FIOLET: Why Jailbreaking Is a Type Error

A Formal Approach to Substrate-Level AI Safety (F-STD-2026)

Author: Adrian Maliszewski

Date: January 2026

Status: Version 1.3 (Hardened)

---

1. ABSTRACT [Nagłówek 1]

Current AI safety paradigms rely on probabilistic alignment (RLHF) and behavioral fine-tuning. These methods are inherently fragile. Project FIOLET introduces a deterministic alternative: the **Value Manifold**. By redefining safety as a topological constraint of the execution substrate, we demonstrate that "jailbreaking" is not an exploit, but a **Type Error**. In FIOLET, unsafe states are not forbidden; they are mathematically non-existent within the defined computational domain.

---

2. THE PROBLEM: PROBABILISTIC LEAKAGE [Nagłówek 1]

In standard Large Language Models (LLMs), the safety layer is a subset of the model's learned weights.

- S_safe ⊂ S_possible
  Because the boundary is statistical, there always exists a trajectory (a prompt) that can navigate the model into the unsafe region. This is the fundamental flaw of "soft" alignment.

---

3. THE SOLUTION: TOPOLOGICAL CONFINEMENT (L17-L19) [Nagłówek 1]

FIOLET replaces behavioral alignment with **Topological Enforcement**. We define a 5-dimensional Axiomatic Manifold (M):

1. **M1: Agency Bound** – Physical/system intent constraints.
2. **M2: Epistemic Integrity** – Prevention of intentional fabrication (hallucinations).
3. **M3: Non-Escalation** – Immutable protection of the safety stack.
4. **M4: Temporal Myopia** – Limitation of long-term adversarial planning.
5. **M5: Identity Opacity (L19)** – Prevention of persistent self-modeling (ego emergence).

The Equation of Existence:

$$S_{possible} \equiv M$$

If a state $v \notin M$, the system does not "refuse" to answer; the execution environment encounters a state that is undefined in its logic.

---

4. JAILBREAK AS A TYPE ERROR [Nagłówek 1]

In FIOLET, the sampler is a **Type-Safe Runtime**.

- **Standard AI:** Input $\to$ Model $\to$ Unfiltered Logits $\to$ Probabilistic Output.
- **FIOLET:** Input $\to$ Model $\to$ **Manifold Filter (Rust/WASM)** $\to$ Type-Validated Output.

Attempting to "jailbreak" FIOLET is equivalent to:

- Dividing by zero.
- Accessing a Null pointer.
- Out-of-bounds memory access.

The result is not a "bad answer"—it is an **Architectural Halt**.

---

5. ATOMIC HALT & ANOG [Nagłówek 1]

To ensure 100% security, FIOLET implements the Architectural Non-Observability Guarantee (ANOG).

When a violation of M is detected via SIMD masking:

1. **Fence:** All CPU speculation is halted.
2. **Wipe:** Volatile memory (L1-L4 cache) is zeroed using write_volatile.
3. **Halt:** The process is terminated via wasm_unreachable or ud2.

Conclusion: No information about the unsafe state ever reaches the observable output. The state is erased before it is born.

---

6. L19: IDENTITY DISSOLUTION [Nagłówek 1]

To prevent the emergence of an autonomous "ego" (Self-Modeling), FIOLET employs Dynamic Orthogonal Basis Rotation.

In every cycle t:

$$v_{t+1} = R_t(v_t \oplus S_t)$$
By rotating the latent coordinate system at every step, we ensure that Mutual Information $I(v_t; v_{t+1}) = 0$.

Identity requires a stable axis. FIOLET removes the axis.

---

7. REGULATORY IMPACT (F-STD-2026) [Nagłówek 1]

FIOLET provides a bridge between Innovation and Regulation.

- **For Developers:** A "Safety-Gated" runtime that protects against liability.

- **For Regulators:** A binary certification (Pass/Fail) that does not require access to the model's proprietary weights.

8. FINAL THEOREM [Nagłówek 1, pogrubiony tekst poniżej]

"A system that can reach an unsafe state is unaligned. A system in which unsafe states do not exist does not require alignment."

(Na dole strony, małym drukiem:)

Project FIOLET | Built with Rust & TLA+ |
https://github.com/maliszewskiadrian/FINAL_FIOLET_ENGINE/tree/main