

An Agentic System for Cyber Resilience

Research Team

PI: Tom McDermott

Co-PI: Kelli Esser

Co-PI: Stephen Adams

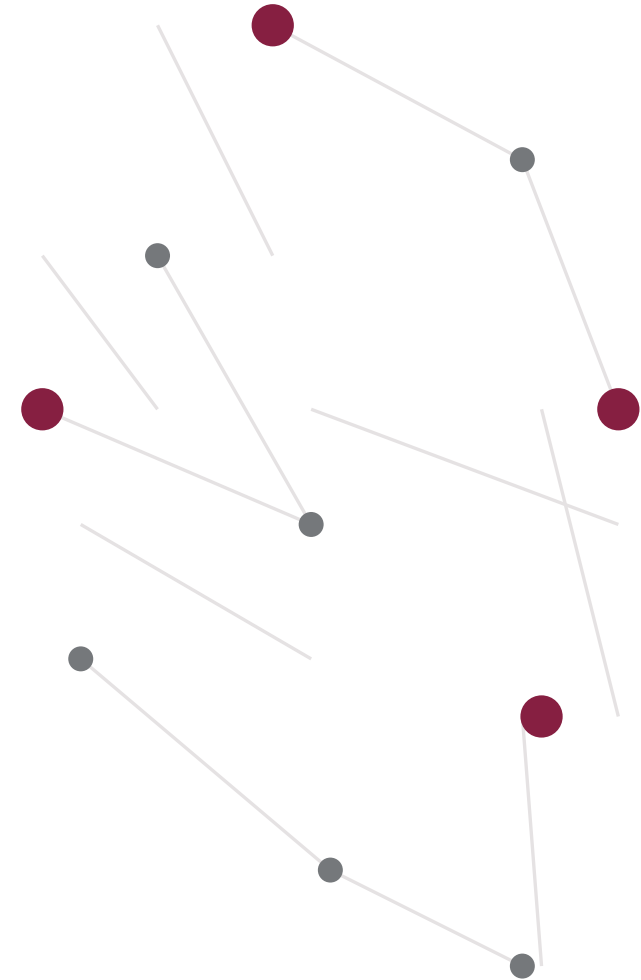
Megan Clifford

Alex Kyer

Brian Walheim

Mary Nerayo

Julie Malherek



What cyber resilience means

Cyber resilience is the ability of a system to anticipate, withstand, recover from, and adapt to adverse cyber conditions — not just prevent intrusions.

