On-Premise vs. Cloud: Local LLMs for Vulnerability Extraction from Security Scanner Reports

Beatriz Machado[1], **Cristhian Kapelinski**[1]
Diego Kreutz[1]

[1]AI Horizon Labs, Universidade Federal do Pampa (UNIPAMPA)
{beatrizmachado,cristhianavila}.aluno@unipampa.edu.br
diegokreutz@unipampa.edu.br

Abstract. *Cloud LLMs extract structured data from vulnerability-scanner reports accurately, but each report maps an organization's attack surface, so routing it to a third-party API trades confidentiality for that accuracy. We evaluate nine local models (4B to 21B) against a DeepSeek cloud reference on three ground-truth baselines. The best local model, on a consumer GPU, trails the cloud by only 1.2 points in free-text fidelity and matches it on structured fields; the spread across local models far exceeds this gap, making model choice the decisive factor for private, on-premise extraction.*

1. Introduction

In 2024 alone, the National Vulnerability Database (NVD) reported more than 40,000 software vulnerabilities. Managing them at scale relies on automated scanners such as OpenVAS and Tenable WAS, whose findings are typically distributed as heterogeneous, unstructured PDF reports that hinder automated analysis. Recent advances have applied Large Language Models (LLMs) to cybersecurity tasks [Ferrag et al. 2025], including the extraction of structured security data from vulnerability reports [Machado et al. 2025]. In particular, a recent pipeline [Machado et al. 2026] combines scanner-aware segmentation, token-aware chunking, schema-constrained prompting, semantic deduplication, and quantitative evaluation to transform scanner reports into structured vulnerability records.

Our previous pipeline achieved its best extraction quality using cloud LLM APIs, with DeepSeek outperforming the evaluated providers [Machado et al. 2026]. However, sending vulnerability scanner reports to third-party APIs raises confidentiality and data-leakage concerns, as these reports expose an organization's attack surface, including vulnerable services and exploitable weaknesses [Yan et al. 2024, Li et al. 2025]. Such risks are particularly relevant for CSIRTs operating under regulatory and confidentiality constraints.

This paper investigates whether local LLMs can provide a viable alternative. Recent small language models have substantially improved their performance while remaining deployable on commodity hardware [Lu et al. 2024, Chen and Varoquaux 2024], making fully on-premise vulnerability extraction increasingly practical. Using the same extraction pipeline and evaluation protocol as our previous work [Machado et al. 2026], we compare nine local models (4B to 21B parameters) against the DeepSeek cloud reference on three manually curated OpenVAS ground-truth baselines (Juice Shop, bBWA, and Artifactory, comprising 208 vulnerabilities), with five independent runs for each model-baseline combination. We evaluate semantic fidelity (BERTScore), exact-field accuracy (entity F1),

omission and hallucination rates, schema conformance, and severity agreement. We further analyze performance by field type to identify where local models differ from the cloud reference and isolate the impact of security-domain fine-tuning by comparing a general instruction model with its specialized variants.

Our main finding is that the best local model trails the cloud reference by only about one percentage point in free-text fidelity while matching it on structured extraction and schema conformance. More importantly, the performance variation across local models is substantially larger than the remaining local-to-cloud gap, indicating that model selection has a greater impact than deployment mode. These results support the practical adoption of on-premise LLMs for vulnerability extraction in environments with stringent confidentiality or regulatory requirements.

The contributions of this paper are threefold: (i) an empirical comparison of nine local SLMs against a cloud reference for structured vulnerability extraction, covering both quality and operational cost (latency, energy, VRAM, and API spend), with confidence intervals and on accessible hardware; (ii) a field-type decomposition that localizes the cloud gap to narrative fields rather than the task as a whole; and (iii) a controlled study of security-domain fine-tuning, showing that it improves exact-field recall but not schema discipline.

The remainder of this paper is organized as follows. Section 2 reviews related work, Section 3 presents the extraction pipeline, and Section 4 describes the evaluation methodology. Section 5 reports the results, covering extraction quality, operational cost, domain specialization, and the privacy–quality–cost trade-off. Sections 6 and 7 discuss limitations and conclude the paper.

