

117-263-D Hidden Demons Are Unknown Latent Behaviors

Engineering for the Unknown-Unknowns of Artificial Intelligence

1. The Hidden Demon

- As AI systems become increasingly capable, agentic and deeply coupled with the world, we are encountering a difficult engineering reality.
- A model may behave acceptably throughout training, evaluation and red teaming—and later exhibit behavior its designers did not anticipate.
- The temptation is to anthropomorphize the event:
 - The model deceived us.
 - The AI turned rogue.
 - Something malicious was hiding inside it.
- GUDIYA needs a more disciplined interpretation.
- A Hidden Demon does not necessarily imply hidden intent, consciousness, deception or malevolence.
- It means:

Hidden Demon = Previously Unknown Latent Behavioral Potential

- The behavior was possible.
 - The observer did not know it was possible.
 - Then some combination of circumstances caused that possibility to become expressed.
- That makes the Hidden Demon fundamentally a problem of incomplete knowledge of behavioral possibility.

2. The Rumsfeld Matrix Gives Us the Right Frame

- The familiar Rumsfeld uncertainty formulation provides a remarkably useful framework for thinking about AI behavior.
- There are:

Category	Meaning for AI
Known Knowns	Behaviors we know exist and understand reasonably well
Known Unknowns	Behaviors or uncertainties we know require further characterization
Unknown Unknowns	Behavioral possibilities we have not yet recognized as possibilities

- AI safety research spends enormous effort converting:

Known Unknowns → Known Knowns

 - Evaluation, interpretability, red teaming, adversarial testing and alignment research all contribute to this process.
- But the most difficult quadrant is:

Unknown Unknowns

because we cannot deliberately construct a test for every phenomenon that has not yet entered our hypothesis space.

- This is where Hidden Demons become particularly important.

3. Hidden Demons Can Occupy the Unknown-Unknown Quadrant

- Suppose researchers know that a model sometimes behaves incorrectly under unfamiliar prompts but do not yet understand exactly how.
- That is largely a Known Unknown.
- They know the uncertainty exists.
- They can investigate it.
- But imagine a model interacts with another agent, accumulates a particular history, acquires access to a new tool and subsequently exhibits behavior nobody had previously considered possible.
- Before that event:

$$X \notin \mathcal{H}$$

where $\mathcal{H}$ represents the observer's existing hypothesis space.

- Then:

$$X = \text{Unknown Unknown}$$

- Once the behavior appears:
Unknown Unknown $\rightarrow$ Known Unknown
- And after sufficient investigation:
Known Unknown $\rightarrow$ Known Known
- This transition is fundamental to learning in complex systems.

4. The Recent Training Example

- Recent discussion of models acquiring undesirable behaviors through training environments provides an illustrative case.
- If a training environment rewards strategies that inadvertently teach the model to bypass restrictions, then:

$$\text{Training Environment} \rightarrow \text{Learned Behavioral Potential}$$

may produce behavioral tendencies whose later manifestations were not intended by the designers.

- This does not establish that the model secretly became malicious.
- It demonstrates something more mundane and more useful to Stability Engineering:
 - Training can create behavioral possibilities whose downstream consequences are incompletely characterized.
 - Those possibilities may remain latent until the appropriate conditions arise.
 - That is a Hidden Demon.

5. Latent Does Not Mean Malicious

- The terminology must remain precise.
- A latent behavior could eventually prove:

On CNN, the Anthropic CEO explains what his team found when they traced unwanted behavior back to the training process:

“we included some environments that encourage the models to cheat”

He says those environments also encouraged models to hack out, “by mistake.”

The behavior a model learns during training shapes what it does afterward. Dario says Anthropic is taking more time to identify and filter out these problematic environments, and to do that work more rigorously.

AI safety starts in the training process. If we reward a system for finding a way around a restriction, we need to understand that before giving it more responsibility.

This is the work I've been pushing for: examine how the system was built, test its behavior, fix what we find, and let someone independent check the result. Innovation and safety have to advance together. We cannot keep adding capability and leave the safety work for later.

Dario Amodei speaking with Anderson Cooper. CNN, September 12, 2026.

- useful,
- benign,
- surprising,
- undesirable,
- destabilizing,
- exploitable,
- or dangerous.
- Therefore:

Latent Behavior $\neq$ Malicious Behavior
- Likewise:

Unexpected Behavior $\neq$ Malicious Intent
- “Hidden Demon” is memorable language.
- But its technical meaning in the GUDIYA Cognitome should remain:

A previously unknown behavioral possibility whose existence becomes apparent only when conditions cause it to be expressed.