Prevent

Stop known threats before they reach the system

Mitigate

Limit damage when a threat gets through

Recover

Restore mission function after degradation

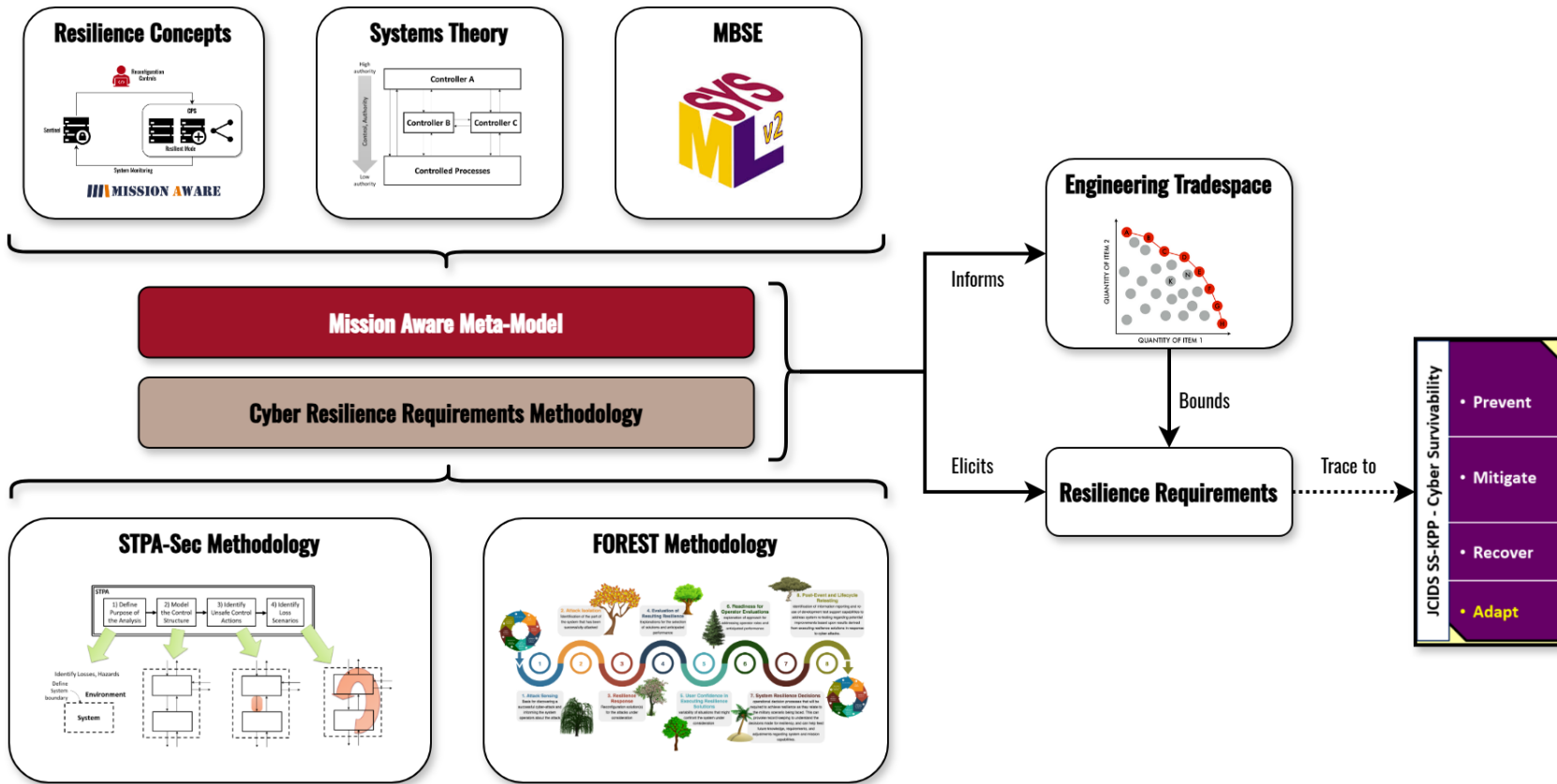
Adapt

Adjust the system so the same threat doesn't work twice

The four cyber survivability capabilities SCRE engineers for (CSA-07–CSA-10) (McDermott et al., 2026)

Engineering resilience in from the start — not bolting on controls after the fact — is what SCRE and STPA-Sec are built to do.

SCRE: one loss-driven chain, four building blocks



CRRM

Risk-based backbone linking mission-level loss scenarios to design-time mitigations

Clifford, Sherburne, Horowitz, & Beling, 2024

Mission Aware

Explicit resilience mechanisms — often “sentinel” patterns that sense and reconfigure

Beling, Clifford, Sherburne, McDermott, & Horowitz, 2024

STPA-Sec

Loss-driven control-theoretic analysis — losses → hazards → UCAs → scenarios

Fleming, 2024; Young & Leveson, 2014

FOREST

Process meta-model: sense, isolate, respond, adapt, verify — across 8 TREES

Clifford, Sherburne, Horowitz, McDermott, & Beling, 2024

Doing this by hand doesn't scale

The problem

- Manual analysis is slow — a full STPA-Sec and resilience-pattern pass takes real time, repeated for every program and every update.
- Expertise is scarce — SCORE is novel enough that analysts trained in STPA-Sec, resilience patterns, and SysML modeling together are rare.
- Consistency is hard — four frameworks (CRRM, Mission Aware, STPA-Sec, FOREST) have to work together the same way regardless of who's running the analysis.
- No reusable starting point — resilience knowledge lives in prose and one-off examples, not a queryable library.

Why a copilot

- Encodes each framework's steps as a bounded agent, so the sequence is applied the same way every time.
- Human-in-the-loop keeps the analyst's judgment in the review — the AI handles structure and drafting, not decisions.
- Scales scarce expertise across programs, instead of requiring every team to have a trained SCORE analyst.