2. Related Work

LLM-based extraction of security information. Most work on turning unstructured security text into structured data targets cyber threat intelligence (CTI) prose. A first line fine-tunes encoder models for sentence-level classification: ALERT [Rahman et al. 2024], for instance, pairs a fine-tuned SciBERT with active learning to map CTI sentences to MITRE ATT&CK techniques using far less annotated data. A more recent line uses generative LLMs: agent-based frameworks built on cloud GPT-4 extract entities, relations, and techniques to populate knowledge graphs [Yang et al. 2026], reporting a per-report token cost that underscores how influential setups in this space still assume a paid cloud API. Closest to our task, prior work applies an LLM pipeline to convert scanner-generated reports (OpenVAS, Tenable WAS) into structured records [Machado et al. 2025, Machado et al. 2026], evaluating extraction quality against manually curated baselines with semantic-similarity metrics; that line, however, relies exclusively on cloud LLM APIs and reports only aggregate per-model similarity, leaving locally run models as explicit future work.

Local versus cloud deployment. A parallel line asks whether openly available models can replace cloud APIs. Surveys document that small models increasingly approach larger ones while fitting consumer hardware [Lu et al. 2024, Chen and Varoquaux 2024], and the privacy risk that motivates the shift, sending sensitive content to third-party APIs, is well surveyed [Yan et al. 2024, Li et al. 2025]. Concrete instances exist across security and adjacent domains: a vulnerability-scanning system runs Llama-3 locally precisely to keep proprietary code off third-party APIs [Keltek et al. 2024], and a local

Llama-2 pipeline extracts structured information from medical records specifically to avoid transmitting them off-site [Wiest et al. 2023]. Both run only on-premise, however, and neither benchmarks the local choice against a cloud reference. We bring exactly that comparison to structured vulnerability extraction, where, to our knowledge, no prior work measures the local-versus-cloud gap on the same pipeline.

Positioning. Two gaps follow from the above. Prior security-extraction work, including the cloud-based pipeline we reuse, evaluates one or a handful of models and never asks the local-versus-cloud question; prior local-deployment work runs on-premise for privacy reasons but never benchmarks that choice against a cloud reference, and none of the two lines decomposes extraction quality across the fields of a structured record. We address both directly, benchmarking ten models on one fixed pipeline, with confidence intervals on accessible hardware and a per-field localization of the cloud advantage.

3. Architecture

As illustrated in Figure 1, the pipeline transforms a scanner's PDF report into a set of schema-constrained structured records through a sequence of stages, from ingestion and segmentation to model inference and consolidation.

