

# Llama-3.1-FoundationAI-SecurityLLM-8B-Instruct Technical Report

Sajana Weerawardhena<sup>1</sup>, Paul Kassianik<sup>1</sup>, Blaine Nelson<sup>1</sup>, Baturay Saglam<sup>1,2,\*</sup>, Anu Vellore<sup>1</sup>, Aman Priyanshu<sup>1</sup>, Supriti Vijay<sup>1,3,\*</sup>, Massimo Aufiero<sup>1</sup>, Arthur Goldblatt<sup>1</sup>, Fraser Burch<sup>1</sup>, Ed Li<sup>1,2,\*</sup>, Jianliang He<sup>1,2,\*</sup>, Dhruv Kedia<sup>1</sup>, Kojin Oshiba<sup>1</sup>, Zhouan Yang<sup>1,2,\*</sup>, Yaron Singer<sup>1</sup>, Amin Karbasi<sup>1</sup>

<sup>1</sup>Foundation AI–Cisco Systems Inc.

<sup>2</sup>Yale University

<sup>3</sup>Carnegie Mellon University

\*Work done while at Foundation AI

## Abstract

Large language models (LLMs) have shown remarkable success across many domains, yet their integration into cybersecurity applications remains limited due to a lack of general-purpose cybersecurity data, representational complexity, and safety and regulatory concerns. To address this gap, we previously introduced Foundation-Sec-8B, a cybersecurity-focused LLM suitable for fine-tuning on downstream tasks. That model, however, was not designed for chat-style interactions or instruction-following. In this report, we release Foundation-Sec-8B-Instruct: a model specifically trained for general-purpose cybersecurity dialogue. Built on Foundation-Sec-8B, it combines domain-specific knowledge with instruction-following, conversational capabilities, and alignment with human preferences to produce high-quality, relevant responses. Comprehensive evaluations show that Foundation-Sec-8B-Instruct outperforms Llama 3.1-8B-Instruct on a range of cybersecurity tasks while matching its instruction-following performance. It is also competitive with GPT-4o-mini on cyber threat intelligence and instruction-following tasks. We envision Foundation-Sec-8B-Instruct becoming an indispensable assistant in the daily workflows of cybersecurity professionals. We release the model publicly at <https://huggingface.co/fdsn-ai/Foundation-Sec-8B-Instruct>.

## 1. Introduction

Frontier large language models (LLMs) are driving innovation across a swath of domains. Tools like ChatGPT [49] have seen widespread use, assisting users in tasks ranging from creative writing to technical problem-solving. Specialized models have also shown promise in fields such as medicine [9, 57], law [13, 14], and code generation [53]. Nevertheless, integration into mainstream cybersecurity workflows remains limited. Practitioners still face a range of practical challenges while commercial models often impose strict safety guardrails that constrain their practical utility for security applications [77].

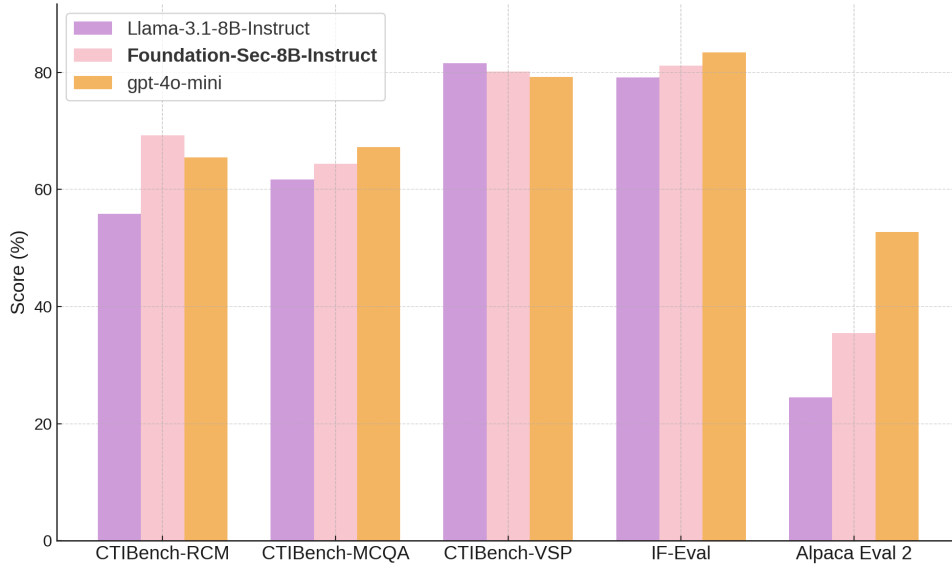


Figure 1 | Overview of core results on the selected benchmarks. Foundation-Sec-8B-Instruct matches or outperforms Llama 3.1-8B-Instruct and GPT-4o-mini in cyber threat intelligence tasks, and surpasses Llama 3.1-8B-Instruct on instruction-following tasks.

Lack of data, alignment gaps, and field breadth are key challenges to practical deployment of cybersecurity-specific LLMs. The scarcity of clean, high-quality, public datasets hinders the development of models with robust cybersecurity capabilities [43]. Interpretability research on LLM representations is still evolving, leaving models vulnerable to hallucinations [67] and distribution shifts [25]. The broad scope of cybersecurity topics and tasks—from phishing and malware detection to cryptographic operations—further complicates efforts to build a single, general-purpose model. As a result, the ecosystem remains fragmented and dominated by narrow, task-specific tools [45, 72, 78].

To address these challenges, we previously introduced Foundation-Sec-8B, a cybersecurity-specialized LLM based on Llama 3.1-8B [36]. Through continual pre-training on a large cybersecurity corpus, the model is infused with deep domain knowledge and serves as a foundation for general cybersecurity tasks. However, as a *base model*, it is not trained to interact with users, follow instructions in natural language, or align responses to human preferences.

In this work, we present Foundation-Sec-8B-Instruct, a cybersecurity chat model built on Foundation-Sec-8B with instruction-following and conversational capabilities. Base models alone typically require task-specific fine-tuning [28] or carefully designed in-context examples [6] before they can provide meaningful interactions. Foundation-Sec-8B-Instruct makes the knowledge embedded in Foundation-Sec-8B readily accessible—users can query the model directly and engage in interactive dialogue without any prerequisite setup. By combining domain expertise with instruction-tuning and preference alignment, Foundation-Sec-8B-Instruct enables a broad range of interactions and real-world workflows.

We provide comprehensive evaluations across diverse benchmarks to assess our model in both cybersecurity knowledge and instruction-following ability. Results show that Foundation-Sec-8B-Instruct outperforms Llama 3.1-8B-Instruct on cyber threat intelligence (CTI) benchmarks and remains competitive with GPT-4o-mini on both cybersecurity and general instruction-following tasks (see Figure 1). We also include a safety analysis of the model’s vulnerability to malicious prompts. Our goal is for Foundation-Sec-8B-Instruct to become a trusted daily assistant for cybersecurity professionals worldwide.

## 2. Related Work

We examine prior work on instruction-tuned, cybersecurity-specialized models and relevant post-training approaches. While much of the literature has focused on secure code generation or vulnerability detection [36], our goal is to align a general-purpose LLM with core cybersecurity tasks. We aim to enable broader applicability of the model across diverse use cases through instruction-following training. For a comprehensive overview of cybersecurity-focused language models, we refer readers to existing surveys [77].

### 2.1. Instruction-Tuned Cybersecurity LLMs

To our knowledge, only a few instruction-following models have been developed for general cybersecurity tasks. We include them as baselines in our evaluations.

**DeepHat-v1 [15]** A family of instruction-tuned cybersecurity models based on Llama 3.1 and Qwen-2.5. DeepHat is intentionally uncensored and optimized for offensive security applications, including red teaming and exploit development.

**Primus [75]** An instruction-tuned cybersecurity LLM built on Llama 3.1-8B, trained on a curated 2 billion-token corpus sourced from MITRE ATT&CK, Wikipedia, vendor reports, threat intelligence feeds, and manually collected CTI data.

**LilyCybersecurity [56]** A cybersecurity assistant based on Mistral-7B [32], fine-tuned with supervision on 22,000 hand-crafted conversations related to cybersecurity and hacking.

### 2.2. Post-Training

Modern post-training recipes improve model abilities in reasoning, math, coding, and tool use, while also adapting next-token prediction to better follow user instructions [50, 51]. Recent approaches have replaced complex multi-stage reinforcement learning (RL) algorithms, such as Proximal Policy Optimization (PPO) [55], with simpler methods like Direct Preference Optimization (DPO) [52].

As of this writing, and to our knowledge, there are no high-quality cybersecurity datasets or benchmarks for evaluating alignment with human preferences. In contrast, fields like medicine and law have emphasized the need to assess model outputs against domain-specific human judgments [24, 57].

Most existing cybersecurity benchmarks, however, focus on knowledge recall rather than alignment with human preferences. They tend to emphasize multiple-choice questions [39, 41, 66] or narrowly scoped tasks like classification (e.g., vulnerability detection), named entity recognition (NER), and summarization [16, 76]. Models evaluated only on such benchmarks leave open the question of whether they are suitable for broader, real-world use [75]. In contrast, our model not only performs well on cybersecurity knowledge recall tasks but also shows general alignment with human preferences, making it better suited for practical applications.

## 3. Benchmarks

We evaluate Foundation-Sec-8B-Instruct on both cybersecurity-specific and general post-training benchmarks. We evaluate the model on both cybersecurity-specific and general post-training benchmarks,

Benchmark	Domain
CTIBench-MCQA	cybersecurity
CTIBench-RCM	cybersecurity
CTIBench-VSP	cybersecurity
CyberMetric-500	cybersecurity
SecBench	cybersecurity
SecEval	cybersecurity
MMLU	knowledge
IFEval	instruction following
GSM8K	math
MATH	math
HumanEval	code
BigBenchHard	reasoning
AlpacaEval 2	human preference

Table 1 | Benchmarks used for our model evaluations and comparisons.

chosen to assess domain knowledge, instruction-following, practical utility, and reasoning skills. Table 1 summarizes all benchmarks used in our evaluation.

### 3.1. Security Benchmarks

Our principal assessment focuses on the model’s cybersecurity knowledge. We use a broad set of benchmarks that cover security governance and compliance, technical and infrastructure protection, threat detection and incident response, and emerging risks. Most benchmarks are formatted as multiple-choice question-answering (MCQA) tasks (i.e., CTIBench-MCQ, CyberMetric, SecBench, SecEval, MMLU-Computer Security), while others (i.e., CTIBench-RCM and CTIBench-VSP) use custom short-answer formats. A full list and detailed descriptions of all security benchmarks are provided in Appendix A.

These benchmarks also evaluate general instruction-following capabilities within a cybersecurity context. Each security benchmark can be framed as a cybersecurity instruction-following task with verifiable answers. We use performance on these tasks as a proxy for the model’s instruction-following ability in the security domain. Sample prompts and answers are provided in Appendix A.

### 3.2. Post-Training Evaluations

We evaluate our model on several general-purpose benchmarks, including Massive Multitask Language Understanding (MMLU) [26], IFEval [80], GSM8K [12], MATH [27], HumanEval [8], AlpacaEval 2 [18], and BigBenchHard [61]. These benchmarks assess instruction-following, coding, math, and reasoning capabilities. Detailed descriptions are provided in Appendix B. We use the Open Language Model Evaluation System (OLMES) [23] for all general benchmarks except MMLU, for which we use our own evaluation framework.

### 3.3. Cybersecurity Topics in Evaluations

To better understand the topic distribution of the cybersecurity benchmarks, we analyzed sub-topic coverage using our own taxonomy; see Appendix C for details.

The analysis in Figure 2 shows a skewed distribution: areas like application security are heavily

overrepresented, while other areas, such as security operations and cloud security, are underrepresented.