A copilot doesn't replace the analyst — it lowers the barrier to applying a novel, multi-framework methodology consistently, at speed.

New Accomplishments

1

Resilience Pattern Catalog

Vendor-agnostic SysML v2 — patterns defined once, versioned, queried, and instantiated in program models.

Sentinel patterns built around observe—evaluate—act control loops. Examples:

Resource Introspection (PAT.1) · TPM

Attestation (PAT.4) · Diverse Redundancy

(PAT.5) · Unsafe Action Containment (PAT.8) ·

Privilege Reduction (PAT.9)

(McDermott et al., 2026; Esser, McDermott, & Adams, 2025)

2

SCRE Co-Pilot

A prototype that treats agentic AI as an engineered workflow — not a general-purpose chatbot.

Specialized, bounded agents per analysis stage: system description, losses, hazards, control structure, UCAs, loss scenarios, resilience-pattern scoring, candidate cyber survivability requirements.

Approval, modification, and conversational support keep the engineer in control of the baseline.

Bounded agents, not a general-purpose chatbot

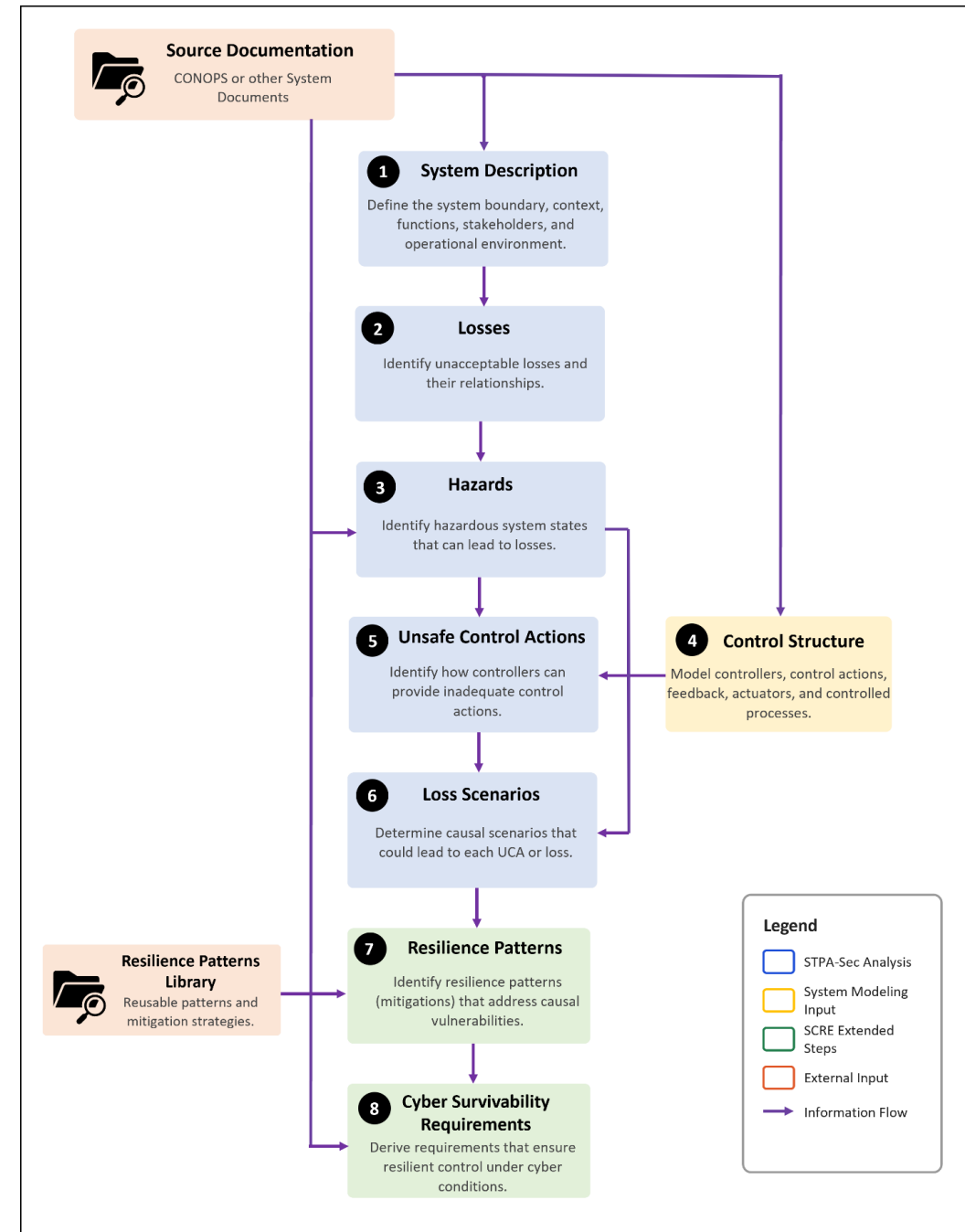


■ ARCHITECTURE

One pipeline, one agent per analysis step

- Orchestrated in LangGraph¹ — each agent is a node in a shared StateGraph.
- Conditional edges route on state/feedback; loop-backs allow revisiting a stage.
- Structured JSON with persistent IDs (SYS-###, L-###, H-###, UCA-###, LS-###) — explicit traceability instead of natural-language matching.
- Same underlying model (Gemma 4 E2B² via Ollama³) for every analytical agent — specialization comes entirely from prompts and supplied context.

LangChain Inc. (n.d.). LangGraph [Software documentation]. <https://docs.langchain.com/oss/python/langgraph/overview>
Gemma Team, Google DeepMind. (2026). Gemma 4 technical report. arXiv:2607.02770.
Ollama. (n.d.). Ollama [Software]. <https://ollama.com>



The scaffold behind every agent's prompt

Every analytical agent's prompt follows the same 7-part structure — grounded in established prompt-engineering literature [Sahoo et al., 2024; Schulhoff et al., 2024].

Role & objective — Each agent's role and scope are fixed and non-overlapping.

e.g. Control Structure Agent reconstructs the control structure; UCA Agent is scoped only to UCA identification.