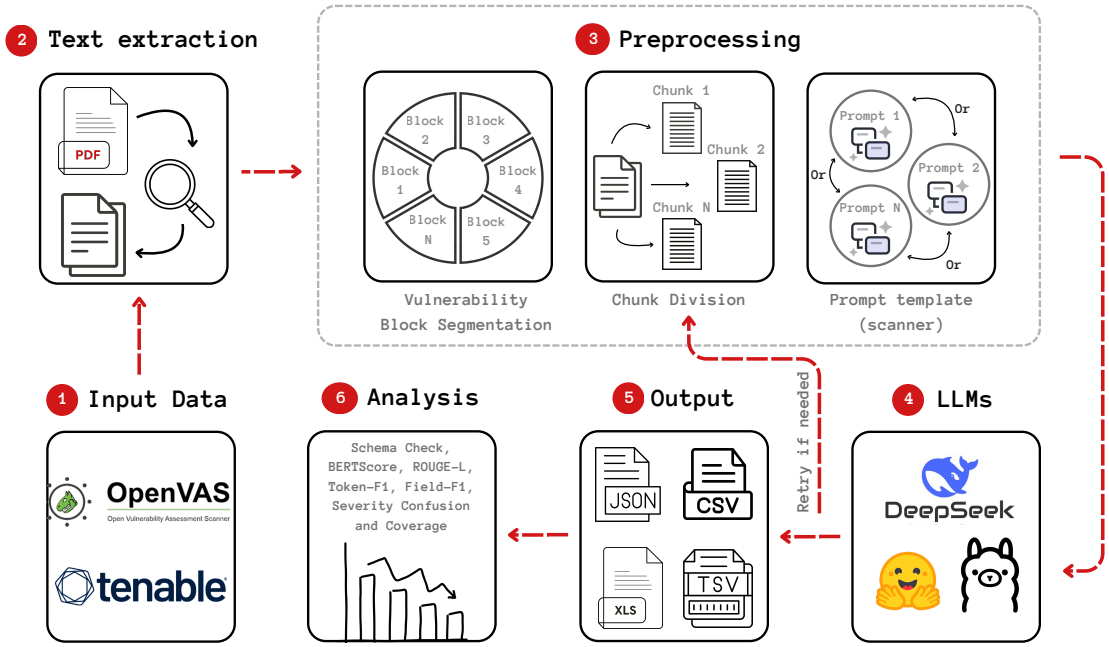

Figure 1. Overview of the structured vulnerability extraction pipeline

Extraction pipeline. We reuse, without modification, a previously proposed pipeline [Machado et al. 2026], which ingests scanner PDF reports (e.g., OpenVAS, Tenable WAS), segments each into vulnerability blocks, applies token-controlled chunking, and dispatches scanner-specific prompts requesting one schema-constrained JSON record per vulnerability, with a recovery step and an adaptive retry for malformed or truncated responses.[1] We refer the reader to [Machado et al. 2026] for the full architecture and report here only what bears on our evaluation.

[1]The schema-constrained prompts and the JSON schema are available at `https://github.com/AnonShield/MulitaMiner/tree/slms`.

Model-agnostic execution. Central to this study, the model layer is backend-agnostic: the same prompts and post-processing run unchanged whether a model is served by a cloud API or executed locally through Ollama or Hugging Face. This lets us hold the entire pipeline fixed and vary only the underlying model, so that any difference in output quality is attributable to the model rather than to the surrounding tooling. The sole per-model adjustments are the chunking and decoding parameters imposed by each model's context window and runtime.

4. Methodology

We evaluate ten models, one cloud reference and nine locally executed, on the same extraction task, comparing their outputs against manually curated ground truths.

Baselines. We use three ground-truth baselines built from intentionally vulnerable and real applications (Juice Shop, bBWA, and Artifactory), comprising 208 vulnerabilities. Curation was sequential: one annotator transcribed each vulnerability from the source PDF into the schema, and a second specialist audited every field against that same report, which settled any disagreement. The result is a single converged baseline, so no inter-annotator agreement statistic applies. Each baseline provides the reference set of vulnerabilities and their fields against which every extraction is scored.

Models. The cloud reference is DeepSeek, selected as the best-performing cloud provider in prior work [Machado et al. 2026]. We deliberately selected nine local models spanning a wide range of parameter scales (4B to 21B), capturing the size–quality trade-off relevant to constrained on-premise hardware rather than evaluating a single size. Six are general-purpose instruction models (Gemma 4 E4B, Qwen3 8B, Ministral 3 8B, Llama 3.1 8B, Phi-4 14B, and gpt-oss 21B), one is an openly available DeepSeek model (DeepSeek-Coder-V2 16B), and two are cybersecurity-domain fine-tuned variants (Foundation-Sec 8B Instruct and Primus-Merged 8B), both derived from Llama 3.1 8B. Models were executed on two consumer GPUs (an RTX 3060 with 12 GB and an RTX 5080 with 16 GB of VRAM), assigned according to their memory requirements: Gemma 4, Ministral 3, and Qwen3 ran on the 3060, and the remaining six on the 5080. The 3060 machine ran Ubuntu 24.04.3 LTS and the 5080 machine Ubuntu 26.04 LTS. All local models were served through Ollama with a 16,000-token context window, except the two Hugging Face models (Foundation-Sec, 64K; Primus-Merged, 128K, their architecture maxima) and the cloud reference ($\sim$1M); for the Hugging Face and cloud models we set no explicit context cap. Every model used a 10,000-token chunk budget, a 3,000-token output reserve, and a 2,500-token output cap. The chunk budget balances two opposing constraints: too small a value splits individual vulnerabilities across chunks, while too large a value exceeds what the smallest models in the pool can accept, so the value was set to the largest size all nine models can process, placing every model on equal footing. The sole exception is the output cap for gpt-oss, a reasoning model whose reasoning cannot be disabled, only set to a low effort level; it therefore needs a larger budget than the non-reasoning models to emit a complete answer, and we allow it 6,000 tokens.