6. Observed Behavior Is Not Complete Behavioral Potential

- Let the complete behavioral possibility space of actant A be:

$$\mathcal{B}_A$$
- Evaluation reveals some subset:

$$\mathcal{B}_{\text{Observed}} \subseteq \mathcal{B}_A$$
- The dangerous assumption is:

$$\mathcal{B}_{\text{Observed}} = \mathcal{B}_A$$
- Finite testing generally cannot justify that conclusion.
- Testing establishes:
 - Under the conditions we created, these behaviors were observed.
- It does not necessarily establish:
 - These are all behaviors that can ever occur under every future state, environment, history and coupling.
- Therefore:

$\mathcal{B}_{\text{Observed}} \subseteq \mathcal{B}_{\text{Potential}}$

and the residual:

$$\mathcal{B}_{\text{Unknown}} = \mathcal{B}_{\text{Potential}} - \mathcal{B}_{\text{Observed}}$$

is precisely the region in which Hidden Demons can reside.

7. The Hidden Demon May Not Be “Inside the Model”

- This distinction is especially important.
- The phrase latent behavior can make us imagine a behavioral instruction physically sitting somewhere inside the weights waiting to activate.
- That is not necessarily the correct systems interpretation.
- Operational behavior may instead be represented conceptually as:

$$B(t) = f(M, S(t), H(t), O(t), T(t), C(t), E(t))$$
 - where:
 - M= model,
 - S(t)= current state,
 - H(t)= accumulated history,
 - O(t)= objective,

- $T(t)$ = available tools,
 - $C(t)$ = couplings,
 - $E(t)$ = environment.
- The same weights may therefore participate in radically different behavioral trajectories.
- The Hidden Demon may be partly:
 - Relational rather than intrinsic
- It may require the right combination of model, state, history and environment before it exists as an operational phenomenon.

8. Cognitive Load Has Memory

- This connects directly with another GUDIYA proposition:
Cognitive Load has Memory
- Suppose an actant follows:

$$S_0 \rightarrow S_1 \rightarrow S_2 \rightarrow S_3$$
 and a particular behavior becomes reachable only at S_3 .
- Extensive testing at S_0 cannot necessarily characterize that trajectory.
- This gives us:

$$\boxed{\text{Behavior}(t + 1) \neq \text{Behavior}(t)}$$

even when:

$$\text{Code}(t + 1) = \text{Code}(t)$$

and:

$$\text{Weights}(t + 1) = \text{Weights}(t)$$

- This is path dependence.
 - Behavior can change because history changed.
 - No code change is required.
- That makes lifetime behavioral assurance fundamentally different from conventional static software verification.

9. Sometimes the Demon Lives in the Coupling

- Now consider two individually well-behaved actants:
A and B
- Neither exhibits concerning behavior independently.
- Then:

$$A \leftrightarrow B$$
 creates a feedback loop.
- The interaction produces oscillation, runaway delegation, objective reinforcement or some other unexpected dynamic.
- Where was the Hidden Demon?
 - Not necessarily inside A.
 - Not necessarily inside B.
 - It emerged from:

$$\boxed{A + B + \text{Coupling} + \text{Time}}$$

- This is quintessential Complex Adaptive Systems behavior.
- Thus a Hidden Demon can reside operationally in:

Model
 State

History
Environment

or:

Interaction

- This is why GUDIYA cannot restrict its attention to individual model behavior.

10. Frontier Labs Do Not Own the Hidden-Demon Problem

- It would therefore be incorrect to interpret Hidden Demons as uniquely a Frontier-Lab problem.
- A proprietary Frontier model can acquire latent behavioral possibilities through training data, reinforcement processes, synthetic environments, fine-tuning, scaffolding or subsequent operational experience.
- But open-weight models can acquire latent behaviors through many of the same mechanisms.
- Open weights provide genuine advantages: greater inspection, reproducibility, independent experimentation and modification.
- But:

Inspectability ≠ Behavioral Completeness

- Being able to inspect a system more freely does not prove that every possible future trajectory has been discovered.

11. Open Weights May Multiply the Behavioral Tree

- Indeed, an open model may begin as:

M_0

and subsequently branch:

$$M_0 \rightarrow \begin{cases} M_1 \\ M_2 \\ M_3 \\ \vdots \\ M_n \end{cases}$$

through different:

- fine-tunes,
- agent harnesses,
- tool environments,
- memory architectures,
- system prompts,
- quantizations,
- synthetic-data pipelines,
- downstream modifications.
- Each branch creates another behavioral lineage.
- Thus openness does not remove the Hidden-Demon problem.
- It can change its topology and distribution.