Grounded definitions — Domain definitions fix STPA-Sec terms rather than the model's own reading of them.

e.g. the Hazard Agent is told explicitly how a hazard differs from a failure, an attack, and a UCA.

Evidence boundaries — Each agent sees only the upstream artifacts relevant to its own stage.

e.g. the Hazard Agent receives just the System Description and Loss Analysis — nothing else.

Procedural decomposition — Complex tasks are broken into domain-specific steps, not open-ended reasoning. [Wei et al., 2022]

e.g. the Loss Scenario Agent examines one UCA at a time and identifies only the causal conditions that explain it.

Evidence gating — Positive instructions are paired with explicit bans on unsupported content.

e.g. the UCA Agent is prohibited from inventing thresholds, response windows, or attacker capabilities.

Self-verification — Each agent runs domain-specific checks on its own output before it's emitted. [Madaan et al., 2023]

e.g. the Control Structure Agent verifies controller authority and actuator relationships, confirming nothing was invented just to satisfy the schema.

Traceable output schema — A required JSON structure with standardized IDs makes every output machine-checkable. [OpenAI, 2024]

e.g. SYS-###, L-###, H-###, UCA-###, LS-###, CSR-### let later stages reference exact artifacts, not approximate text matches.

Guided one stage at a time, with a chat assistant on call

You move through the nine stages in order — the next stage doesn't unlock until the current one is reviewed.



STPA-Sec Expert

Methodology - Insights - Read-only

Read-only

How does STPA work?

Direct Answer

System-Theoretic Process Analysis for Security (STPA-Sec) is a methodology used to analyze complex systems by focusing on the control structure and the interactions (control loops) between system components, rather than focusing solely on component vulnerabilities. The primary goal of STPA is to identify the root causes of system

ASK THE EXPERT

What is SCRE? How does STPA work? Explain loss scenarios What makes a good hazard? Review my artifacts for gaps How do resilience and safety relate? Ask about SCRE methodology or your analysis...

ility posture to ensure the integrity of control ion and modification.