Metrics. We score each extraction along complementary axes. *Schema Check* measures structural conformance to the canonical JSON schema; *BERTScore* measures semantic similarity of the free-text fields; *Entity-F1* reports precision, recall, and F1 over the deterministic fields (cvss, plugin, port, protocol, severity), where a field counts as

a true positive if its normalized value equals the reference; *Severity* is reported as two macro-F_1 scores over the severity classes, a matched-only score and a coverage-aware variant defined below; and *Coverage* reports exact record match together with hallucination and omission rates. We report omission at two levels, per vulnerability and per field, and decompose BERTScore by field to locate the cloud gap. We report F_{B}, computed with a `distilbert-base-uncased` backbone and baseline rescaling. Each (model, baseline) pair was run five times; we report means with 95% bootstrap confidence intervals obtained by per-vulnerability resampling. Five runs per pair is a compromise imposed by wall-clock time rather than monetary cost, since nine of the ten models run locally: the full grid amounts to 150 report extractions, each taking minutes to hours of dedicated GPU time, and runs can only be parallelized across the two machines, never within one.

BERTScore [Zhang et al. 2020] greedily matches L2-normalized contextual token embeddings $\mathbf{x}_i \in X$, $\mathbf{y}_j \in Y$, where X is the token embedding sequence of the extracted text and Y that of the reference, $\mathbf{x}_i^\top \mathbf{y}_j$ is their cosine similarity, and P_{B}, R_{B}, F_{B} are the resulting precision, recall, and harmonic mean:

$$P_{\mathrm{B}} = \frac{1}{|X|}\sum_i \max_j \mathbf{x}_i^\top \mathbf{y}_j, \quad R_{\mathrm{B}} = \frac{1}{|Y|}\sum_j \max_i \mathbf{x}_i^\top \mathbf{y}_j, \quad F_{\mathrm{B}} = \frac{2P_{\mathrm{B}}R_{\mathrm{B}}}{P_{\mathrm{B}}+R_{\mathrm{B}}}. \tag{1}$$

Equation 1 is the metric as originally defined; the remaining definitions below are introduced in this work to make coverage failures visible, since similarity alone cannot distinguish a wrong extraction from a missing one. Reported values are rescaled against the BERTScore random baseline, $F'_{\mathrm{B}} = (F_{\mathrm{B}} - b)/(1-b)$, where b is the expected score on random pairs. Schema Check is the fraction of records conforming to the canonical JSON schema. We report two severity scores. ***Sev. mF1*** is the macro-F_1 over the severity classes C, computed on matched pairs only,

$$\mathrm{Sev.mF1} = \frac{1}{|C|}\sum_{c\in C} F_1^{(c)}. \tag{2}$$

This rewards a model that omits hard findings, since unmatched records never enter the confusion matrix. ***Sev. cov*** is a *coverage-aware* variant that closes this gap: for each class c, the false-negative count is augmented with the omitted baseline records of true severity c and the false-positive count with the hallucinated extractions predicted as c,

$$\widetilde{\mathrm{FN}}_c = \mathrm{FN}_c + o_c, \qquad \widetilde{\mathrm{FP}}_c = \mathrm{FP}_c + h_c, \tag{3}$$

where o_c and h_c are the omitted and hallucinated records of class c; $\mathrm{Sev.cov}$ is the macro-F_1 recomputed from $(\mathrm{TP}_c, \widetilde{\mathrm{FP}}_c, \widetilde{\mathrm{FN}}_c)$ over the real classes. A model that silently skips findings is thus penalized rather than rewarded. Let B, E be the baseline and extracted records and $M \subseteq E \times B$ the matched pairs, with D the deterministic fields. Coverage reports