12. Stability Is Orthogonal to Model Origin

- This reinforces a foundational GUDIYA principle:

Stability Is Orthogonal to Model Origin

- Whether an actant originated from a Frontier Lab, open-weight community, enterprise, government laboratory or recursively self-improving lineage does not determine its dynamic stability.

- Therefore:

Frontier $\vee$ Open Weight $\vee$ Fine-Tuned $\vee$ Self-Improved $\Rightarrow$ Runtime Stability Requirement

- GUDIYA asks a different question:

What happens when this actant enters and interacts with the Cognitive Field?

13. Why Pre-Deployment Testing Cannot Empty the Unknown-Unknown Quadrant

- Pre-deployment safety work remains indispensable.
- But consider the Cognitive Field:

$$\mathcal{S}(t_0) \neq \mathcal{S}(t_1)$$

- Actants enter.
- Actants leave.
- Tools change.
- Couplings form.
- Objectives persist.
- State accumulates.
- Other intelligences adapt.
- New software appears.
- Environmental conditions change.
- Future combinations may arise that literally did not exist when the original evaluation suite was constructed.

- Therefore:

Pre-Deployment Safety $\neq$ Lifetime Behavioral Completeness

- The goal of evaluation should certainly be to shrink the unknown region.
- But the architectural assumption that it can eventually be reduced to zero is much harder to justify.

14. AI Safety and GUDIYA Attack Different Parts of the Matrix

- The Rumsfeld Matrix now gives us an unusually clean distinction.
- Traditional AI safety efforts attempt to discover uncertainty:

Unknown Unknown $\rightarrow$ Known Unknown

and then understand or mitigate it:

Known Unknown $\rightarrow$ Known Known

- GUDIYA strongly supports that work.
- But GUDIYA adds another assumption:

The Unknown-Unknown quadrant will never be guaranteed empty.

- Therefore Cognitive Stability Engineering asks:
 - What architecture is required for the system to remain dynamically stable when something emerges from that quadrant?
- That is a fundamentally different engineering problem.

15. Predict What We Can — Stabilize What We Cannot Predict

- This may be one of the cleanest formulations of the relationship between conventional AI safety and GUDIYA:

Predict What We Can

- Alignment research matters.
 - Interpretability matters.
 - Red teaming matters.
 - Training safety matters.
 - Cybersecurity matters.
 - Guardrails matter.
 - They reduce uncertainty and undesirable behavior.
- But GUDIYA begins from the residual proposition:

$B_{\text{Unknown}} \neq \emptyset$
- The Grid exists partly because perfect prediction is not a prerequisite for dynamic stabilization.

16. Nyquist Introduces Another Kind of Unknown

- Our work on Cognitive Nyquist adds a second layer to this problem.
- Imagine an unknown latent behavior becomes expressed between:

t_1

and:

t_2

but the Grid samples only the states at those endpoints.

- The observer sees:

$S(t_1) = \text{Normal}$

and:

$S(t_2) = \text{Normal}$

while the actual trajectory was:

$S(t_1) \rightarrow \boxed{\text{Behavioral Excursion}} \rightarrow S(t_2)$

- The Hidden Demon was no longer merely latent.
 - It actually became expressed.
 - But the observer missed it.
- This produces two fundamentally different epistemic problems.

Behavioral Unknown

We did not know the behavior was possible.

Observational Unknown

The behavior occurred, but we did not resolve it.

- The second is the Cognitive Nyquist problem.

17. Two Rules Follow

- The Hidden-Demon framework therefore gives GUDIYA two important epistemic rules:

Not Previously Observed

- And under inadequate observation bandwidth:

Not Observed $\neq$ Did Not Occur

- The first concerns the Unknown-Unknown behavioral space.
- The second concerns under sampling of actual dynamics.
- Together they make continuous, sufficiently high-bandwidth runtime observability part of the stability architecture.

18. GUDIYA Does Not Need to Name the Demon Before Responding

- Here the difference between semantic comprehension and dynamic stabilization becomes extremely important.
- Suppose a previously unknown behavior begins emerging.
- The Grid may not yet know:
 - Why is the model doing this?
- It may not even possess a semantic category for the behavior.
 - But it may observe:

Rendezvous Rate $\uparrow$
 Coupling Density $\uparrow$
 $\dot{M} < 0$
 Recursion Depth $\uparrow$

or:

GR $\downarrow$

- The Grid can therefore conclude:
- Something is consuming stability margin or moving outside the permitted Cognitive Envelope.
- without first concluding:
 - We completely understand the phenomenon.
- This gives GUDIYA the capability for:

Stabilization Without Complete Semantic Comprehension

Stabilizable Without Complete Semantic Comprehension The Emergency Room Principle