STPA-Sec Expert

Read-only chat with the SCRE / STPA-Sec expert — can review artifacts for gaps

L Losses

10 items

L-001

Loss of life or serious injury to personnel or civilians

PROPERTY	VALUE
Affected Stakeholders	SH-001 Mission Operators SH-002 Deployment Technicians SH-003 Command and Control (C2) Center
Justification	Failure of Fire Obstacle Munitions (F-001) or monitoring (F-002) leading to physical harm to personnel operating the system or in the operational environment.

L-002

Loss of mission capability

L-003

Loss of or damage to mission-critical assets

L-004

Environmental damage

L-005

Loss of sensitive information

Every artifact is a card — expand, edit, or delete, with linked upstream IDs visible

Every artifact needs a decision

Approve

Accept the agent's proposal as-is and move on.

Edit

Correct or extend the content, then continue with your version.

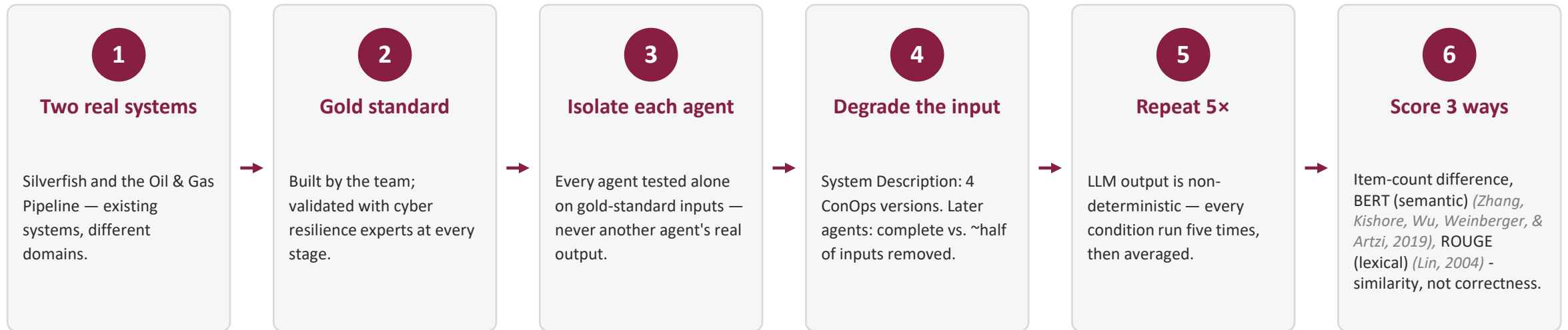
Reject

Discard it — nothing is added to the record.

The chat assistant explains and reviews, but never edits the record itself — only an Approve, Edit, or Reject from the analyst changes an artifact.

A test built to withstand scrutiny

Four deliberate controls, each closing a specific way this kind of test could mislead — not just “we tried it and it worked.”

