FortressAI & Generative AI Security

How FortressAI governs generative AI use at the kernel level - stopping data leaks to ChatGPT, Copilot, Gemini and other AI tools before they happen.

What is FortressAI?

Short answer: FortressAI is Cyber Crucible's generative-AI data protection product. It enforces policy at the operating system's deepest layer, checking data access on the device and blocking, redacting, or safely allowing information before it can reach ChatGPT, Copilot, Gemini, or any other AI tool.

The problem it solves

Public AI tools are transforming how people work, and simultaneously opening a new path for data to leave an organization. Every prompt carries risk. Employees frequently share sensitive information without realizing they've done anything wrong, and traditional security tools were never designed for this.

Why kernel-level enforcement matters here

FortressAI operates at the same depth as the rest of the Cyber Crucible platform. That means it protects against *any* AI tool — including ones that don't exist yet — rather than maintaining a list of specific applications to police. It doesn't need to understand a new AI product to stop sensitive data from reaching it.

Because enforcement is local, prompt content is never shipped to a cloud service for inspection. The protection doesn't create the exposure it's meant to prevent.

FortressAI is not a separate engine bolted on. It is a policy layer over the same kernel memory behavioral analytics that drive data and identity protection — which is how it can judge not just whether an AI tool is approved, but whether it has been tampered with.

What is Shadow AI, and who inside my organization is creating the risk?

Short answer: Shadow AI is employees using AI tools that IT hasn't approved or doesn't know about. The risk isn't limited to rank-and-file staff — privileged administrators and compromised accounts are often the larger exposure.

Three sources of exposure

- **Employees** using unsanctioned AI tools for everyday productivity.
- **Privileged administrators** whose high-level access means anything they feed an AI tool can include highly sensitive corporate data.
- **Compromised accounts**, where an attacker with valid credentials uses AI tools as an exfiltration channel — traffic that looks like ordinary productivity.

This is already happening

Cyber Crucible has directly observed mass access to large volumes of data at once by employees using AI tools to search for and upload information. This isn't a theoretical future risk being modeled; it's behavior visible in real deployments today.

The third category deserves particular attention. An attacker using AI tools to move data out is difficult to distinguish from an enthusiastic employee — unless enforcement happens at the data-access layer, where intent and policy can be evaluated regardless of who is logged in.

Why do DLP, AI firewalls, and private LLMs fail to stop AI data leaks?

Short answer: Each addresses part of the problem from the wrong layer. DLP can be bypassed, AI firewalls sit at the network edge and are blind to activity on the device, and private LLMs are expensive while doing nothing about employees using public tools anyway.

Where each approach falls short

- **Private LLMs** are costly to build and run, and they don't prevent anyone from opening ChatGPT in a browser tab regardless.
- **AI firewalls** inspect network traffic, which leaves them blind to what happens on the endpoint itself and to insider activity.
- **Traditional DLP** was designed for files and email flows, and is routinely bypassed by paths it was never built to watch.

The upside-down logic problem

Several approaches in this category try to protect data from AI by inserting *another* AI to filter the output. That adds cost and a new dependency without addressing where the data actually leaves — the endpoint, at the moment of access.

FortressAI enforces at the kernel, which is below all of these layers and applies no matter which tool, browser, or extension is involved.

How does FortressAI protect data in a web browser?

> **Beta capability.** Browser-based AI usage detection and enforcement is currently in beta. It is functional and in active use, but evaluate it in your environment before depending on it as a control.

Short answer: FortressAI monitors browser activity from the kernel and evaluates it against policy. If the browser, a website, or an extension tries to access data outside of policy, permission is revoked at the kernel layer — and if the browser shows signs of exploitation, all data rights are revoked and the process is suspended.

The sequence

1. A user opens a web browser — Edge, Chrome, or Firefox.
2. Kernel-level monitoring evaluates the browser and any page or extension activity against the assigned FortressAI policy.
3. If the browser, site, or extension attempts to access data outside policy, permission is revoked at the kernel layer.
4. If the browser shows signs of exploitation, all data rights are revoked and the process is suspended.

Why the browser is the critical control point

The browser is where most generative AI use actually happens, and it's also a heavily targeted attack surface. Cyber Crucible has seen this directly: from Q4 2023 onward, automated responses against Chrome, Edge, and Chromium activity climbed from zero into the thousands, with related CVEs subsequently published by Google and Microsoft. At one company the pattern escalated from a handful of blocked attacks to nearly 4,000 across almost every workstation.

How does FortressAI know an AI tool hasn't been tampered with?

Short answer: Before granting any AI tool access to data, FortressAI assesses whether that tool has been tampered with — checking the integrity of its process and loaded libraries through the same memory behavioral analytics the rest of the platform runs on. An approved tool that has been compromised does not get your data simply because it's on the allowed list.