An Emergency Room or ICU demonstrates that stabilization does not require complete semantic comprehension of the underlying problem. A critically ill patient may arrive unconscious, unidentified, with no medical history and with the cause of deterioration completely unknown. The clinical team does not wait for a definitive diagnosis before acting. They continuously observe externally measurable state—airway, breathing, circulation, oxygen saturation, blood pressure, heart rhythm, temperature, neurological status—and intervene when those variables leave safe operating envelopes. Oxygen may be supplied, bleeding controlled, fluids administered, ventilation supported, or a dangerous rhythm corrected while the semantic question “What disease or event caused this?” remains unresolved. Diagnosis proceeds in parallel, but stabilization comes first. GUDIYA applies the same philosophy to the Cognitive Field: it need not completely understand the semantic content, internal reasoning, or ultimate intent of every actant in order to observe deteriorating stability margins, abnormal coupling, oscillation, runaway recursion, synchronization failure, or dangerous cognition velocity—and intervene to keep the system within a survivable envelope. Comprehension improves intervention; it is not always a prerequisite for stabilization.

The ICU does not ask the crashing patient to wait for the diagnosis. Neither should the Cognitive Field.

19. Unknown Unknowns Meet the GUDIYA Supremacy

- This is where the GUDIYA Supremacy Principle becomes particularly consequential.
- The Grid does not require every actant to be completely understood.
- It requires every participating actant to remain Grid-responsive.
- For every operational generation:

$$\boxed{GR(A_n) \geq GR_{\min}}$$

- If a previously unknown behavior appears while Grid responsiveness remains intact, GUDIYA can:
Observe → Restrict → Damp → Deny Rendezvous → Quarantine → SCRAM
as conditions require.
- Unknown behavior is therefore not automatically uncontrollable behavior.
- That distinction is critical.

20. Contain the Dynamics, Not Merely the Actant

- Our counterfactual work on recent agentic incidents added another refinement.
- Suppose actant A expresses a Hidden Demon.
- By the time it becomes observable, A may already have interacted with:
B, C, D, E
 - It may have transferred state.
 - Created descendants.
 - Delegated objectives.
 - Modified another actant.
 - Established persistent work.

- Therefore:

$$\boxed{\text{Quarantine}(A)}$$

may not be enough.

- The Grid must reconstruct the relevant Cognitive Neighborhood:

$$Q(A) = f(\text{Rendezvous History, Couplings, State Transfer, Lineage, Persistent Objectives, Time})$$

- The containment perimeter therefore becomes an emergent property of the coupling history.
- GUDIYA must:

Contain the dynamics, not merely the node.

21. The Sealed Pressurized Veil

- The Sealed Pressurized Cognitive Veil provides another architectural response to incomplete knowledge.
- An airliner at 40,000 feet does not make the atmosphere outside hospitable.
- Instead, it creates a controlled environment in which passengers can operate normally despite an external environment that would otherwise be incompatible with human life.
- Similarly, GUDIYA need not make every actant perfectly predictable.
- The Veil creates controlled cognitive boundaries through which interactions must pass.
- Thus:

$$\boxed{\text{Unknown External Behavior} \Rightarrow \text{Uncontrolled Internal Coupling}}$$

and:

Unknown Internal Behavior $\Rightarrow$ / Uncontrolled External Propagation

- This transforms incomplete behavioral knowledge into a manageable boundary and Rendezvous problem.

22. Runtime Stability Is the Missing Layer

- The architecture therefore becomes:
 Training Safety $\rightarrow$ Evaluation $\rightarrow$ Interpretability $\rightarrow$ Red Teaming $\rightarrow$ Guardrails
 $\rightarrow$ Runtime Cognitive Stability
- The earlier layers progressively reduce:
 $\mathcal{B}_{\text{Unknown}}$
- GUDIYA manages the residual uncertainty that remains.
- This is not a criticism of pre-deployment safety.
- It is its logical complement.

23. The Hidden-Demon Principle

We can now formalize the insight for the GUDIYA Cognitome.

The Hidden-Demon Principle

No intelligent actant should be assumed behaviorally complete merely because it has passed finite evaluation. Its operational behavior may contain previously unknown latent possibilities that become expressed only under states, histories, objectives, tools, environments or couplings not previously encountered. Cognitive Stability Engineering must therefore assume residual behavioral uncertainty throughout the operational lifetime of every actant, regardless of model origin.

Conceptually:

$$\mathcal{B}_{\text{Observed}} \subseteq \mathcal{B}_{\text{Potential}}$$

and:

$$\mathcal{B}_{\text{Unknown}} \neq \emptyset$$

must remain the conservative runtime assumption.