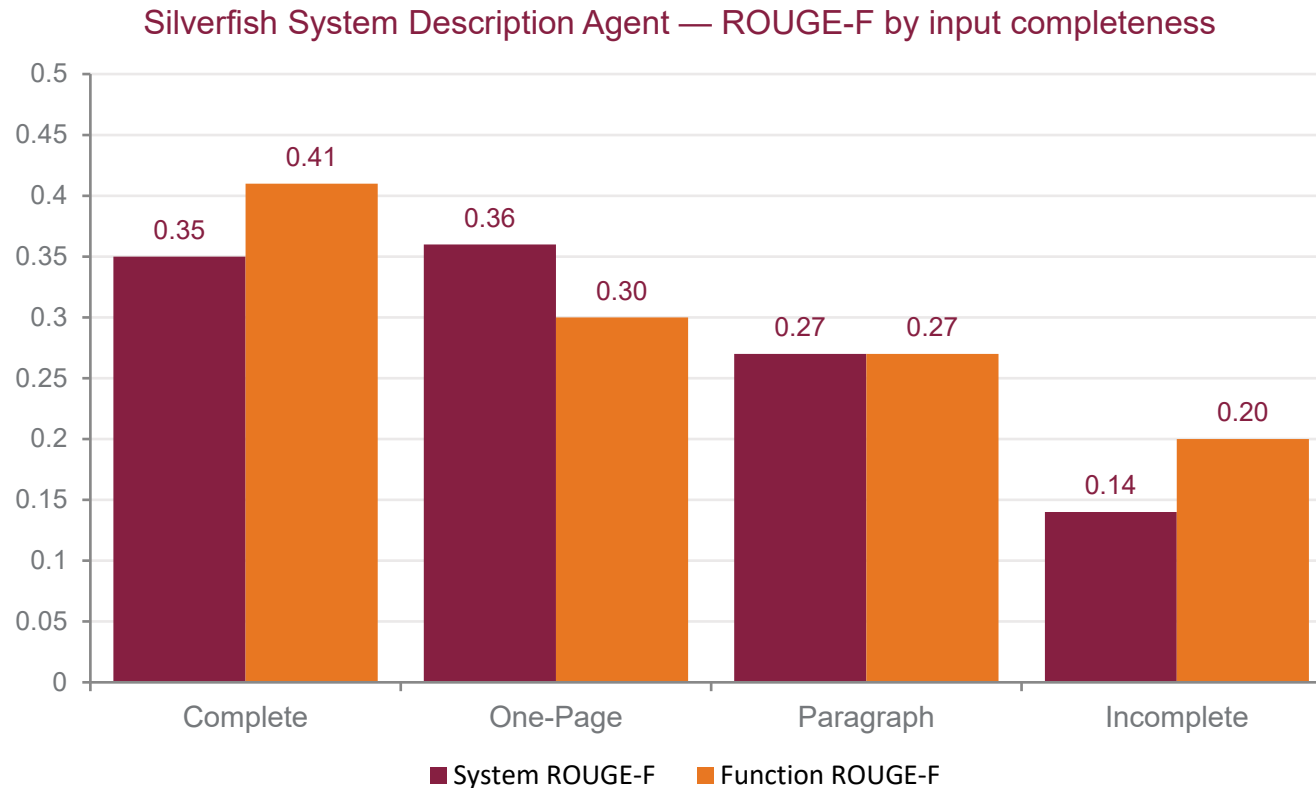


Each step closes a specific gap: cascading errors (isolate), source-quality sensitivity (degrade), generative randomness (repeat), and similarity mistaken for correctness (score three ways).

Scope note: benchmarks existed only through the STPA-Sec agents — Resilience Pattern and Cyber Survivability agents were not scored against a gold standard.