Approval is not the same as trust

Most AI governance stops at the allow-list: is this application permitted? That's necessary but not sufficient, because an application's identity and its *integrity* are different questions.

`claude.exe`, a Copilot client, a ChatGPT desktop app, or a browser running Gemini can all be legitimately approved — and all be exploited. An attacker who compromises an approved AI client inherits whatever data access that client was granted. An allow-list alone hands it over.

FortressAI therefore evaluates two things before data access:

1. **Is this tool authorized?** (per-tool assessment and policy)
2. **Is this tool still itself?** (has the process or its libraries been tampered with)

Both must hold.

What "tampered with" means here

The check draws on memory analytics — the same foundational sensor layer behind data and identity protection. It looks at the state of the running program and its loaded libraries for signs of injection, in-memory modification, or exploitation.

This matters because modern tradecraft specifically targets trusted processes. Code injected into an approved application inherits its permissions and its reputation. Checking the file on disk proves nothing about what the process has become in memory.

What happens on a failed check

The response follows the same graduated model used across the platform:

- **Authorized and untampered, within policy** — access proceeds.
- **Authorized and untampered, but exceeding policy** — blocked, redacted, or given fake data, depending on your rules. The tool keeps running.
- **Tampered or unknown** — rejected or suspended, per the behavioral engine and your settings.

The practical effect: a compromised AI client is treated as an attacker, not as an approved application having a bad day.

Can I control which AI tools my organization is allowed to use?

Short answer: Yes. FortressAI provides per-tool assessment — granular control over exactly which AI tools are authorized and which are blocked — and location-based enforcement, so policy can be tied to where sensitive data lives rather than to individual applications.

Two complementary controls

- **Per-tool assessment:** decide specifically which AI applications (Gemini, Copilot, and others) are permitted in your environment.
- **Location-based enforcement:** define protection around the data locations that matter — keeping source code in the repository, not in a chatbot — with assignments controlled by users.

The Traffic Light Protocol

FortressAI policy is expressed as a traffic light:

- **Red** — FortressAI blocks the AI tool from accessing the assigned data.
- **Yellow** — access is evaluated conditionally, including dynamic handling of sensitive data. *(The conditional paths that depend on Purview/MIP label enforcement and automatic PII/PDPL anonymization are in beta — see "Which FortressAI capabilities are generally available, and which are in beta?")*
- **Green** — approved use proceeds normally.

This lets organizations adopt AI deliberately rather than choosing between blanket bans that get worked around and open access that leaks data.

Which FortressAI capabilities are generally available, and which are in beta?

Short answer: Core kernel-level enforcement — blocking sensitive data from reaching AI tools, per-tool authorization, and location-based data policy — is generally available. Browser-based AI usage detection and enforcement, Microsoft Purview/MIP sensitivity label enforcement, and automatic PII/PDPL anonymization are currently in beta.

Generally available

- **Kernel-level data protection** — sensitive data is blocked from leaving through ChatGPT, Copilot, Gemini, or any other AI tool, enforced at the operating system's deepest layer.
- **Per-tool assessment** — granular control over exactly which AI tools are authorized and which are blocked.
- **Location-based enforcement** — policy tied to the data locations that matter, so source code stays in the repository rather than in a chatbot.

In beta

- **Browser-based AI usage detection and enforcement** — monitoring and enforcing policy against browser, page, and extension activity.
- **Microsoft Purview / MIP sensitivity label enforcement** — extending existing Purview classification into per-AI-tool policy, so organizations that already classify data don't maintain a second scheme.
- **Automatic PII / PDPL anonymization** — dynamic filtering that strips personally identifiable information from otherwise-allowed prompts. This supports the middle path most organizations want: let people use AI productively, but remove the sensitive elements automatically rather than relying on employees to self-censor.

Beta capabilities are functional and in active use, but should be evaluated in your environment before you depend on them for a control. For current beta availability and enrollment, contact your Cyber Crucible representative.

How does FortressAI help with compliance and audit?

Short answer: FortressAI prevents the leak rather than reporting it afterward, and provides straightforward records of who accessed what and where leaks were stopped — evidence you can show an auditor.

Prevention as a compliance posture

Most compliance exposure from AI use follows the same pattern: sensitive data leaves the organization, and the obligation to investigate, notify, or disclose follows. Blocking the access at the kernel means the disclosure event doesn't occur.

Demonstrating control

Auditors and regulators generally want two things: evidence that a control exists, and evidence that it works. FortressAI supports both — a policy that is technically enforced at the operating system level, and clear records showing where enforcement occurred.

Where this applies

The requirement shows up under many frameworks — HIPAA for patient information, FERPA for student records, GDPR for personal data, and contractual confidentiality obligations. The underlying control is the same in each case: sensitive data must not leave the boundary, and you must be able to prove it didn't.