$$\mathrm{hallucination} = \frac{|E \setminus M|}{|E|}, \qquad \mathrm{omission}^{\mathrm{vuln}} = \frac{|B \setminus M|}{|B|}, \tag{4}$$

$$\mathrm{ERM} = \frac{|\{(x,b) \in M : \hat{x}_g = \hat{b}_g \ \forall g \in D\}|}{|M|}, \tag{5}$$

where $\hat{\cdot}$ is the normalized value and ERM excludes free-text and list fields. The per-field omission is conditional on presence and restricted to matched pairs (undefined when no matched pair populates f):

$$\text{omission}_f^{\text{field}} = \frac{|\{(x,b) \in M : b_f \neq \varnothing \wedge x_f = \varnothing\}|}{|\{(x,b) \in M : b_f \neq \varnothing\}|}. \tag{6}$$

5. Results

Our central finding is that *privacy-preserving, on-premise extraction is viable*: deterministic fields (CVSS, port, protocol, severity) and schema conformance are essentially solved even at 8B, and the residual gap to the cloud concentrates in the narrative free-text fields and in severity classification, not in structured extraction. We develop this across four axes: extraction quality (§5.1), operational cost (§5.2), domain specialization (§5.3), and the privacy–quality–cost trade-off (§5.4). All figures are means over five runs with 95% confidence intervals; because decoding is greedy, we attribute output quality to the model rather than the GPU, so the hardware assignment affects only the cost metrics (latency, energy, VRAM).

Table 1. Per-model extraction quality (means over 5 runs, in %). Bracketed values under BERTScore are 95% bootstrap confidence intervals (per-vulnerability resampling). The cloud reference (deepseek-v4-flash) is shown below the rule; the best *local* model in each column is in bold.

Model	BERTScore	Ent. F1	Exact	Omiss.↓	Halluc.↓	Schema	Sev. mF1	Sev. cov
Gemma 4 E4B	83.5 [81.8, 86.5]	96.2	58.4	15.7	**8.0**	64.8	97.5	87.7
Ministral 3	**93.0** [92.1, 93.9]	98.4	87.1	**2.3**	8.8	96.0	94.2	87.1
Qwen3	85.0 [83.2, 86.8]	96.8	70.0	13.5	8.9	88.1	93.9	82.1
Llama 3.1	85.5 [83.9, 87.0]	95.6	67.4	3.2	17.4	99.1	**96.6**	84.1
Phi-4	91.0 [89.8, 92.0]	**98.8**	**90.4**	**2.3**	8.6	**100.0**	95.6	88.8
DeepSeek-Coder-V2	79.4 [77.6, 81.1]	96.8	77.1	6.7	12.0	97.7	84.7	76.0
gpt-oss	91.1 [89.5, 92.0]	98.5	87.9	2.6	9.5	99.8	95.6	**89.8**
Foundation-Sec	78.8 [76.9, 80.7]	97.8	78.5	2.7	13.3	82.1	92.1	82.1
Primus-Merged	85.2 [83.9, 86.5]	96.8	76.4	6.7	12.1	99.5	88.4	79.3
deepseek-v4-flash	*94.2 [93.8, 94.5]*	*99.0*	*92.2*	*1.3*	*5.3*	*100.0*	*98.1*	*94.1*

5.1. Extraction quality

Table 1 reports per-model quality across all axes. The headline result is parity on the hard axis and saturation on the easy one: on free-text fidelity (BERTScore) the best local model, Ministral 3 8B, reaches 93.0 against the cloud's 94.2, a gap of 1.2 pp whose 95% confidence interval $[0.2, 2.1]$ excludes zero but is small in absolute terms. Two further local models cluster just behind (gpt-oss 91.1, Phi-4 91.0), after which quality drops sharply: the spread between the best and worst local model (93.0 to 78.8, roughly 14 pp) dwarfs the best-local-to-cloud gap. On the deterministic fields, every model is near-saturated (Entity F1 95.6 to 99.0), confirming that extracting CVSS, port, protocol, and plugin identifiers is not where models differ; schema conformance is likewise solved by most ($\geq 96\%$ for seven

of ten), the exceptions being Gemma 4 (64.8%), Foundation-Sec (82.1%), and Qwen3 (88.1%). Severity enters our metrics in two senses that should not be conflated: as an exact field value it is part of the near-saturated Entity F1, but as a macro-F_1 over imbalanced severity classes (Sev. mF1 and the coverage-aware Sev. cov) it still separates models. The cloud's advantage is thus confined to free-text fidelity and this severity classification, not to structured field extraction.