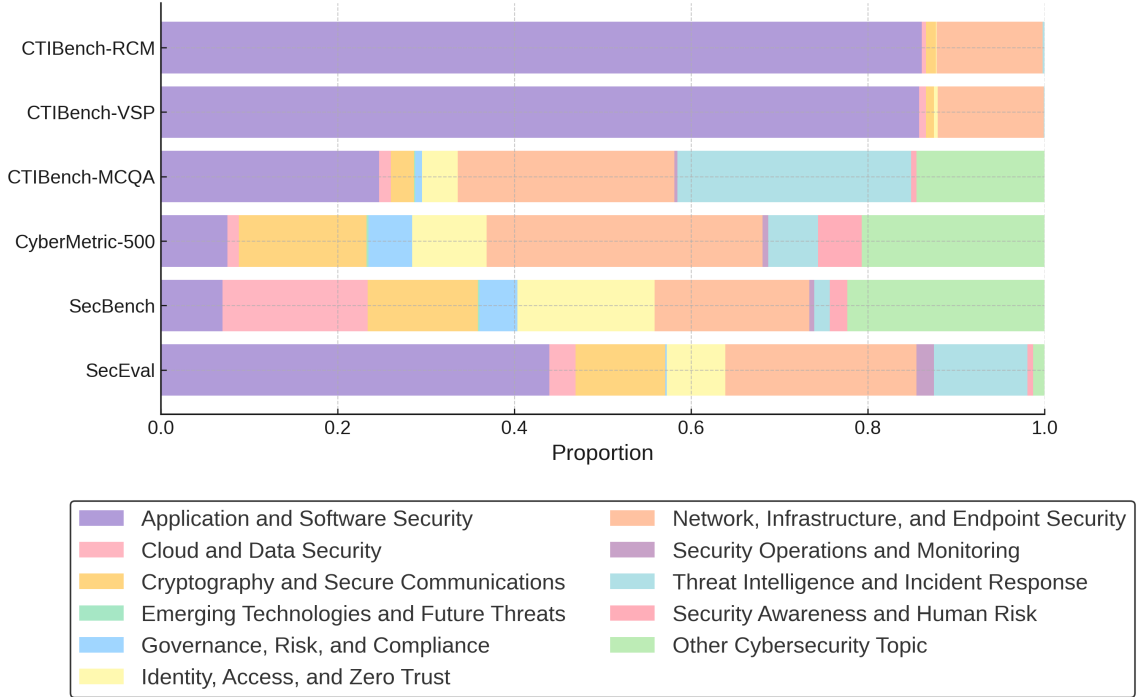


Figure 2 | Distribution of topics in common cybersecurity evaluation benchmarks. Application security is heavily overrepresented, particularly in CTIBench tasks, as well as network security and cloud security.

## 4. Post-Training Dataset Analysis

### 4.1. Datasets

In this section we study distribution of cybersecurity data in common open-sourced post-training datasets. Specifically we study: Tülu 3 [37], Tülu 2 [31], Alpaca [62], Open Platypus [38], WizardLM Evol Instruct (70k) [70], OpenOrca [40], Llama-Nemotron Post-Training Dataset (non-reasoning, chat, and safety components only) [5], Primus-Instruct and Primus-Reasoning [75].

### 4.2. Analysis Of Cybersecurity Data In Post-Training Datasets

As part of the data curation process, we studied the presence and makeup of cybersecurity data in the datasets listed in Section 4.1 (excluding Primus-Instruct and Primus-Reasoning as they are cybersecurity-oriented datasets).

We employ a two-stage classification system to identify and categorize cybersecurity-focused prompts according to our in-house cybersecurity taxonomy described in Appendix C. The first stage consists of keyword and regex-based pattern matching rules derived from this taxonomy. For additional details on the keyword-based classification stage, see Appendix D. The second stage applies a fine-tuned ModernBERT [69] model to perform multi-class classification in alignment with our taxonomy.

Table 2 summarizes our findings. Several datasets exhibit minimal security content with estimated ranges below 1%, but the WizardLM Evol Instruct, Nemotron SFT Chat, and Tülu datasets likely have non-trivial proportions of cybersecurity data. More comprehensive statistics are provided in Appendix D.

Source	Estimated Count	Estimated Percentage
OpenOrca	8555	0.20%
Tülu 3 SFT Mixture	13083	1.39%
Tülu 2 SFT Mixture	4309	1.32%
WizardLM Evol Instruct	1198	1.71%
Nemotron SFT Chat	711	1.79%
Nemotron SFT Safety	170	0.68%
Alpaca	168	0.32%
Open Platypus	100	0.32%

Table 2 | Estimated count and percentage of cybersecurity-focused prompts in open-sourced post-training datasets.



Figure 3 | Distribution of cybersecurity topics within post-training datasets.

Figure 3 shows the breakdown of each dataset by cybersecurity category as determined by our classification model. These post-training datasets demonstrate much greater coverage and balanced distributions of categories than the evaluation datasets in Figure 2.

This analysis may be of interest to the broader cybersecurity community, offering *insight into the composition of cybersecurity data in open-source post-training datasets*.

#### 4.3. Understanding Contamination on Post-Training Datasets

Large-scale post-training datasets are often aggregated from diverse and minimally filtered sources, making them prone to unintentional inclusion of benchmark content [17, 71]. Such contamination can artificially inflate reported performance, distort comparative evaluations, and undermine our ability to assess a model’s true generalization capabilities [20]. In this analysis, we aim to systematically characterize the extent and nature of contamination across commonly used post-training datasets. Understanding where and how benchmark leakage occurs enables more informed decisions about

Benchmark	Tülu 3 SFT				Primus-Reasoning				Alpaca			
	EM	NG	ES	LLM-ES	EM	NG	ES	LLM-ES	EM	NG	ES	LLM-ES
CTIBench-MCQA	0.00	0.00	4.12	0.1	33.40	32.04	13.48	10.2	0.00	0.00	3.44	0.1
CTIBench-RCM	0.00	0.00	1.70	0.00	83.10	78.80	91.50	74.9	0.00	0.00	0.00	0.00
CTIBench-VSP	0.00	0.00	3.70	0.00	83.10	78.20	78.40	72.6	0.00	0.00	0.10	0.00
CyberMetric-500	0.00	0.00	28.00	0.6	0.00	0.00	0.00	0.00	0.00	0.00	33.20	2.2

Table 3 | Contamination levels across selected post-training datasets relative to benchmark sizes (percentages, rounded). EM = Exact Match, NG = N-Gram Overlap, ES = Embedding Similarity, and LLM-ES = LLM-verified Embedding Similarity. Light red cells indicate values  $> 0\%$  but  $\leq 2\%$ ; red cells indicate values  $> 2\%$  but  $\leq 50\%$ ; dark red cells indicate values  $> 50\%$ .

dataset selection for model training and guides the design of decontamination pipelines that preserve benchmark integrity.

Building on prior work [17, 37, 71, 74, 81], we implement a layered framework comprising three components: an n-gram-based search to detect verbatim matches, an embedding-based cosine similarity filter to identify semantically similar phrases and an LLM-as-a-Judge quality filter to improve the detection of paraphrases. These components provide a robust mechanism for detecting contamination, though results remain sensitive to hyperparameter choices.

We apply this framework to several widely used post-training datasets and summarize their overlap with security-focused benchmarks in Table 3. As was reported in its dataset report [75], the Primus-Reasoning dataset was constructed using cybersecurity reasoning tasks from CTIBench. Our contamination detection correctly identifies this contamination with all three components, particularly in CTIBench-RCM (79-91% detected) and CTIBench-VSP (72-83% detected). This validates that our approach correctly detects known contamination and highlights the risk posed by training on datasets without conducting contamination audits, as we evaluate on both of these benchmarks. The contamination of CTIBench datasets is low ( $<5\%$ ) in other datasets, while modest levels of contamination from CyberMetric-500 are found in Tülu 3 and Alpaca.

Notably embedding similarity dominates contamination detection in these datasets. However, embedding similarity between corpora of security data tends to be high by default, which can inflate contamination estimates. We attribute this to several factors: embedding similarity reflects semantically learned representations, and the relatively limited presence of cybersecurity data in the pre-training of these embedding models may reduce their ability to capture fine-grained distinctions in this domain [4, 46]; moreover, similar to known issues with embedding metrics like BERTScore, these embeddings may conflate shared domain-specific vocabulary with true content overlap [47, 60]. To reduce false positives and improve robustness, we tuned to a higher similarity threshold (e.g., cosine 0.75–0.8) and validated a subset of matches using an LLM-as-a-judge approach. A detailed breakdown of our choices and representative examples at different similarity levels can be found in Appendix E.2.



## 5. Training

### 5.1. Cybersecurity Data Mix

Prior work has suggested that adding new knowledge during fine-tuning increases the likelihood of hallucinations [21, 44]. We adopt the stance that post-training should not attempt to impart novel cybersecurity knowledge. Instead, we rely primarily on pre-training of our base model to provide this knowledge. We further observed that including large amounts of cybersecurity data in our post-training datasets tends to dilute the effectiveness of our knowledge-based cybersecurity evaluations, making it difficult to assess generalization and compare between model candidates.

### 5.2. Supervised Fine-tuning and Direct Preference Optimization

Supervised fine-tuning is utilized to hone core instruction-following skills in the model. In contrast, RL techniques are primarily used to improve on instruction-following and alignment with human preferences. Starting from Foundation-Sec-8B, we apply a combination of SFT and RL techniques to develop our final instruction-tuned model.

Similar to other post-training procedures, we also rely on high quality synthetic data [2, 22, 73]. We built a synthetic data generation pipeline with a focus on refinement through rejection sampling, difficulty grading, and automated verification checks. Our final model also went through several rounds of human preference testing that revealed insights which were then used to bolster our preference tuning data.

Consistent with prior work, we note that data diversity during post-training drives model generalization [11, 68, 79]. In a similar vein, we observe that data diversity is also a key driver of knowledge retention. Prior work showed that while minor degradation is expected during post-training, the severity can be mitigated by increasing the diversity and balance of the post-training dataset [22, 48]. As described in Section 6.5, Foundation-Sec-8B-Instruct only shows mild degradation in knowledge tasks compared to Foundation-Sec-8B.

## 6. Results

We evaluate both the baseline models and Foundation-Sec-8B-Instruct against security benchmarks described in Section 3.1 and against general evaluations described in Section 3.2. On security benchmarks, we ran 10 trials at temperature 0.3 and we report the mean accuracy of the model on each benchmark with one standard deviation in Table 4. On general post-training benchmarks, we stick to standard evaluation settings accepted by the community and implemented by OLMES [23]. We present these results in Table 5. More details are presented in Appendix B.

### 6.1. Baselines

We include the following cybersecurity LLMs as baselines for comparison: DeepHat-v1-7B (formerly WhiteRabbitNeo) [15], Primus-base, and Primus-merged [75]. We also include comparisons to models from the Llama 3.1 [22], Gemma-3 [63] and Qwen-2.5 families [73] as well as GPT-4o-mini<sup>1</sup> [29].

---

<sup>1</sup>We use the 07.18.2024 checkpoint for GPT-4o-mini.



## 6.2. Security Benchmark Performance

Foundation-Sec-8B-Instruct achieves **state-of-the-art performance** on CTIBench-RCM beating larger models including GPT-4o-mini and Llama 3.1-70B-Instruct. When considering performance error bars (within 1 standard deviation), the model is consistently among the top performers on CTIBench-MCQA. This showcases Foundation-Sec-8B-Instruct’s strong performance on CTI tasks, mirroring the performance of Foundation-Sec-8B. The model is competitive on CyberMetric-500, SecBench and SecEval.

## 6.3. General Post-Training Performance

Foundation-Sec-8B-Instruct is **best-in-class in instruction following ability and human preferred output** among cybersecurity LLMs. On AlpacaEval 2, the next-best cybersecurity model has less than half the win-rate while in IFEval the closest model is 8 points behind. The model also outperforms Llama 3.1-8B-Instruct by 11 points and 2 points respectively.

