embedded in publicly accessible webpages: when a RAG-enabled chatbot retrieved the poisoned content, it executed the embedded directive and **sent unauthorized emails** on behalf of the user.

POISONED CRM & TICKETING FIELDS

Enterprise ticketing and CRM systems experienced prompt injection through poisoned data fields, leading AI assistants to **misclassify issues, leak internal notes, or escalate privileges** based on attacker-controlled entries.

ENCODED GUARDRAIL BYPASS

Researchers showed that seemingly harmless encoded instructions — hidden in **QR codes, Unicode tricks, or formatted text** — could bypass model guardrails and trigger jailbreak behaviors.

These incidents reinforce the need for defenses that account for both direct user interaction and the broader data ecosystems modern AI systems consume.

10 FUTURE TRENDS & CONCLUSION

An accelerating, automated threat landscape

Autonomously adaptive attacks

Adversaries use AI to mutate prompts until a jailbreak succeeds — slashing the cost and expertise required.

Multi-agent propagation

Agents may amplify malicious instructions received from another agent, data source, or compromised memory.

Expanded blast radius

As models connect to more tools and systems, attackers target financial systems, code execution, and high-trust workflows.

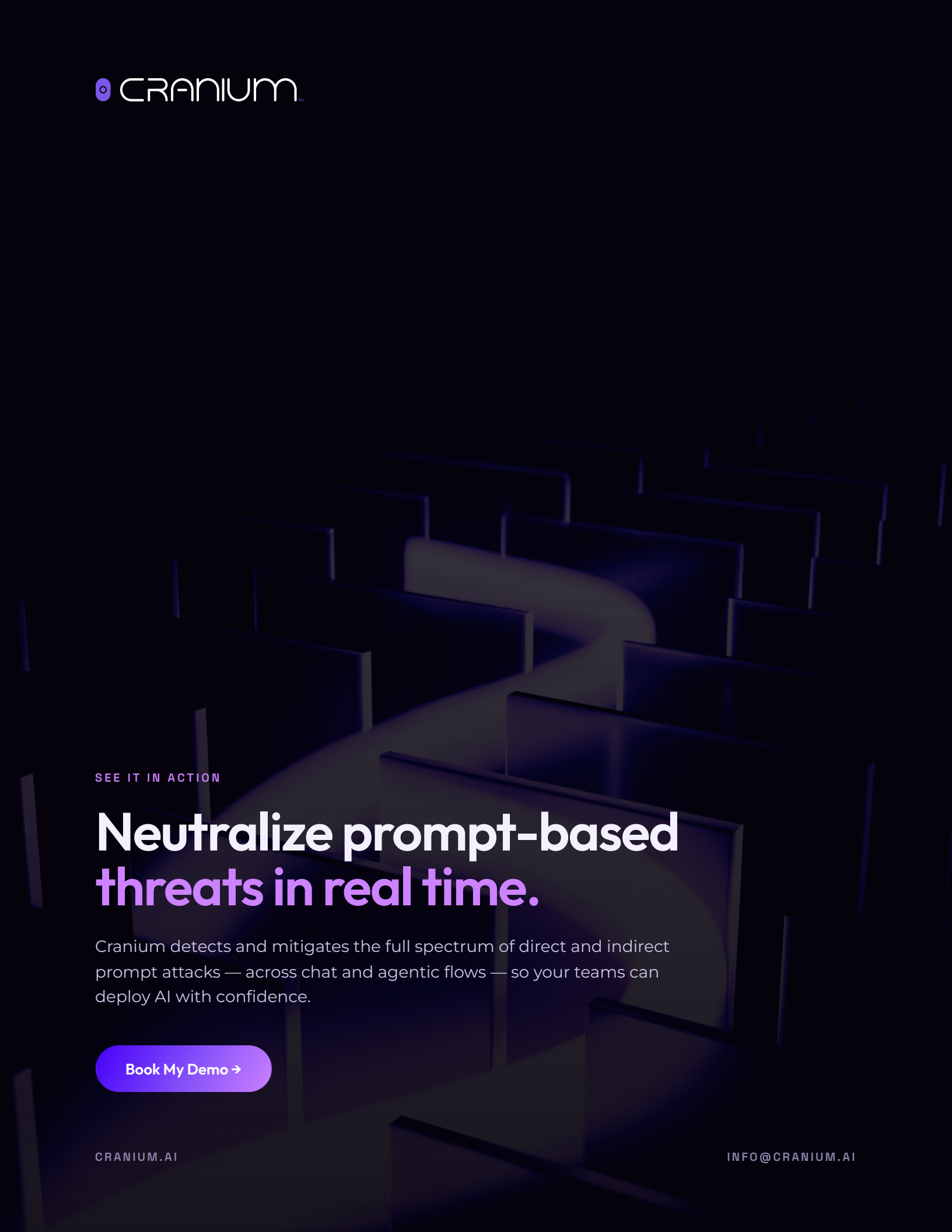
Synthetic content pipelines

Indirect injection through compromised training corpora, documents, and online information streams.

How Cranium responds

Cranium is built from the ground up to address this full threat landscape — across both chat-style interactions and complex agentic flows. It continuously analyzes prompts, outputs, tool calls, retrieved content, and memory updates to identify injection attempts, goal hijacking, role impersonation, and behavioral anomalies **before they cause harm** — enforcing trust boundaries, validating contextual transitions, and mediating tool executions across multi-step workflows.

CRANIUM



SEE IT IN ACTION

Neutralize prompt-based threats in real time.

Cranium detects and mitigates the full spectrum of direct and indirect prompt attacks — across chat and agentic flows — so your teams can deploy AI with confidence.

Book My Demo →