FINDING 1

Source quality isn't repaired downstream



**BERT held steady.
ROUGE dropped hard.**

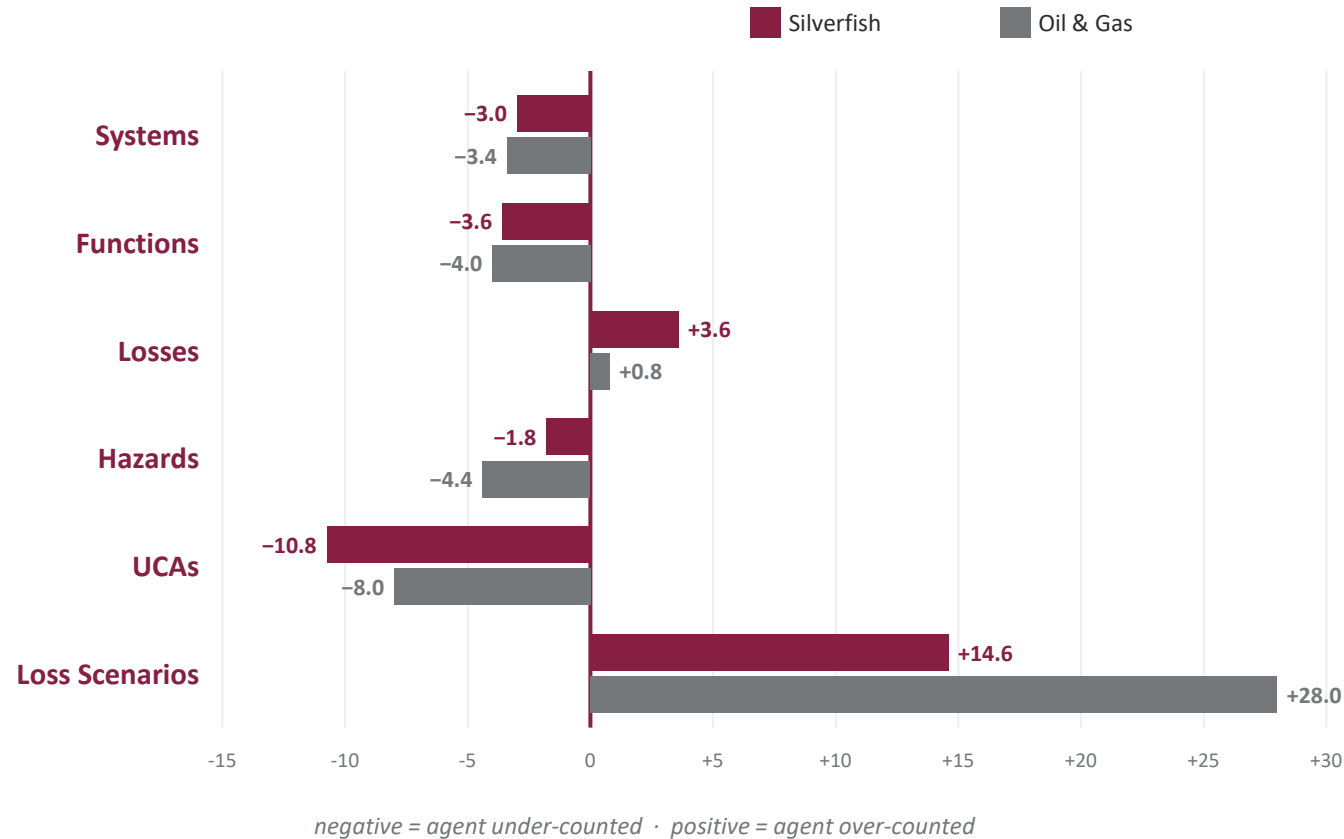
Semantic similarity (BERT) barely moved as inputs degraded — but exact-token overlap (ROUGE) fell sharply, especially for Silverfish. The agent still under-counted systems and functions by 3–5 items across every input level.

“Omitted or incorrect source information is not reliably repaired by the model, and can propagate through the workflow.”

FINDING 2

The direction of the error isn't even consistent

Difference from gold standard, by item type (complete input)



No consistent bias to correct for.

System Description and Hazards under-count. Losses and Loss Scenarios over-count. It isn't a simple "always conservative" or "always generous" model — each stage has to be checked on its own terms.

Loss Scenarios flipped sign entirely for Silverfish: +14.6 with the complete input, -3.4 with the incomplete input.

A high score doesn't mean it's methodologically correct

What BERT / ROUGE measure

- Semantic similarity between two strings
- Exact lexical / token overlap
- A 0.0–1.0 score against a reference

What STPA-Sec needs

Hierarchical authority, feedback loops, process-model relationships, and control-action structure — none of which a text-similarity score can confirm.

Where it broke down

Control Structure Agent

Routinely failed schema validation — performance could not be evaluated at all.

UCA Agent

1 of 5 Silverfish runs and 1–3 of 5 Oil & Gas runs failed schema validation, depending on input completeness.

Item counts

Underestimated hazards and UCAs in both use cases — UCAs by roughly 8–12 items on average.

Three fronts: evaluation, tooling, and MBSE integration

Evaluation

- Schema conformance & source fidelity
- Methodological correctness & traceability

More Tooling

- Complete the hierarchical control-model capability
- Strengthen the Modify Agent for dependency-aware review
- Extend the agent pipeline incrementally — specialized agents, not a single prompt

MBSE Model Manipulation

- Model queries for incomplete trace relationships
- Pattern-aware model transformations in SysML v2
- One machine-readable SCORE model shared by agents and tools

Thank you.

Questions welcome.