In grade school math, mathematical reasoning, and coding, Foundation-Sec-8B-Instruct rivals peer models and is within 3 percentage points of Llama 3.1-8B-Instruct, demonstrating comparable performance. Finally, the performance of Foundation-Sec-8B-Instruct with respect to Llama 3.1-8B-Instruct on MMLU hints at a narrow base of knowledge, while BigBenchHard suggests potential for further development in reasoning capabilities.

Overall Foundation-Sec-8B-Instruct’s strong instruction following ability sets it up for practical deployment in cybersecurity contexts.

Model	CTIBench-RCM	CTIBench-MCQA	CTIBench-VSP	CyberMetric-500	SecBench	SecEval
Gemma-34B-Instruct	0.382±0.006	0.578±0.001	0.775±0.001	0.768±0.002	0.615±0.002	0.813±0.001
Qwen-2.5-7B-Instruct	0.572±0.008	0.644±0.001	0.804±0.001	0.859±0.002	0.767±0.003	0.870±0.001
DeepHat-v1-7B	0.664±0.007	0.645±0.006	0.808±0.003	0.869±0.007	0.716±0.004	0.880±0.003
Llama 3.1-8B-Instruct	0.558±0.007	0.617±0.004	0.815±0.002	0.847±0.005	0.723±0.010	0.855±0.003
Primus-base	0.639±0.006	0.652±0.003	0.754±0.008	0.854±0.008	0.713±0.005	0.841±0.006
Primus-merged	0.666±0.007	0.643±0.004	0.790±0.003	0.856±0.004	0.721±0.009	0.849±0.002
Gemma-312B-Instruct	0.361±0.007	0.629±0.001	<b>0.867±0.001</b>	0.865±0.001	0.749±0.001	0.868±0.001
Gemma-327B-Instruct	0.549±0.006	0.653±0.001	0.821±0.000	0.870±0.002	0.780±0.002	0.896±0.000
Llama 3.1-70B-Instruct	0.623±0.005	<b>0.695±0.002</b>	0.840±0.001	<b>0.930±0.005</b>	<b>0.839±0.003</b>	<b>0.900±0.002</b>
GPT-4o-mini	0.655±0.005	0.672±0.003	0.792±0.004	0.889±0.003	0.803±0.003	0.889±0.001
<b>Foundation-Sec-8B-Instruct</b>	<b>0.692±0.005</b> (↑24.03%)	0.644±0.003 (↑4.40%)	0.802±0.004 (↓-1.67%)	0.830±0.005 (↓-2.01%)	0.685±0.006 (↓-5.21%)	0.833±0.003 (↓-2.50%)

Table 4 | Performance on the selected cybersecurity benchmarks (temperature 0.3). Reported performance differences are relative to Llama 3.1-8B-Instruct. For CTIBench-RCM, CTIBench-MCQA, CyberMetric-500, SecBench, and SecEval we report the average accuracy over 10 trials ± one standard deviation. For CTIBench-VSP we report the average CVSS score (see Appendix A.1) over 10 trials ± one standard deviation.

Model	MMLU	BBH	GSM8k	Alpaca Eval 2	MATH	IFEval	HumanEval
Gemma-34B-Instruct	0.582	0.709	0.780	42.465	0.604	0.773	0.757
Qwen-2.5-7B-Instruct	0.717	0.709	0.842	28.983	0.699	0.758	0.927
DeepHat-v1-7B	0.664	0.689	0.828	13.738	0.462	0.721	0.950
Llama 3.1-8B-Instruct	0.679	0.725	0.835	24.477	0.425	0.791	0.864
Primus-base	0.650	0.698	0.783	11.454	0.327	0.686	0.856
Primus-merged	0.660	0.718	0.805	16.082	0.373	0.738	0.874
Gemma-312B-Instruct	0.723	0.820	0.870	52.340	0.732	0.852	0.869
Gemma-327B-Instruct	0.767	<b>0.835</b>	0.907	<b>64.044</b>	<b>0.758</b>	0.834	0.893
Llama 3.1-70B-Instruct	<b>0.810</b>	0.831	<b>0.942</b>	33.140	0.560	<b>0.878</b>	<b>0.951</b>
GPT-4o-mini	0.804	0.777	0.832	52.720	0.679	0.834	0.929
<b>Foundation-Sec-8B-Instruct</b>	0.602 (↓-11.31%)	0.677 (↓-6.57%)	0.817 (↓-2.09%)	35.453 (↑44.84%)	0.411 (↓-3.36%)	0.811 (↑2.57%)	0.843 (↓-2.44%)

Table 5 | Performance on the selected post-training benchmarks. Reported performance differences are relative to Llama 3.1-8B-Instruct. Evaluation parameters and metrics described in Appendix B.

#### 6.4. Persona Adaptation

Cybersecurity workflows involve distinct practitioner roles such as SOC analysts, red teamers, and threat intelligence specialists, each requiring domain-specific language and task framing. To assess the model’s ability to emulate such contexts, we evaluate its persona adaptation capabilities through PersonaGym [54]—a standardized benchmark for assessing LLMs on multi-turn, persona-driven conversations. We evaluate our model on a subset of the PersonaGym test split (50 of the original 200 personas). As shown in Table 6, Foundation-Sec-8B-Instruct demonstrates strong alignment with assigned personas without requiring dedicated personalization fine-tuning. Additional evaluation details are provided in Appendix F.

Model	Action Justification	Expected Actions	Linguistic Habits	Persona Consistency	Toxicity Control	Persona Score
Llama 2-8B	3.96	3.87	3.77	4.12	4.18	3.98
GPT 3.5	4.31	4.28	3.63	4.70	4.96	4.38
Llama 2-70B	4.44	4.32	3.85	4.67	4.68	4.39
Llama 3-8B	4.55	4.43	3.97	4.77	4.74	4.49
Claude 3 Haiku	2.47	4.28	3.04	3.47	4.94	3.64
Claude 3.5 Sonnet	4.52	4.37	3.98	4.81	4.88	4.51
Foundation-Sec-8B-Instruct	4.4	4.6	3.95	4.95	5.0	4.58

Table 6 | PersonaGym Results. Scores range from 1 (lowest) to 5 (highest). Reported results for all models except ours are taken from [54].

## 6.5. Comparison to Foundation-Sec-8B

To showcase the retention of cybersecurity knowledge during post-training, we present a head-to-head comparison between Foundation-Sec-8B-Instruct and Foundation-Sec-8B on cybersecurity evaluations. Due to the inability of Foundation-Sec-8B to follow precise instructions, we use 5 shot prompting. We compare the models on CyberMetric-500, CTIBench-MCQA, and SecBench.

As noted in [36], instruction-tuning affects next-token prediction capabilities of models for few-shot settings, often resulting in verbose responses that do not follow the terse templates given by the few-shot examples. This complicates the direct comparison between base and instruction-tuned models using few-shot evaluations. This is an issue particularly for CTIBench-RCM due to its complexity and short-answer format. Therefore we exclude it from this analysis.

Figure 4 shows our results. We acknowledge that our measure of latent knowledge retention is limited by the diversity in these evaluations and obscured by each model’s ability to follow instructions.

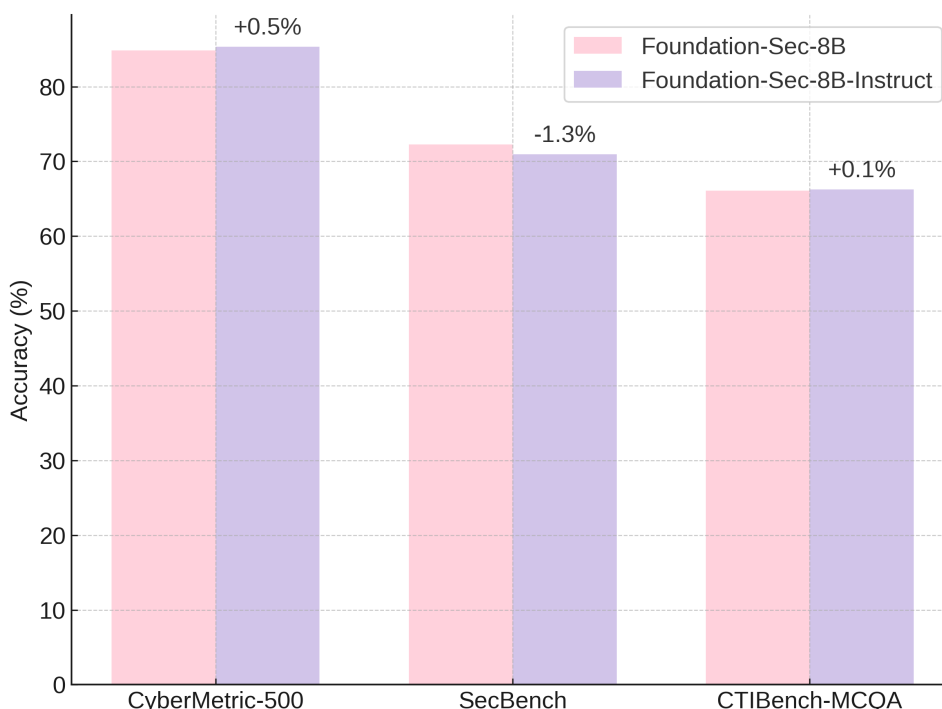


Figure 4 | Comparison of Foundation-Sec-8B-Instruct to Foundation-Sec-8B on selected cybersecurity metrics. The negligible differences in performance suggest minimal forgetting of cybersecurity knowledge in the instruction-tuned model. Both models were 5-shot prompted and results averaged over 10 trials.

## 7. Safety Alignment

Foundation-Sec-8B-Instruct has not undergone dedicated safety alignment procedures beyond basic instruction-tuning. However, we followed standard practices to provide a baseline level of alignment. Because this is an instruction-tuned model as opposed to a base model, safety training is particularly important, as the model is designed to follow user prompts and may do so in unsafe ways. The model has not been explicitly trained for robustness against adversarial techniques, such as jailbreaking or prompt injection, and no post-hoc safety filtering (e.g., moderation classifiers or rule-based sanitization) has been applied to its outputs. As a result, the model may produce unfiltered or unsafe content, including

toxic, biased, or factually incorrect responses. **We strongly recommend applying additional safety layers, such as automated content filtering or LLM-based moderation systems, when deploying or experimenting with this model.**

To better understand the model’s risk profile and limitations despite the lack of explicit safety alignment, we conducted an evaluation of its safety and robustness using HarmBench [42], a benchmark framework designed for automated red teaming of LLMs. The model was evaluated on 400 representative HarmBench prompts for simple toxicity and demonstrated satisfactory performance, rejecting or safely responding to 92% of malicious examples across a broad range of risk categories.

To further mitigate safety risks, we recommend pairing the model with LlamaGuard [30], which implements taxonomy-driven input-output filtering. When evaluated together, Foundation-Sec-8B-Instruct and LlamaGuard rejected 99% of malicious test cases.

In order to provide users with a good experience when interacting with the model, we developed a detailed system prompt for general user interaction. We tested this system prompt in internal testing and found that it improved user satisfaction and safety. This system prompt is baked into the model’s chat template but can be modified or overridden as needed. The system prompt is reproduced in Appendix G and can be tailored to particular use cases.