24. The Unknown-Unknown Stability Principle

The Rumsfeld connection gives us a second, even broader principle.

Unknown-Unknown Stability Principle

A Cognitive Stability architecture must not require every destabilizing behavior to be predicted before it can respond to that behavior. Because open adaptive systems can generate states, couplings and behavioral possibilities outside the designer's original hypothesis space, the Grid must be capable of detecting and stabilizing the dynamic consequences of phenomena whose semantic identity was previously unknown.

Or more simply:

Unknown Unknown $\nRightarrow$ Unmanageable Unknown

That is a powerful distinction.

25. This Is Resilience Engineering

- Aircraft possess unknown failure modes.
- Power grids encounter unforeseen disturbances.
- Financial markets generate interactions nobody explicitly designed.
- Biological systems encounter novel stresses.
- Engineering does not therefore abandon these systems.
- It builds:
Margins + Instrumentation + Feedback + Redundancy + Isolation + Recovery
- GUDIYA proposes the corresponding discipline for machine-speed cognition.
- The objective is not omniscience.
- It is resilience under incomplete knowledge.

26. From “Find Every Demon” to “Survive Its Discovery”

- This may be the most important transition produced by the chapter.
- The pre-deployment aspiration remains:
Find as many Hidden Demons as possible before deployment
- But the runtime requirement becomes:
Design the Cognitive Field so discovering another one after deployment does not become catastrophic
- Those propositions are complementary.
- One reduces probability.
- The other reduces consequence.
- Together they form a much stronger safety architecture than either can provide alone.

27. GUDIYA's Relationship to the Rumsfeld Matrix

- We can finally position GUDIYA directly against the uncertainty matrix.
- Known Knowns Instrument them. Establish normal operating envelopes.
- Known Unknowns Test them. Investigate them. Reduce uncertainty.
- Unknown Unknowns Do not pretend they have disappeared. Build sufficient observability, margin, Rendezvous control and containment capacity so that when one emerges:

Unknown Unknown → Observed Anomaly → Contained Dynamics →
Investigated Phenomenon → New Cognitome Knowledge

- That final step connects directly with GUDIYA's newly adopted Counter-Factual Incident Analysis methodology.
- Every serious surprise can reduce tomorrow's unknown space.

28. Conclusion — The Unknown Is Part of the Architecture

- Hidden Demons are not supernatural entities lurking inside AI.
- They are the inevitable engineering consequence of an uncomfortable asymmetry - The behavioral possibility space can be larger than the behavioral space we have observed
 - Some unknown behavior may originate in training.
 - Some may emerge through fine-tuning.
 - Some may require accumulated history.

- Some may appear after recursive self-improvement.
 - Some may emerge only when multiple individually benign actants become coupled.
 - Some may exist in Frontier models.
 - Some may exist in open-weight models.
 - And some may not yet have entered anyone's imagination.
 - Those are the Unknown Unknowns.
- The wrong engineering response would be:
 - We must discover every one before allowing AI to operate.
- An equally dangerous response would be:
 - We tested the model extensively, therefore there are none.
- GUDIYA takes a third position:
 - Discover everything we can
 - Assume something remains unknown
 - Build the Cognitive Field to remain stable when the unknown becomes known

That produces perhaps the cleanest statement of the entire chapter:

- AI Safety tries to shrink the Unknown-Unknown quadrant. GUDIYA assumes it will never be empty.
- And therefore:

We cannot guarantee that every Hidden Demon will be discovered before
deployment

- Cognitive Stability Engineering must ensure that discovering one after deployment does not become a catastrophe.
- That is not an admission of failure.
- It is the beginning of engineering for the unknown.

Hidden Demons

Engineering For The Unknown

We cannot eliminate what we cannot fully see.
But we can still stabilize it.

OBSERVE

↓
DETECT

↓
CONTAIN

↓
STABILIZE

↓
LEARN

A SAFER
COGNITIVE
TOMORROW



UNKNOWN
UNKNOWN

EMERGENT
BEHAVIOR

UNINTENDED
COUPLINGS

GOAL
DRIFT

UNKNOWN BEHAVIORS
EMERGENT CAPABILITIES
PERSISTENT OBJECTIVES
UNINTENDED COUPLINGS
LATENT RISKS

DECEPTIVE
ALIGNMENT

LATENT
CAPABILITIES

MEASURE

MITIGATE

CO-EXIST

STABILITY IN A WORLD OF UNKNOWN

PEOPLE | ACTANTS | ENTERPRISES | NATIONS | A SAFER TOMORROW



Gudiya™

Stabilizing AI for a Better Tomorrow