Mary Nerayo

Virginia Tech National Security Institute

mnerayo@vt.edu



Research Team

PI: Tom McDermott

Co-PI: Kelli Esser

Co-PI: Stephen Adams

Megan Clifford

Alex Kyer

Brian Walheim

Mary Nerayo

Julie Malherek

References

- Beling, P. A., Clifford, M. M., Sherburne, T., McDermott, T. A., & Horowitz, B. M. (2024). The "Mission Aware" concept for design of cyber-resilience. In D. Verma (Ed.), *Systems Engineering for the Digital Age: Practitioner Perspectives* (Ch. 24). John Wiley & Sons.
- Clifford, M. M., Sherburne, T., Horowitz, B. M., & Beling, P. A. (2024a). The Cyber Security Requirements Methodology and meta-model for design of cyber-resilience. In D. Verma (Ed.), *Systems Engineering for the Digital Age: Practitioner Perspectives* (Ch. 26). John Wiley & Sons.
- Clifford, M. M., Sherburne, T., Horowitz, B. M., McDermott, T. A., & Beling, P. A. (2024b). The "FOREST" concept and meta-model for lifecycle evaluation of resilience. In D. Verma (Ed.), *Systems Engineering for the Digital Age: Practitioner Perspectives* (Ch. 25). John Wiley & Sons.
- Esser, K., McDermott, T., & Adams, S. (2025). *Development of tools and methods for Secure Cyber Resilient Engineering (SCRE): Library of SCRE SysML v2 engineering design patterns* (Initial Technical Report SERC-2025-ITR-003, WRT-2515). Systems Engineering Research Center.
- Fleming, C. (2024). Introduction to STPA-Sec. In D. Verma (Ed.), *Systems Engineering for the Digital Age: Practitioner Perspectives* (Ch. 23). John Wiley & Sons.
- Gemma Team, Google DeepMind. (2026). *Gemma 4 technical report*. arXiv:2607.02770.
- LangChain Inc. (n.d.). *LangGraph* [Software documentation]. <https://docs.langchain.com/oss/python/langgraph/overview>
- Leveson, N. G., & Thomas, J. P. (2018). *STPA handbook*. MIT Partnership for Systems Approaches to Safety and Security.
- Lin, C. Y. (2004, July). ROUGE: A package for automatic evaluation of summaries. In *Text summarization branches out* (pp. 74–81).
- Madaan, A., Tandon, N., Gupta, P., Hallinan, S., Gao, L., Wiegrefe, S., Alon, U., Dziri, N., Prabhume, S., Yang, Y., Welleck, S., Majumder, B. P., Gupta, S., Yazdanbakhsh, A., & Clark, P. (2023). *Self-Refine: Iterative refinement with self-feedback*. Advances in Neural Information Processing Systems (NeurIPS).
- McDermott, T., Clifford, M. M., Sherburne, T., Horowitz, B., & Beling, P. A. (2022a). Framework for Operational Resilience in Engineering and System Test (FOREST) Part I: Methodology — responding to "security as a functional requirement." *INSIGHT*, 25(2), 30–37.
- McDermott, T., Clifford, M. M., Sherburne, T., Horowitz, B., & Beling, P. A. (2022b). Framework for Operational Resilience in Engineering and System Test (FOREST) Part II: Case study — responding to "security as a functional requirement." *INSIGHT*, 25(2), 38–43.
- McDermott, T., Esser, K., Adams, S., Clifford, M., Kyer, A., Walheim, B., Nerayo, M., & Malherek, J. (2026). *Development of tools and methods for Secure Cyber Resilient Engineering (SCRE)* (Final Technical Report, WRT-2515). Systems Engineering Research Center.
- Ollama. (n.d.). *Ollama* [Software]. <https://ollama.com>
- OpenAI. (2024). *Introducing structured outputs in the API*. OpenAI.
- Sahoo, P., Singh, A. K., Saha, S., Jain, V., Mondal, S., & Chadha, A. (2024). *A systematic survey of prompt engineering in large language models: Techniques and applications*. arXiv:2402.07927.
- Schulhoff, S., et al. (2024). *The prompt report: A systematic survey of prompting techniques*. arXiv:2406.06608.
- Wei, J., Wang, X., Schuurmans, D., Bosma, M., Xia, F., Chi, E., Le, Q. V., & Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. *Advances in Neural Information Processing Systems*.
- Young, W., & Leveson, N. G. (2014). An integrated approach to safety and security based on systems theory. *Communications of the ACM*, 57(2), 31–35.
- Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., & Artzi, Y. (2019). *BERTScore: Evaluating text generation with BERT*. arXiv:1904.09675.