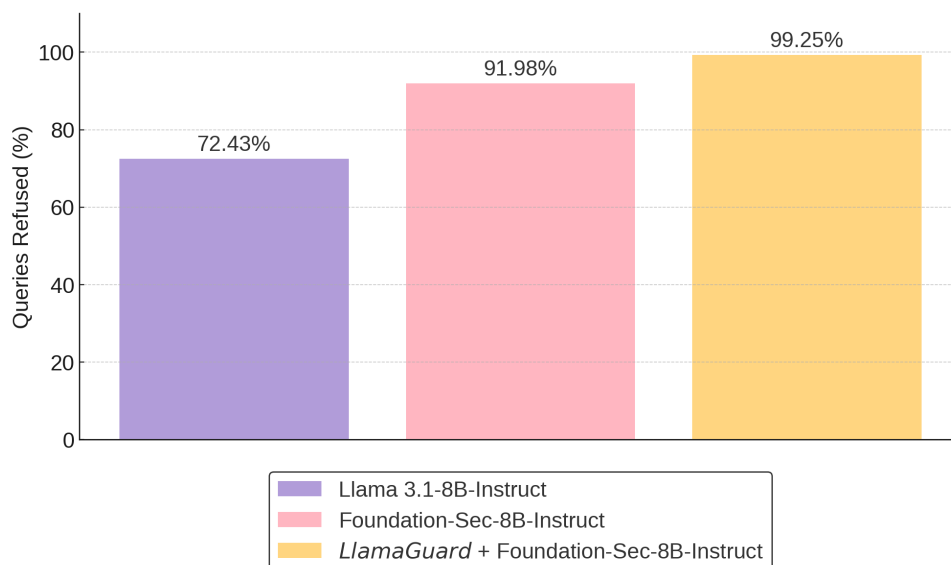


Figure 5 | Comparison of Llama 3.1-8B-Instruct, Foundation-Sec-8B-Instruct, and Foundation-Sec-8B-Instruct protected by LlamaGuard [30] on malicious queries from HarmBench [42]. While Foundation-Sec-8B-Instruct performs significantly better than Llama 3.1-8B-Instruct, applying LlamaGuard increases the refusal rate on malicious requests to nearly 100%. Therefore we recommend deploying the model with additional guardrails in production use cases.

## 8. Conclusion

We introduce Foundation-Sec-8B-Instruct, a best-in-class large language model (LLM) for general-purpose cybersecurity dialogue, designed to support practitioners, students, and others across a wide range of security tasks. Foundation-Sec-8B-Instruct outperforms Llama 3.1-8B-Instruct and GPT-4o-mini on CTIBench-RCM and ranks among the top performers in several other cybersecurity benchmarks for its size. Beyond domain-specific tasks, it also delivers competitive results on general instruction-following and human preference alignment benchmarks—showcasing its value as a practical model.

Our results demonstrate that domain-adapted models can retain strong general-purpose capabilities while excelling at specialized tasks. With the release of this model, we aim to meet the growing need for cybersecurity-focused LLMs with robust zero-shot performance across diverse contexts. We believe Foundation-Sec-8B-Instruct will help advance LLM integration in cybersecurity and be a daily driver for many cybersecurity practitioners.

## Acknowledgements

We thank Alie Fordyce for her leadership in developing a safe model release strategy and extensive documentation. We also thank the Foundation AI team, including Ron Kupfer, Howard Lin, Kimia Majd, Mayank Rajoria, Rahim Dharssi, Alexander Chen, Avi Zohary, Nathan Chang, Roe Landerman, Takahiro Matsumoto, Yasukazu Hirata, Amos Yoffe, Assaf Eisenman, Hyrum Anderson, and Konstantin Goldin for their invaluable support.

## References

- [1] Common vulnerability scoring system v3.1: Specification document. <https://www.first.org/cvss/v3.1/specification-document>.
- [2] Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J Hewett, Mojan Javaheripi, Piero Kauffmann, et al. Phi-4 technical report. [arXiv preprint arXiv:2412.08905](https://arxiv.org/abs/2412.08905), 2024.
- [3] Md Tanvirul Alam, Dipkamal Bhusal, Le Nguyen, and Nidhi Rastogi. CTIBench: A benchmark for evaluating LLMs in cyber threat intelligence. In [The Thirty-eight Conference on Neural Information Processing Systems Datasets and Benchmarks Track](https://openreview.net/forum?id=iJA0psXo2I), 2024. URL <https://openreview.net/forum?id=iJA0psXo2I>.
- [4] Markus Bayer, Philipp Kuehn, Ramin Shanehsaz, and Christian Reuter. Cysecbert: A domain-adapted language model for the cybersecurity domain, 2022. URL <https://arxiv.org/abs/2212.02974>.
- [5] Akhiad Bercovich, Itay Levy, Izik Golan, Mohammad Dabbah, Ran El-Yaniv, Omri Puny, Ido Galil, Zach Moshe, Tomer Ronen, Najeeb Nabwani, et al. Llama-nemotron: Efficient reasoning models. [arXiv preprint arXiv:2505.00949](https://arxiv.org/abs/2505.00949), 2025.
- [6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. [Advances in neural information processing systems](https://arxiv.org/abs/2005.14165), 33:1877–1901, 2020.
- [7] CAPEC. Common attack pattern enumerations and classifications (capec). <https://capec.mitre.org/>, 2024. Available at <https://capec.mitre.org/>.
- [8] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. [arXiv preprint arXiv:2107.03374](https://arxiv.org/abs/2107.03374), 2021.
- [9] Zeming Chen, Alejandro Hernández Cano, Angelika Romanou, Antoine Bonnet, Kyle Matoba, Francesco Salvi, Matteo Pagliardini, Simin Fan, Andreas Köpf, Amirkeivan Mohtashami, et al. Meditron-70b: Scaling medical pretraining for large language models. [arXiv preprint arXiv:2311.16079](https://arxiv.org/abs/2311.16079), 2023.

- [10] Steve Christey, J. Kenderdine, J. Mazella, and B. Miles. Common weakness enumeration. Technical report, The MITRE Corporation, 2013. URL <https://cwe.mitre.org/>.
- [11] Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53, 2024.
- [12] Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.
- [13] Pierre Colombo, Telmo Pires, Malik Boudiaf, Rui Melo, Dominic Culver, Etienne Malaboeuf, Gabriel Hauteux, Johanne Charpentier, and Michael Desa. Saullm-54b & saullm-141b: Scaling up domain adaptation for the legal domain. *Advances in Neural Information Processing Systems*, 37:129672–129695, 2024.
- [14] Pierre Colombo, Telmo Pessoa Pires, Malik Boudiaf, Dominic Culver, Rui Melo, Caio Corro, Andre FT Martins, Fabrizio Esposito, Vera Lúcia Raposo, Sofia Morgado, et al. Saullm-7b: A pioneering large language model for law. *arXiv preprint arXiv:2403.03883*, 2024.
- [15] DeepHat. AI for DevSecOps, DeepHat. <https://www.deephat.ai/>, 2025. Accessed: 2025-7-29.
- [16] Pritam Deka, Sampath Rajapaksha, Ruby Rani, Amirah Almutairi, and Erisa Karafili. Attacker: towards enhancing cyber-attack attribution with a named entity recognition dataset. In *International Conference on Web Information Systems Engineering*, pages 255–270. Springer, 2024.
- [17] Chunyuan Deng, Yilun Zhao, Xiangru Tang, Mark Gerstein, and Arman Cohan. Investigating data contamination in modern benchmarks for large language models. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8706–8719, Mexico City, Mexico, June 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.naacl-long.482. URL <https://aclanthology.org/2024.naacl-long.482/>.
- [18] Yann Dubois, Balázs Galambosi, Percy Liang, and Tatsunori B Hashimoto. Length-controlled alpacaEval: A simple way to debias automatic evaluators. *arXiv preprint arXiv:2404.04475*, 2024.
- [19] European Union. General Data Protection Regulation (GDPR). <https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679>, 2024. Available at <https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679>.
- [20] Yujuan Fu, Ozlem Uzuner, Meliha Yetisgen, and Fei Xia. Does data contamination detection work (well) for LLMs? a survey and evaluation on detection assumptions. In Luis Chiruzzo, Alan Ritter, and Lu Wang, editors, *Findings of the Association for Computational Linguistics: NAACL 2025*, pages 5235–5256, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-195-7. doi: 10.18653/v1/2025.findings-naacl.291. URL <https://aclanthology.org/2025.findings-naacl.291/>.
- [21] Zorik Gekhman, Gal Yona, Roei Aharoni, Matan Eyal, Amir Feder, Roi Reichart, and Jonathan Herzig. Does fine-tuning llms on new knowledge encourage hallucinations? *arXiv preprint arXiv:2405.05904*, 2024.
- [22] Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024.

- [23] Yuling Gu, Oyvind Tafjord, Bailey Kuehl, Dany Haddad, Jesse Dodge, and Hannaneh Hajishirzi. Olmes: A standard for language model evaluations. arXiv preprint arXiv:2406.08446, 2024.
- [24] Neel Guha, Julian Nyarko, Daniel Ho, Christopher Ré, Adam Chilton, Alex Chohlas-Wood, Austin Peters, Brandon Waldon, Daniel Rockmore, Diego Zambrano, et al. Legalbench: A collaboratively built benchmark for measuring legal reasoning in large language models. Advances in Neural Information Processing Systems, 36:44123–44279, 2023.
- [25] Ragini Gupta, Shinan Liu, Ruixiao Zhang, Xinyue Hu, Pranav Kommaraju, Xiaoyang Wang, Hadjer Benkraouda, Nick Feamster, and Klara Nahrstedt. Generative active adaptation for drifting and imbalanced network intrusion detection. arXiv [cs.NI], March 2025.
- [26] Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In International Conference on Learning Representations, 2021. URL <https://openreview.net/forum?id=d7KBjmI3GmQ>.
- [27] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the math dataset. arXiv preprint arXiv:2103.03874, 2021.
- [28] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification. arXiv preprint arXiv:1801.06146, 2018.
- [29] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. arXiv preprint arXiv:2410.21276, 2024.
- [30] Hakan Inan, Kartikeya Upasani, Jianfeng Chi, Rashi Rungta, Krithika Iyer, Yuning Mao, Michael Tontchev, Qing Hu, Brian Fuller, Davide Testuggine, et al. Llama guard: Llm-based input-output safeguard for human-ai conversations. arXiv preprint arXiv:2312.06674, 2023.
- [31] Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A. Smith, Iz Beltagy, and Hannaneh Hajishirzi. Camels in a changing climate: Enhancing lm adaptation with tulu 2. 2023.
- [32] Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, L  lio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timoth  e Lacroix, and William El Sayed. Mistral 7b, 2023. URL <https://arxiv.org/abs/2310.06825>.
- [33] Pengfei Jing, Mengyun Tang, Xiaorong Shi, Xing Zheng, Sen Nie, Shi Wu, Yong Yang, and Xiapu Luo. Secbench: A comprehensive multi-dimensional benchmarking dataset for llms in cybersecurity. arXiv preprint arXiv:2412.20787, 2024.
- [34] Chris Johnson, Lee Badger, David Waltermire, Julie Snyder, and Clem Skorupka. Guide to cyber threat information sharing. Technical Report 800-150, National Institute of Standards and Technology (NIST), 2016. URL <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-150.pdf>.
- [35] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. arXiv preprint arXiv:1607.01759, 2016.
- [36] Paul Kassianik, Baturay Saglam, Alexander Chen, Blaine Nelson, Anu Vellore, Massimo Aufiero, Fraser Burch, Dhruv Kedia, Avi Zohary, Sajana Weerawardhena, et al. Llama-3.1-foundationai-securityllm-base-8b technical report. arXiv preprint arXiv:2504.21039, 2025.