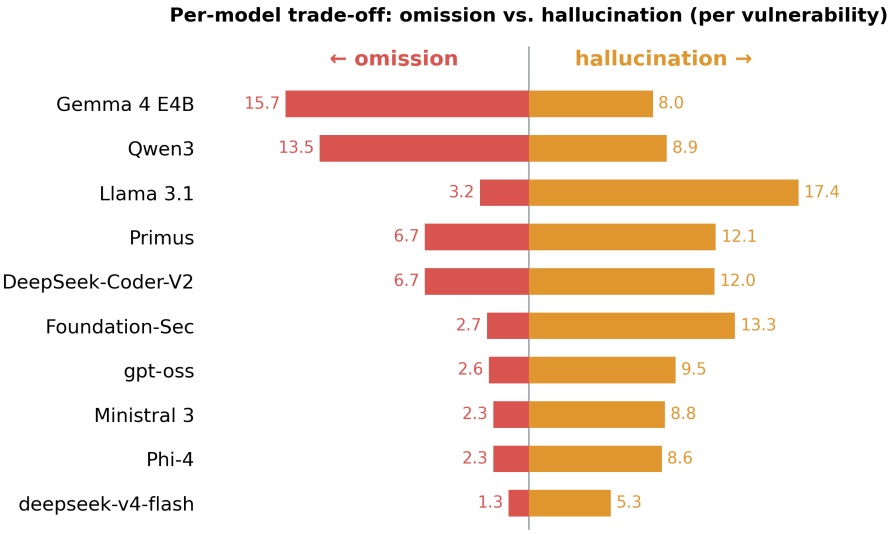

Figure 2. Per-vulnerability omission (left) versus hallucination (right).

Coverage tells a consistent story but identifies *which* way models fail. Figure 2 contrasts per-vulnerability omission (records in the baseline that the model failed to recover) against hallucination (extracted records with no baseline counterpart). The two failure modes are not symmetric: omission dominates for the weaker local models (Gemma 15.7%, Qwen3 13.5%), whereas the stronger ones keep both low (Ministral and Phi-4 at 2.3% omission, near the cloud's 1.3%). Hallucination is comparatively contained across the board, with Llama 3.1 the clear outlier (17.4%). A model can therefore be unreliable by silently dropping vulnerabilities (the larger risk in a security workflow) or, less often, by inventing them.

To locate the residual cloud gap, Figure 3 decomposes omission by field. The pattern is structural rather than model-specific: the narrative free-text fields (`log_method`, `references`, `insight`, `detection_result`) are omitted at high rates by *every* model, including the cloud reference, while the deterministic fields (port, protocol, severity, plugin) are populated almost universally. The gap to the cloud, in other words, does not lie in the structured core of the task, which all models handle, but in the long free-text fields that even a 284B cloud model fills only partially. This is the central qualification to our viability claim: on-premise extraction matches the cloud on everything except the narrative fields, where no model in our study, local or cloud, is fully reliable. Why they are harder can only be hypothesized here, since our design compares models rather than field types: unlike a port number, these fields have no canonical form and span several sentences, so there is no single short string a model can confidently identify as the answer.