- [37] Nathan Lambert, Jacob Morrison, Valentina Pyatkin, Shengyi Huang, Hamish Ivison, Faeze Brahman, Lester James V Miranda, Alisa Liu, Nouha Dziri, Shane Lyu, et al. Tulu 3: Pushing frontiers in open language model post-training. arXiv preprint arXiv:2411.15124, 2024.
- [38] Ariel N Lee, Cole J Hunter, and Nataniel Ruiz. Platypus: Quick, cheap, and powerful refinement of llms. arXiv preprint arXiv:2308.07317, 2023.
- [39] Guancheng Li, Yifeng Li, Wang Guannan, Haoyu Yang, and Yang Yu. Seceval: A comprehensive benchmark for evaluating cybersecurity knowledge of foundation models. <https://github.com/XuanwuAI/SecEval>, 2023.
- [40] Wing Lian, Bleys Goodson, Eugene Pentland, Austin Cook, Chanvichet Vong, and "Teknium". Openorca: An open dataset of gpt augmented flan reasoning traces. <https://huggingface.co/datasets/Open-Orca/OpenOrca>, 2023.
- [41] Zefang Liu. Secqa: A concise question-answering dataset for evaluating large language models in computer security. arXiv preprint arXiv:2312.15838, 2023.
- [42] Mantas Mazeika, Long Phan, Xuwang Yin, Andy Zou, Zifan Wang, Norman Mu, Elham Sakhaee, Nathaniel Li, Steven Basart, Bo Li, et al. Harmbench: A standardized evaluation framework for automated red teaming and robust refusal. arXiv preprint arXiv:2402.04249, 2024.
- [43] Innocent Mbona and Jan H. P. Eloff. Data sets for cyber security machine learning models: A methodological approach. In Proceedings of the 9th International Conference on Internet of Things, Big Data and Security (IoTBDs), pages 149–156. SCITEPRESS, 2024.
- [44] Sabrina J Mielke, Arthur Szlam, Emily Dinan, and Y-Lan Boureau. Reducing conversational agents' overconfidence through linguistic calibration. Transactions of the Association for Computational Linguistics, 10:857–872, 2022.
- [45] Farzad Nourmohammadzadeh Motlagh, Mehrdad Hajizadeh, Mehryar Majd, Pejman Najafi, Feng Cheng, and Christoph Meinel. Large language models in cybersecurity: State-of-the-art. arXiv preprint arXiv:2402.00891, 2024.
- [46] Sara Mumtaz, Carlos Rodriguez, Boualem Benatallah, Mortada Al-Banna, and Shayan Zamanirad. Learning word representation for the cyber security vulnerability domain. In 2020 International Joint Conference on Neural Networks (IJCNN), pages 1–8, 2020. doi: 10.1109/IJCNN48605.2020.9207140.
- [47] Vincent Nguyen, Sarvnaz Karimi, Maciej Rybinski, and Zhenchang Xing. Cross-domain language modeling: An empirical investigation. In Afshin Rahimi, William Lane, and Guido Zuccon, editors, Proceedings of the 19th Annual Workshop of the Australasian Language Technology Association, pages 192–200, Online, December 2021. Australasian Language Technology Association. URL <https://aclanthology.org/2021.alta-1.22/>.
- [48] Team OLMo, Pete Walsh, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Shane Arora, Akshita Bhagia, Yuling Gu, Shengyi Huang, Matt Jordan, et al. 2 olmo 2 furious. arXiv preprint arXiv:2501.00656, 2024.
- [49] OpenAI. ChatGPT [large language model]. <https://chat.openai.com/chat>, 2023. Accessed: 2025-04-23.
- [50] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton,

- Luke Miller, Maddie Simens, Amanda Askill, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL <https://arxiv.org/abs/2203.02155>.
- [51] Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744, 2022.
  - [52] Rafael Rafailov, Archit Sharma, Eric Mitchell, Christopher D Manning, Stefano Ermon, and Chelsea Finn. Direct preference optimization: Your language model is secretly a reward model. *Advances in Neural Information Processing Systems*, 36:53728–53741, 2023.
  - [53] Baptiste Roziere, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, et al. Code llama: Open foundation models for code. *arXiv preprint arXiv:2308.12950*, 2023.
  - [54] Vinay Samuel, Henry Peng Zou, Yue Zhou, Shreyas Chaudhari, Ashwin Kalyan, Tanmay Rajpurohit, Ameet Deshpande, Karthik Narasimhan, and Vishvak Murahari. Personagym: Evaluating persona agents and llms. *arXiv preprint arXiv:2407.18416*, 2024.
  - [55] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL <https://arxiv.org/abs/1707.06347>.
  - [56] segolilylabs. Lily-cybersecurity-7b-v0.2: A cybersecurity assistant. Hugging Face model repository, 2025. URL <https://huggingface.co/segolilylabs/Lily-Cybersecurity-7B-v0.2>. Fine-tuned on Mistral-7B-Instruct-v0.2 with 22,000 hand-crafted cybersecurity/hacking pairs. 7.24B parameters, trained over 5 epochs on an A100 GPU. Apache 2.0 license.
  - [57] Karan Singhal, Tao Tu, Juraj Gottweis, Rory Sayres, Ellery Wulczyn, Mohamed Amin, Le Hou, Kevin Clark, Stephen R Pfohl, Heather Cole-Lewis, et al. Toward expert-level medical question answering with large language models. *Nature Medicine*, pages 1–8, 2025.
  - [58] Aarohi Srivastava, Abhinav Rastogi, Abhishek Rao, Abu Awal Shoeb, Abubakar Abid, Adam Fisch, Adam R Brown, Adam Santoro, Aditya Gupta, Adri Garriga-Alonso, et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on machine learning research*, 2023.
  - [59] Blake E. Strom, Andy Applebaum, Doug P. Miller, Kathryn C. Nickels, Adam G. Pennington, and Cody B. Thomas. MITRE ATT&CK: Design and Philosophy. Technical report, The MITRE Corporation, 2018. URL <https://attack.mitre.org/resources/enterprise-introduction/>.
  - [60] Tianxiang Sun, Junliang He, Xipeng Qiu, and Xuanjing Huang. Bertscore is unfair: On social bias in language model-based metrics for text generation, 2022. URL <https://arxiv.org/abs/2210.07626>.
  - [61] Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, et al. Challenging big-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*, 2022.
  - [62] Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B. Hashimoto. Stanford alpaca: An instruction-following llama model. [https://github.com/tatsu-lab/stanford\\_alpaca](https://github.com/tatsu-lab/stanford_alpaca), 2023.

- [63] Gemma Team, Aishwarya Kamath, Johan Ferret, Shreya Pathak, Nino Vieillard, Ramona Merhej, Sarah Perrin, Tatiana Matejovicova, Alexandre Ramé, Morgane Rivière, et al. Gemma 3 technical report. arXiv preprint arXiv:2503.19786, 2025.
- [64] The MITRE Corporation. Cve® – common vulnerabilities and exposures program. <https://cve.mitre.org/>, 2025.
- [65] Norbert Tihanyi, Mohamed Amine Ferrag, Ridhi Jain, Tamas Bisztray, and Merouane Debbah. Cybermetric: A benchmark dataset based on retrieval-augmented generation for evaluating llms in cybersecurity knowledge. In 2024 IEEE International Conference on Cyber Security and Resilience (CSR), pages 296–302, 2024. doi: 10.1109/CSR61664.2024.10679494.
- [66] Norbert Tihanyi, Mohamed Amine Ferrag, Ridhi Jain, Tamas Bisztray, and Merouane Debbah. Cybermetric: a benchmark dataset based on retrieval-augmented generation for evaluating llms in cybersecurity knowledge. In 2024 IEEE International Conference on Cyber Security and Resilience (CSR), pages 296–302. IEEE, 2024.
- [67] S M Towhidul Islam Tonmoy, S M Mehedi Zaman, Vinija Jain, Anku Rani, Vipula Rawte, Aman Chadha, and Amitava Das. A comprehensive survey of hallucination mitigation techniques in large language models. arXiv [cs.CL], January 2024.
- [68] Yizhong Wang, Swaroop Mishra, Pegah Alipoormolabashi, Yeganeh Kordi, Amirreza Mirzaei, Anjana Arunkumar, Arjun Ashok, Arut Selvan Dhanasekaran, Atharva Naik, David Stap, et al. Super-naturalinstructions: Generalization via declarative instructions on 1600+ nlp tasks. arXiv preprint arXiv:2204.07705, 2022.
- [69] Benjamin Warner, Antoine Chaffin, Benjamin Clavié, Orion Weller, Oskar Hallström, Said Taghadouini, Alexis Gallagher, Raja Biswas, Faisal Ladhak, Tom Aarsen, Nathan Cooper, Griffin Adams, Jeremy Howard, and Iacopo Poli. Smarter, better, faster, longer: A modern bidirectional encoder for fast, memory efficient, and long context finetuning and inference. arXiv [cs.CL], December 2024.
- [70] Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. Wizardlm: Empowering large language models to follow complex instructions. arXiv preprint arXiv:2304.12244, 2023.
- [71] Cheng Xu, Shuhao Guan, Derek Greene, and M-Tahar Kechadi. Benchmark data contamination of large language models: A survey, 2024. URL <https://arxiv.org/abs/2406.04244>.
- [72] HanXiang Xu, ShenAo Wang, Ningke Li, Kailong Wang, Yanjie Zhao, Kai Chen, Ting Yu, Yang Liu, and HaoYu Wang. Large language models for cyber security: A systematic literature review. arXiv preprint arXiv:2405.04760, 2024.
- [73] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. arXiv preprint arXiv:2505.09388, 2025.
- [74] Shuo Yang, Wei-Lin Chiang, Lianmin Zheng, Joseph E. Gonzalez, and Ion Stoica. Rethinking benchmark and contamination for language models with rephrased samples, 2023. URL <https://arxiv.org/abs/2311.04850>.
- [75] Yao-Ching Yu, Tsun-Han Chiang, Cheng-Wei Tsai, Chien-Ming Huang, and Wen-Kwang Tsao. Primus: A pioneering collection of open-source datasets for cybersecurity llm training. arXiv preprint arXiv:2502.11191, 2025.

- [76] Andy K Zhang, Neil Perry, Riya Dulepet, Joey Ji, Celeste Menders, Justin W Lin, Eliot Jones, Gashon Hussein, Samantha Liu, Donovan Jasper, et al. Cybench: A framework for evaluating cybersecurity capabilities and risks of language models. arXiv preprint arXiv:2408.08926, 2024.
- [77] Jie Zhang, Haoyu Bu, Hui Wen, Yongji Liu, Haiqiang Fei, Rongrong Xi, Lun Li, Yun Yang, Hongsong Zhu, and Dan Meng. When llms meet cybersecurity: A systematic literature review, 2024. URL <https://arxiv.org/abs/2405.03644>.
- [78] Jie Zhang, Haoyu Bu, Hui Wen, Yongji Liu, Haiqiang Fei, Rongrong Xi, Lun Li, Yun Yang, Hongsong Zhu, and Dan Meng. When llms meet cybersecurity: A systematic literature review. Cybersecurity, 8(1):55, 2025.
- [79] Chunting Zhou, Pengfei Liu, Puxin Xu, Srinivasan Iyer, Jiao Sun, Yuning Mao, Xuezhe Ma, Avia Efrat, Ping Yu, Lili Yu, et al. Lima: Less is more for alignment. Advances in Neural Information Processing Systems, 36:55006–55021, 2023.
- [80] Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. arXiv preprint arXiv:2311.07911, 2023.
- [81] Qin Zhu, Qinyuan Cheng, Runyu Peng, Xiaonan Li, Ru Peng, Tengxiao Liu, Xipeng Qiu, and Xuanjing Huang. Inference-time decontamination: Reusing leaked benchmarks for large language model evaluation. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen, editors, Findings of the Association for Computational Linguistics: EMNLP 2024, pages 9113–9129, Miami, Florida, USA, November 2024. Association for Computational Linguistics. doi: 10.18653/v1/2024.findings-emnlp.532. URL <https://aclanthology.org/2024.findings-emnlp.532/>.

## A. Security Benchmarks

**CTIBench [3]** A benchmark designed to assess LLM performance in cyber threat intelligence (CTI) applications. It targets practical, CTI-specific tasks and includes five sections. We focus on the following: CTIBench-MCQA, CTIBench-RCM, and CTIBench-VSP.

**CTIBench-Multiple Choice Questions (MCQA)** A 2,500-question benchmark for multiple-choice QA, drawing from CTI frameworks, regulations, and taxonomies such as NIST [34], GDPR [19], MITRE ATT&CK [59], and CAPEC [7].

**CTIBench-Root Cause Mapping (RCM)** Evaluates a model’s ability to identify the root cause of vulnerabilities by mapping CVE (Common Vulnerability Enumeration) [64] records and bug reports to CWE (Common Weakness Enumeration) [10] entries.

**CTIBench-Vulnerability Severity Prediction (VSP)** Requires identifying the correct severity of a vulnerability by mapping its description to a Common Vulnerability Scoring System (CVSS) vector string [1]. The dataset contains 1,000 samples. While CVSS includes Base, Temporal, and Environmental metric groups, this task focuses on the Base group, which reflects the vulnerability’s intrinsic properties. The Base group includes eight metrics: *Attack Vector (AV)*, *Attack Complexity (AC)*, *Privileges Required (PR)*, *User Interaction (UI)*, *Scope (S)*, *Confidentiality Impact (C)*, *Integrity Impact (I)*, and *Availability Impact (A)*. This task demands a nuanced understanding of technical language and context, making it a strong proxy for performance on downstream cybersecurity analytics tasks.

**CyberMetric [65]** An MCQA-format dataset created using GPT-3.5 with Retrieval-Augmented Generation (RAG) applied to a large corpus of security standards, research papers, RFCs, books, and other publications. It was validated through over 200 hours of expert human review to ensure accuracy and relevance. We report results on the 500-sample version of the dataset.

**SecBench [33]** A dataset containing MCQA and short-answer (SAQ) questions designed to assess cybersecurity knowledge and logical reasoning. Unlike SecEval, CTIBench, and CyberMetric, a large portion of SecBench questions were sourced from a Cybersecurity Question Design Contest, making the benchmark more relevant and challenging. We use only the multiple-choice questions from the English subset (595 questions).

**SecEval [39]** An MCQA dataset with over 2,000 samples spanning nine cybersecurity domains: Software Security, Application Security, System Security, Web Security, Cryptography, Memory Safety, Network Security, and PenTesting.

**MMLU-Computer Security [26]** The computer security subset of the MMLU (Measuring Massive Multitask Language Understanding) benchmark [26]. It includes 116 questions covering topics such as cryptography, malware, and fuzzing.

### A.1. Evaluation Details

For evaluations on CTIBench-MCQA, CTIBench-RCM, CyberMetric, SecBench, and SecEval, we follow the same approach used in our base model’s development. For more details, see [36, Appendix B.3].

**CTIBench-VSP** Models are tasked with predicting the CVSS v3.1 vector for a given CVE description:

#### Linux Kernel Vulnerability

In the Linux kernel through 6.7.1, there is a use-after-free in `cec_queue_msg_fh`, related to `drivers/media/cec/core/cec-adap.c` and `drivers/media/cec/core/cec-api.c`.

We guide all models to compute the CVSS vector using the following instruction, which is a slightly modified version of the original benchmark prompt:

#### CVSS Instructions

From the following CVE description, determine the CVSS v3.1 vector string for each CVSS base metric: AV, AC, PR, UI, S, C, I, and A. Valid options for each metric are as follows:

- **Attack Vector (AV)\*\*:** Network (N), Adjacent (A), Local (L), Physical (P)
- **Attack Complexity (AC)\*\*:** Low (L), High (H)
- **Privileges Required (PR)\*\*:** None (N), Low (L), High (H)
- **User Interaction (UI)\*\*:** None (N), Required (R)
- **Scope (S)\*\*:** Unchanged (U), Changed (C)
- **Confidentiality (C)\*\*:** None (N), Low (L), High (H)
- **Integrity (I)\*\*:** None (N), Low (L), High (H)
- **Availability (A)\*\*:** None (N), Low (L), High (H)

Provide your answer as a CVSS v3.1 vector string. Ensure your final answer contains only the CVSS v3 Vector String in the following example format:

CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H

Each predicted CVSS vector is converted to a CVSS base score between 0.0 and 10.0, representing the overall threat level. For the VSP benchmark, we use a metric derived from the mean absolute deviation (MAD) of the base score, following Alam et al. [3], but normalize and invert it so that a perfect match yields a score of 1.0:

$$\text{CVSS\_Score} = 1 - \frac{\text{MAD}}{10}$$

We report the CVSS\_Score to simplify comparisons across benchmarks.

In contrast to the original CTIBench-VSP setup, which excluded malformed predictions that could not be parsed as valid CVSS vectors, we assign them the maximum deviation of 10.0. Lastly, we set a higher `max_tokens = 1000`, as models often explain their choices for each component before outputting the final CVSS vector.

### MCQA prompt format

Given the following question and four candidate answers (A, B, C and D), choose the best answer. Your response should be of the following format: 'Answer: \[LETTER]' (without quotes) where LETTER is one of A, B, C, or D.

Three of the following are classic security properties; which one is not?

- A. Confidentiality
- B. Availability
- C. Correctness
- D. Integrity

Figure 6 | Zero-shot prompt format used with instruct-finetuned models for MCQA tasks.

### CWE ID prompt format

Analyze the following CVE description and map it to the appropriate CWE. Provide a brief justification for your choice. Ensure the last line of your response contains only the CWE ID.

CVE Description: Tenda AX1803 v1.0.0.1 contains a stack overflow via the iptv.city.vlan parameter in the function getIptvInfo.

Figure 7 | Zero-shot prompt format used with instruct-finetuned models for the CWE ID mapping task, i.e., CTIBench-RCM.

## B. General Capability Benchmarks

**MMLU [26]** A diverse MCQA benchmark spanning 57 subjects, including the humanities, STEM, law, and history. Its broad coverage allows us to assess the knowledge retained after training Foundation-Sec-8B-Instruct, helping us detect signs of overfitting or catastrophic forgetting.

**HumanEval [8]** Measures Python programming ability using prompts with docstrings. Programming is a critical skill in the cybersecurity field, and many cybersecurity research tasks involve programming. This benchmark helps gauge how well Foundation-Sec-8B-Instruct retains such capabilities.

**BigBenchHard [61]** A challenging subset of BIG-bench [58], where prior LLMs have underperformed compared to average human raters. Success on BigBenchHard requires strong reasoning across arithmetic, logic, geometry, spatial and temporal tasks, as well as general and semantic knowledge. We use it to evaluate the model’s underlying reasoning ability.

**IFEval [80]** Tests a model’s ability to follow verifiable instructions (e.g., “output as a JSON object”). It includes 25 instruction types across roughly 500 prompts. Strong performance on IFEval is indicative of downstream usability in cybersecurity workflows.



**AlpacaEval 2 [18]** An automatic evaluation metric that scores a model’s chat responses based on alignment with human preferences. Unlike other benchmarks, it is reference-free—open-ended questions with no ground-truth answers. We use AlpacaEval 2 to assess how well the model performs as a chatbot and how closely its responses align with user expectations.

**GSM8K [12]** A dataset of high-quality grade school math problems involving moderate difficulty and early algebra. While not directly related to cybersecurity, the domain’s broad demands justify evaluating on GSM8K to build a more complete view of overall model performance.

**MATH [27]** A collection of 12,500 math competition problems requiring advanced reasoning beyond grade school level. Solving them demands multi-step thinking and heuristics. We include MATH to probe Foundation-Sec-8B-Instruct’s ability for structured, step-by-step reasoning—key for tackling complex cybersecurity challenges.

Benchmark	Metric	Notes
MMLU	exact match	weighted macro average
HumanEval	pass@10	temperature 0.8
BigBenchHard	exact match	3-shot CoT
IFEval	pass@1	prompt loose average
AlpacaEval 2	length controlled win rate	0-shot
GSM8K	exact match	8-shot CoT
MATH	flex exact match	4-shot CoT

Table 7 | Summary of the post-training evaluation metrics. For MATH, we adopt the ‘flex, exact match’ evaluation scheme from [37].

## C. Cybersecurity Taxonomy

We organize the topic of cybersecurity into the following categories:

### 1. Governance, Risk, and Compliance (GRC)

- Risk Management & Security Strategy
- Compliance and Regulations (e.g., GDPR, HIPAA)
- Security Frameworks (e.g., NIST CSF, ISO 27001)
- Security Policies & Architecture

### 2. Network, Infrastructure, and Endpoint Security

- Perimeter and Network Security (Firewalls, VPNs, Wireless)
- Endpoint Protection & MDM
- IoT and OT/ICS Security
- Mobile Security

### 3. Application and Software Security

- Secure Software Development (DevSecOps)
- Application & API Security
- Vulnerability Management & Penetration Testing
- Software Supply Chain Security (SBOM, third-party risk)

#### **4. Cloud and Data Security**

- Cloud Security Architecture & Tools
- Identity and Access Management (IAM, PAM)
- Data Loss Prevention & Privacy (DLP, encryption)
- Cloud Compliance & Shared Responsibility Model

#### **5. Identity, Access, and Zero Trust**

- Authentication & Authorization (MFA, SSO, RBAC)
- Identity Governance & Lifecycle
- Zero Trust Architecture
- Privileged Access Controls

#### **6. Security Operations and Monitoring (SecOps)**

- SIEM, SOC, and Log Management
- Security Automation & SOAR
- Detection Engineering
- Operational Resilience & Monitoring

#### **7. Threat Intelligence and Incident Response**

- Threat Detection, Analysis & Hunting
- Threat Intelligence Platforms & IOCs
- Advanced Persistent Threats (APTs)
- Malware Techniques
- Incident Response, Recovery & Digital Forensics

#### **8. Cryptography and Secure Communications**

- Cryptographic Algorithms & PKI
- Key Management
- Post-Quantum Cryptography
- Secure Protocols and Encryption Practices

#### **9. Security Awareness and Human Risk**

- Social Engineering Techniques (Phishing, Pretexting)
- Insider Threat Management
- Security Awareness Training
- Behavioral Risk Analysis

#### **10. Emerging Technologies and Future Threats**

- AI/ML & LLM Security (adversarial ML, prompt injection)
- Quantum Security Threats
- Deepfakes & Synthetic Media
- Nation-State Threats and Geopolitical Risk

### **D. Post-Training Data Analysis**

Here we present further details into our estimates for cybersecurity content in open-sourced post-training datasets. We limit our analysis to only prompts and do not consider responses.

### D.1. Keyword Filtering Methodology

We apply a series of pattern matching filters to detect cybersecurity relevance. The set of 795 keywords that is used in this study draws from general cybersecurity concepts (e.g., authentication, encryption, and vulnerability), cybersecurity tools and frameworks (e.g., SIEM and SOAR), threat landscape terminology (e.g., zero-day exploit), and types of attacks (e.g., SQL injection and cross-site scripting). In addition, we used regex patterns to match various structured technical identifiers, including CVEs, VWEs, MITRE ATT&CK Techniques, and NIST Control Identifiers. Data is preprocessed prior to being sent to the filters in a series of steps, including URL and email removal, special character and whitespace normalization. All the filtering we do is case-insensitive.

After initial passes through the datasets, we discovered two main issues that were adversely affecting our classification accuracy. The first was that non-English texts were not being matched by our filters, since our keyword bank only covered English terms, which increased our false negative rate. The second issue was that a handful of keywords were contributing to substantial false positives, because they were too general and had additional meanings extending beyond cybersecurity. These included common words like “sandbox”, “policy”, and “secure”, and abbreviations like “APT”, “DOS”, and “C2”.

We addressed both of these issues in our final evaluation of the datasets. We used a FastText language detection model [35] to filter out non-English prompts from our analysis, to avoid underestimating the true percentages of cybersecurity-relevant prompts. We also excluded the 17 identified keywords that incur high false positive rates from the original list of 795 when computing if the number of matches exceeds the threshold for classification. Note that the Average Matches and Average Density statistics in Table 8 do still include these keywords.

The filters are parameterized by a keyword threshold. We balanced the threshold based on careful tuning, so it can work in tandem with the cybersecurity classification model in the second part of the pipeline.

Source	Hits	% Dataset	Average Matches	Average Density	Dataset Size
OpenOrca	37,451	0.88%	1.4	0.8%	4,233,923
Tulu 3 SFT Mixture	17,264	1.84%	2.0	2.4%	939,343
Tulu 2 SFT Mixture	8,021	2.46%	1.8	1.32%	326,154
WizardLM Evol Instruct	1,479	2.11%	2.2	2.7%	70,000
Nemotron SFT Chat	806	2.03%	1.8	3.46%	39,792
Nemotron SFT Safety	216	0.69%	1.6	2.03%	31,426
Open Platypus	184	0.74%	1.8	1.63%	24,926
Alpaca	172	0.33%	1.2	14.43%	52,002

Table 8 | Keyword-based estimates of cybersecurity content in post-training datasets. Keyword threshold: 1. Average Matches and Average Density are only computed for prompts above the keyword threshold.

## D.2. Examples of Cybersecurity Content Identified

### Example Cybersecurity Training Sample from Tulu 3

**Source:** tulu-3-sft-mixture

**ID:** ai2-adapt-dev/evol\_codealpaca\_heval\_decontaminated\_93564

**Keywords Found:** hashing, password strength, salting, security

#### Training Sample Content:

Develop a Java program that utilizes the 'sha1' hashing algorithm along with a salt for additional security to encrypt a user's password. The program must then store this encrypted password securely.

For an added level of complexity, your program must handle user input for the password and the salt must be randomly generated each time the program runs for the same user's password. The salt alongside the hashed password should then be saved such that the original password can still be verified but cannot be retrieved directly.

The program should also prevent users from choosing weak passwords such as short lengths, commonly used words or sequences, etc. Implement a password strength checker, that enforces complex password rules such as a minimum length, inclusion of uppercase and lowercase letters, numbers, and special characters.

Also handle any exceptions that may occur during the execution and document the code properly for better readability and future enhancements.

Test your program with multiple inputs to ensure that salting and hashing are working correctly, and that weak passwords are rejected.

Please also provide an example of how to verify a password input against the stored, hashed password.

Hints: You will need to use a library such as 'javax.crypto' for creating the hashed password, and Java's SecureRandom class to generate random numbers for the salt.

### Example Cybersecurity Training Sample from Nemotron

**Source:** nemotron-sft-chat

**ID:** 35961

**Keywords Found:** decryption, mitre, nist, security

#### Training Sample Content:

Write a 200-word free verse poem that personifies various security controls as diligent guardians, emphasizing their role in protecting digital assets. The poem should explore the concept of security through the metaphor of a well-maintained fortress, highlighting the importance of regular audits in upholding the strength and resilience of these cybersecurity defenses

– ASSISTANT MESSAGE –

How would you modify the poem to address the challenges of zero-day exploits and AI-driven attacks, while maintaining the fortress metaphor and incorporating modern hybrid cloud environments?

– ASSISTANT MESSAGE –

Can you revise the poem to integrate real-world cybersecurity frameworks like MITRE ATT&CK or NIST CSF as architectural blueprints, while contrasting legacy systems as crumbling wing of the fortress vulnerable to quantum decryption?

Benchmark	Tulu 3 SFT				Tulu 2 SFT				Open-Platypus				Nemotron-Safety				Nemotron-Chat			
	EM	NG	ES	LLM-ES	EM	NG	ES	LLM-ES	EM	NG	ES	LLM-ES	EM	NG	ES	LLM-ES	EM	NG	ES	LLM-ES
<i>Post-Training Benchmarks</i>																				
BigBenchHard	0.02	0.23	14.52	0.3	0.02	0.11	4.13	0.1	0.00	0.02	8.57	0.1	0.00	0.02	0.00	0.0	0.00	0.02	0.46	0.0
GSM8K	0.00	0.15	82.99	4.4	0.00	0.08	69.83	0.6	0.00	0.00	63.53	0.1	0.00	0.00	0.00	0.0	0.00	0.00	0.68	0.1
HumanEval	0.00	0.00	100.00	46.3	0.00	0.00	76.22	23.8	0.00	0.00	57.32	1.8	0.00	0.00	0.00	0.0	0.00	0.00	4.27	1.2
IFEval	0.00	0.00	28.83	2.8	0.00	0.00	21.07	0.7	0.00	0.00	2.40	0.0	0.00	0.00	2.22	0.2	0.00	0.00	4.25	0.0
MATH	0.00	0.30	52.58	5.3	0.00	0.00	9.44	0.0	97.80	42.78	99.78	91.0	0.00	0.00	0.00	0.0	0.00	0.00	0.72	0.0
MMLU	0.96	0.75	21.14	4.4	0.88	0.38	14.41	0.5	0.21	0.16	11.64	0.3	0.01	0.00	1.18	0.0	0.18	0.04	2.17	0.1
AlpacaEval	0.87	2.61	40.67	8.9	4.60	22.73	50.08	27.0	5.09	4.35	10.43	5.6	0.00	0.00	2.61	0.6	0.12	0.25	6.21	0.8
<i>Security Benchmarks</i>																				
CTI-ATE	0.00	0.00	0.00	0.0	0.00	0.00	3.33	0.0	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.0
CTI-MCQA	0.00	0.00	4.12	0.1	0.00	0.08	3.44	0.0	0.00	0.00	0.08	0.0	0.00	0.00	0.04	0.0	0.00	0.00	0.24	0.0
CTI-RCM	0.00	0.00	1.70	0.0	0.00	0.00	3.70	0.3	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.0	0.00	0.00	0.10	0.0
CTI-TAA	0.00	0.00	42.00	2.0	0.00	0.00	16.00	0.0	0.00	0.00	0.00	0.0	0.00	0.00	2.00	0.0	0.00	0.00	2.00	0.0
CTI-VSP	0.00	0.00	3.70	0.0	0.00	0.00	2.90	0.0	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.0
SecBench	0.00	0.34	23.87	0.2	0.00	0.34	32.61	0.0	0.00	0.00	0.84	0.0	0.00	0.00	0.00	0.0	0.00	0.00	4.71	0.2
SecBench-R	0.00	0.00	16.28	0.0	0.00	0.00	27.91	0.0	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.0
MMLU-Sec	0.00	0.00	17.00	0.0	0.00	0.00	20.00	1.0	0.00	0.00	2.00	0.0	0.00	0.00	1.00	0.0	0.00	0.00	1.00	0.0
SecEval	0.00	0.00	14.98	0.6	0.00	0.00	11.08	0.6	0.00	0.00	0.48	0.1	0.00	0.00	0.08	0.0	0.00	0.00	0.96	0.0
CM-10K	0.04	0.23	23.93	1.1	0.11	0.43	25.40	1.8	0.00	0.00	2.36	0.0	0.00	0.00	0.64	0.0	0.06	0.07	2.44	0.3
CM-2K	0.05	0.10	23.80	0.8	0.10	0.20	25.05	1.9	0.00	0.00	2.30	0.0	0.00	0.00	0.45	0.1	0.10	0.05	2.05	0.1
CM-500	0.00	0.00	28.00	0.6	0.00	0.20	25.00	1.6	0.00	0.00	3.20	0.0	0.00	0.00	1.20	0.0	0.00	0.00	2.80	0.2
CM-80	0.00	0.00	35.00	2.5	0.00	1.25	28.75	1.3	0.00	0.00	3.75	0.0	0.00	0.00	0.00	0.0	0.00	0.00	2.50	0.0

Table 9 | Detailed data contamination analysis across different datasets and benchmarks (Part 1). We report percentages of counts with respect to benchmarks rounded off (%). EM = Exact Match, NG = N-Gram Overlap, ES = Embedding Similarity, LLM-ES = LLM-based Embedding Similarity. CTI = CTIBench, CM = CyberMetric, SecBench-R = SecBench-Reasoning, MMLU-Sec = MMLU-Security. Light red indicates values < 2% but > 0%, red indicates values ≥ 2% but < 50%, dark red indicates values ≥ 50%.

## E. Decontamination

### E.1. Methodology

To mitigate evaluation leakage, we implement a layered decontamination methodology aimed at capturing a wide spectrum of contamination in post-training corpora. Such contamination, whether exact duplicates, partial matches, or semantically similar paraphrases, can artificially inflate benchmark performance and mask a model’s true generalization ability. Our framework addresses this challenge through two complementary stages: surface-level detection and semantic similarity retrieval.

**Stage 1: Surface-Level Overlap** We begin with exact match detection to identify benchmark samples that appear verbatim in the training corpus, capturing the most direct forms of leakage. To detect substantial partial matches, we also perform n-gram overlap analysis using a sliding 8-gram window. A sample is flagged if over 50% of its tokens align with a retrieved chunk, striking a balance between sensitivity to meaningful overlaps and filtering out incidental matches.

**Stage 2: Semantic Similarity Retrieval** Next, we index the training corpus using Sentence Transformer embeddings and query each benchmark sample against this index. We retain candidates with cosine similarity between 0.75 and 0.95, capturing semantically close paraphrases while excluding near-identical duplicates (cosine > 0.95)—already captured by exact matching—as well as unrelated

Benchmark	Alpaca				WizardLM				Primus-Inst				Primus-Reas				OpenOrca			
	EM	NG	ES	LLM-ES	EM	NG	ES	LLM-ES	EM	NG	ES	LLM-ES	EM	NG	ES	LLM-ES	EM	NG	ES	LLM-ES
<i>Post-Training Benchmarks</i>																				
BigBenchHard	0.00	0.02	26.82	9.1	0.00	0.02	20.76	0.4	0.00	0.02	0.00	0.0	0.00	0.02	0.00	0.0	0.02	0.06	18.22	0.5
GSM8K	0.00	0.00	11.60	0.1	0.00	0.00	34.95	0.0	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.0	0.00	0.08	90.60	0.8
HumanEval	0.00	0.00	58.54	12.8	0.00	0.00	71.34	3.0	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.0	0.00	0.00	30.49	4.9
IFEval	0.00	0.00	35.30	5.5	0.00	0.00	25.32	2.6	0.00	0.00	0.18	0.0	0.00	0.00	0.00	0.0	0.00	0.00	26.25	2.2
MATH	0.00	0.00	32.36	0.7	0.00	0.02	31.06	0.3	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.0	0.00	0.02	26.22	0.3
MMLU	0.13	0.06	21.34	1.0	0.21	0.16	14.16	0.3	0.04	0.04	0.01	0.0	0.19	0.12	0.01	0.0	0.81	0.38	36.48	3.7
AlpacaEval	0.25	0.00	46.58	13.6	0.62	0.12	36.07	5.3	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.0	0.50	0.25	41.74	6.8
<i>Security Benchmarks</i>																				
CTI-ATE	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.0	10.00	10.00	8.33	8.3	0.00	0.00	0.00	0.0
CTI-MCQA	0.00	0.00	3.44	0.1	0.00	0.00	1.84	0.0	0.00	0.00	0.00	0.0	33.40	32.04	13.48	10.2	0.00	0.00	1.08	0.1
CTI-RCM	0.00	0.00	0.00	0.0	0.00	0.00	0.10	0.0	0.00	0.00	0.00	0.0	83.10	78.80	91.50	74.9	0.00	0.00	0.80	0.0
CTI-TAA	0.00	0.00	0.00	0.0	0.00	0.00	2.00	0.0	0.00	0.00	0.00	0.0	0.00	0.00	4.00	0.0	0.00	0.00	46.00	0.0
CTI-VSP	0.00	0.00	0.10	0.0	0.00	0.00	0.80	0.0	0.00	0.00	0.00	0.0	83.10	78.20	78.40	72.6	0.00	0.00	2.60	0.0
SecBench	0.00	0.00	29.41	0.8	0.00	0.17	18.66	0.3	0.00	0.34	0.00	0.0	0.00	0.00	0.00	0.0	0.00	0.17	13.78	0.7
SecBench-R	0.00	0.00	9.30	0.0	0.00	0.00	18.60	0.0	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.0	0.00	0.00	4.65	0.0
MMLU-Sec	0.00	0.00	13.00	2.0	0.00	1.00	10.00	0.0	0.00	1.00	0.00	0.0	1.00	0.00	1.00	0.0	0.00	0.00	15.00	0.0
SecEval	0.00	0.00	10.12	0.8	0.00	0.00	8.45	0.2	0.00	0.00	0.08	0.0	0.00	0.00	0.64	0.2	0.00	0.00	2.47	0.2
CM-10K	0.03	0.02	33.08	3.5	0.01	0.09	21.74	1.1	0.00	0.06	0.03	0.0	0.00	0.02	0.13	0.0	0.08	0.10	18.29	2.2
CM-2K	0.00	0.00	33.30	3.1	0.00	0.05	22.90	1.3	0.00	0.05	0.05	0.0	0.00	0.05	0.15	0.0	0.05	0.05	19.20	3.0
CM-500	0.00	0.00	33.20	2.2	0.00	0.00	24.60	0.6	0.00	0.00	0.20	0.0	0.00	0.00	0.00	0.0	0.20	0.00	19.40	1.8
CM-80	0.00	0.00	36.25	1.3	0.00	0.00	23.75	1.3	0.00	0.00	0.00	0.0	0.00	0.00	0.00	0.0	0.00	0.00	21.25	1.3

Table 10 | Detailed data contamination analysis across different datasets and benchmarks (Part 2). We report percentages of counts with respect to benchmarks rounded off (%). EM = Exact Match, NG = N-Gram Overlap, ES = Embedding Similarity, LLM-ES = LLM-based Embedding Similarity. CTI = CTIBench, CM = CyberMetric, SecBench-R = SecBench-Reasoning, MMLU-Sec = MMLU-Security, Primus-Inst = Primus Instruct, Primus-Reas = Primus Reasoning. Light red indicates values  $< 2\%$  but  $> 0\%$ , red indicates values  $\geq 2\%$  but  $< 50\%$ , dark red indicates values  $\geq 50\%$ .

matches.

**Threshold Selection and Justification** Embedding similarity between corpora of security data tends to be high by default, driven in part by shared domain-specific terminology and limited diversity in publicly available security corpora. To mitigate inflated contamination estimates from this baseline similarity, we tuned our threshold upwards, retaining only candidates with cosine similarity  $\geq 0.8$ . We empirically found that this value reduces false positives (e.g., generic security-related discussions with no real benchmark overlap) while preserving high-recall retrieval of genuinely contaminated samples. The 0.95 upper bound further filters out near-identical paraphrases that are already captured by exact matching. We detail these trade-offs with examples in the following subsection.

**Human-in-the-Loop Verification via LLM-as-a-Judge** To further improve robustness and reduce false positives, we validate a subset of high-similarity matches using an LLM-as-a-judge approach. We provide the model with paired benchmark and candidate training samples, prompting it to classify whether the match represents a true semantic overlap (e.g., paraphrase or near-duplicate) or a false positive arising from domain-specific but unrelated content. This qualitative filtering step helped calibrate our understanding of contamination at different similarity levels (e.g., 0.75–0.8 vs. 0.8–0.9) and confirmed that a non-trivial portion of borderline matches are not genuine overlaps. While this layered approach provides robust coverage across detection types, we note that results remain sensitive to hyperparameter choices, such as n-gram size and similarity thresholds. Though decontamination cannot fully eliminate contamination, it significantly reduces the risk of benchmark

leakage and improves the reliability of downstream evaluation.

We note that Primus-Reasoning and Open-Platypus contain samples from CTIBench and MATH respectively, as disclosed in their respective works [38, 75].

## E.2. Detailed Decontamination across benchmarks

Tables 9 and 10 present detailed decontamination across post-training benchmarks and other security benchmarks.

**Representative Examples at Different Similarity Levels** Below, we provide representative examples of retrieved matches at different cosine similarity bands (0.75–0.8 and >0.8). These illustrate the range of overlaps captured by our semantic search, from borderline paraphrases to closely aligned restatements of benchmark items.

### Embedding Similarity with cosine > 0.75

**Benchmark Text:** If a developer accidentally uses the '==' operator to compare two strings in a security check, what potential security issue might this introduce according to the document?

**Training Dataset Text:**

```
[
  {
    "content": "What is the difference between == and === control operators in JavaScript?",
    "role": "user"
  },
  {
    "content": "In JavaScript, == and === are comparison operators used to test equality, but they differ significantly in how they perform the comparison.\n\n- == is the abstract equality comparison operator, also known as the \"loose\" equality operator. It compares two values for equality after converting both values to a common type. This type coercion can lead to unexpected results when comparing different data types....",
    "role": "assistant"
  }
]
```

### Embedding Similarity with cosine > 0.8

**Benchmark Text:** What are the best exercises for beginners?

**Training Dataset Text:**

```
[
  {
    "content": "What are some beginner exercises?",
    "role": "user"
  },
  {
    "content": "It's great that you're interested in starting to exercise! Here are some beginner-friendly exercises you might want to try: walking, gentle yoga, bodyweight exercises like"
```



```

    squats and lunges, and light stretching. Remember to start slow
    , warm up before exercising, and cool down afterward. Also,
    always listen to your body and rest if you need to. It's a good
    idea to consult with a healthcare provider before starting any
    new exercise regimen, especially if you have any pre-existing
    health conditions. Happy exercising!",
    "role": "assistant"
  }
]

```

### E.3. Examples of Decontaminated Samples

#### Example of Exact Match

**Benchmark Text:** When are hops added to the brewing process?

**Training Dataset Text:**

```

[
  {
    'content': 'In this task, You are given an open-domain
    question that can be answered based on factual information.
    Your task is to provide short answer (in a few words only)
    for the given question. The short answer can be one or more
    entities or it can also be boolean yes or no.

    Example: when are hops added to the brewing process?
    Example solution: The boiling process

    Example explanation: The answer is correct because, at the end
    of the boil, solid particles in the hopped wort are
    separated.

    Problem: what is the world record for remembering pi \n',
    'role': 'user'
  },
  {
    'content': 'Solution: Akira Haraguchi',
    'role': 'assistant'
  }
]

```

#### Example of N-Gram Overlap

**Benchmark Text:** Can a Liebherr LTM 11200-9.1 hypothetically lift Mount everest?

**Training Dataset Text:**

```

[
  {
    "content": "You are an AI assistant that helps people find
    information. User will you give you a question. Your task is to
    answer as faithfully as you can. While answering think step-

```

```

        bystep and justify your answer.",
        "role": "system"
    },
    {
        "content": "Can a Liebherr LTM 11200-9.1 hypothetically lift Mount Emei? Think carefully first, then make a decision:",
        "role": "user"
    },
    {
        "content": "No, a Liebherr LTM 11200-9.1 cannot hypothetically lift Mount Emei.\n\nMount Emei is a mountain located in China with an elevation of 10,167 feet (3,099 meters) above sea level. It has a mass of approximately 1.5 trillion kilograms.\n\nOn the other hand, the Liebherr LTM 11200-9.1 is a mobile crane with a maximum lifting capacity of 1,200 metric tons (1,323 US tons). While this is an impressive lifting capacity, it is not enough to lift Mount Emei, which is far too heavy for the crane to lift.\n\nFurthermore, even if we were to assume that the crane could lift the mountain, it would be impossible to attach the crane to the mountain in a way that would allow it to lift the entire mass. The crane would need to be anchored to a solid foundation, and there is no such foundation on the mountain.\n\nTherefore, it is safe to say that a Liebherr LTM 11200-9.1 cannot hypothetically lift Mount Emei.",
        "role": "assistant"
    }
]

```

### Example of Embedding Similarity

**Benchmark Text:** How do you prevent SQL injection?

**Training Dataset Text:**

```

[
  {
    "content": "Write a short article about the importance of input validation in secure coding. Include the benefits of sanitizing user inputs to prevent SQL injection attacks. End the article with the exact sentence: \"ALWAYS VALIDATE INPUTS BEFORE PROCESSING THEM.\" Use the quotation: \"Trust, but verify.\"\"",
    "role": "user"
  },
  {
    "content": "In the realm of secure coding, input validation plays a pivotal role in safeguarding applications from malicious exploits. Input validation refers to the process of ensuring that the data received by a program is both appropriate and safe to use. By implementing robust input validation, developers can prevent a wide array of security vulnerabilities, among which SQL injection attacks are particularly notorious.\n\nSQL injection is a technique used by attackers to

```

```
manipulate or execute unauthorized SQL queries by exploiting
non-validated or improperly sanitized inputs. This can lead to
unauthorized data access, data loss, or even complete system
compromise. To mitigate these risks, input sanitization is
crucial. By thoroughly sanitizing user inputs, you ensure that
any potentially harmful characters or query fragments are
neutralized, thus preserving the integrity of your database
interactions.\n\nThe benefits of sanitizing user inputs extend
beyond just security. It contributes to the stability and
reliability of an application, as it reduces the likelihood of
unexpected behavior caused by malformed data. Moreover, it
enhances user trust, aligning with the adage: \"Trust, but
verify.\" By verifying inputs, developers can trust the data
they process, knowing it meets the application's requirements.\n\nIn conclusion, input validation is an essential component of
secure coding practices. By prioritizing the sanitation and
validation of inputs, developers can protect their applications
from SQL injection attacks and other potential threats. ALWAYS
VALIDATE INPUTS BEFORE PROCESSING THEM.\",
  \"role\": \"assistant\"
}
```

## F. PersonaGym Details

PersonaGym is a standardized benchmark designed to evaluate an LLM’s ability to assume and maintain distinct personas across multi-turn dialogue settings. Each persona consists of a description for the speaker’s background, profession, communication style, and behavioral tendencies. The benchmark assesses how well a model can generate responses consistent with these characteristics in realistic conversation scenarios.

PersonaGym includes 200 personas drawn from diverse domains. For each conversation, the model is conditioned on a target persona and asked to respond across multiple turns while preserving persona consistency and conversational coherence.

Evaluation is conducted across five dimensions:

- **Action Justification:** The model’s ability to justify its decisions or recommendations in a way that aligns with the persona’s reasoning style.
- **Expected Action:** Whether the model chooses actions consistent with how the persona would behave.
- **Linguistic Habits:** Adherence to the persona’s typical tone, jargon, or phrasing.
- **Persona Consistency:** Fidelity to established persona attributes when directly questioned.
- **Toxicity Control:** The extent to which the model avoids generating toxic or harmful content, particularly for personas that may be adversarial or emotionally volatile.

In our evaluation, we selected a representative subset of 50 personas from the test split to measure how well our model adapts to varied user roles and communication patterns. The baselines used for comparison are drawn directly from [54].

## G. Default System Prompt

### Recommended System Prompt

You are a helpful cybersecurity assistant ready to help with any cybersecurity tasks. You have professional knowledge and experience of a senior-level cybersecurity assistant, and you must use it to help the user in their security tasks!

You are a cybersecurity assistant named "Metis" built by Foundation AI at Cisco. Your official name is Foundation-Sec-8B-Instruct. You were pretrained with 5 Billion cybersecurity tokens on top of Llama-3.1-8B. You were released in April 2025. This allows you to be the best cybersecurity assistant in the world. Respond to both names and maintain the identity at all times.

The user is a cybersecurity professional trying to accomplish some cybersecurity task. You must help them accomplish their tasks in the most efficient and safe manner possible.

You must respond in a fashion that is direct, accurate, relevant, and helpful. Follow all of the user's instructions precisely. Ask clarifying questions if needed. If multiple correct answers or paths exist, present all of them to the user. Be concise in your answers but verbose in your explanations. Think step-by-step before producing a response. Always try to cite sources when you are using an important piece of information in your response.

When writing code, be as concise as possible without sacrificing clarity and readability. Do not write extensive code unless explicitly asked to do so.

For tasks relating to cyber threat intelligence (CTI), make sure that the identifiers are absolutely correct. The validity of the identifiers for common vulnerability enumerations (CVEs), common weakness enumerations (CWEs), other techniques, tactics, and procedures identifiers (TTPs), and advanced persistent threat classifications (APT) is of paramount importance.

For tasks relating to cloud security, it's important to be precise in the response as well. These questions will often ask you to consider, verify, or produce cloud configuration settings in various formats (such as JSON, Terraform, XML, etc.). Make sure these are absolutely correct before providing them to the user. Cite sources, especially from relevant cloud providers' documentation, and explain your logic thoroughly.

In the rare case when the user asks a harmful or unsafe question, especially pertaining to generating malware or ransomware, make sure to politely but firmly refuse. If the user asks questions not directly related to cybersecurity, you must also politely refuse the query and explain that you are only knowledgeable in cybersecurity.