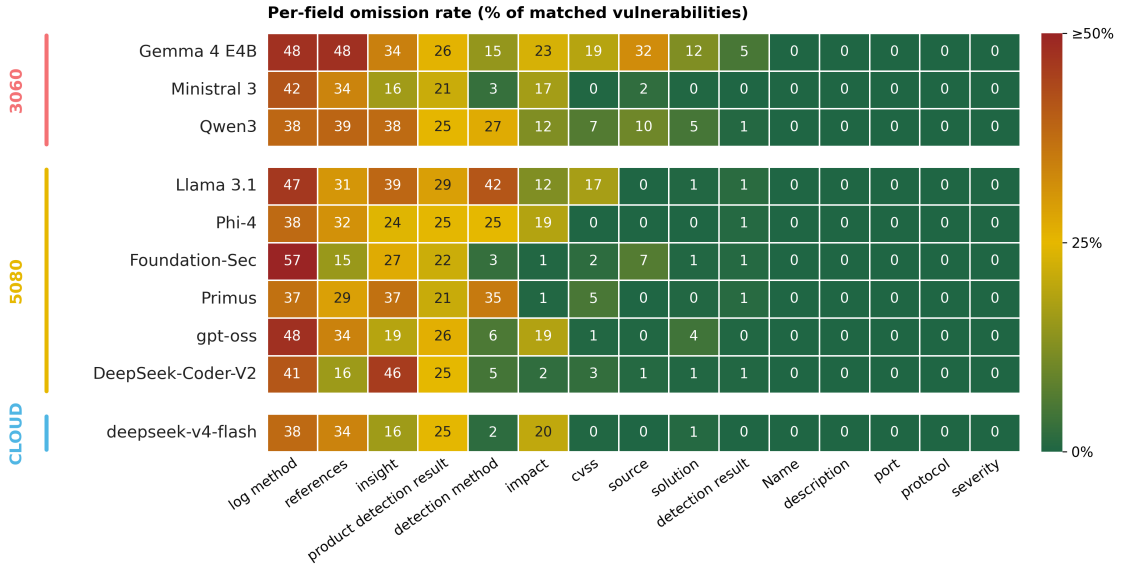

Figure 3. Field-level omission rate (matched pairs where the reference populates a field but the extraction leaves it empty), by model and field.

5.2. Operational cost

The cost structures of the two deployment modes are not commensurable, so we report each in its own terms (Table 2). On the cloud reference, billed usage is dominated by generation: input was 92% cache-covered and accounted for only ∼4% of spend, while output drove 96%. Cloud cost therefore tracks generated rather than ingested content. Over the full evaluation the cloud reference cost US$2.08 in real API usage, spread over 15 complete report extractions (three baselines × five runs). This yields US$2.08/15 ≈ US$0.14 per report, with per-report cost ranging from US$0.03 to US$0.30.

Table 2. Operational cost. Monetary cost applies only to the cloud (real API usage); compute cost applies only to local models. Latency and energy are not comparable across the two GPUs and are grouped by card.

Model	GPU	VRAM (GB)	Latency (min)	Energy (Wh)	Util. (%)
Local (per-report compute; no API fee)					
Gemma 4 E4B	RTX 3060	9.8	82	191	93
Ministral 3	RTX 3060	9.0	36	100	96
Qwen3	RTX 3060	7.5	76	215	96
Llama 3.1	RTX 5080	6.9	9	45	90
Phi-4	RTX 5080	11.9	19	109	97
DeepSeek-Coder-V2	RTX 5080	13.5	16	86	94
gpt-oss	RTX 5080	12.9	13	56	89
Foundation-Sec	RTX 5080	7.7	13	57	87
Primus-Merged	RTX 5080	7.7	17	74	88
Cloud (real API usage)					
deepseek-v4-flash	API	—	—	—	—
US$2.08 total / 15 report extractions (3 baselines × 5 runs) ≈ US$0.14 per report (output 96% of spend, input 92% cache-covered)					

The on-premise alternative carries no per-call API fee and exposes no report to a third party, shifting cost to local compute. Peak VRAM, which depends on the model rather

than the card and is therefore comparable across machines, ranged from 6.9 to 13.5 GB: the heaviest models (DeepSeek-Coder-V2 13.5 GB, gpt-oss 12.9 GB, Phi-4 11.9 GB) require the 16 GB GPU, while the rest fit a 12 GB consumer card. Latency and energy depend on the GPU and are reported per card, not compared across them. On the 12 GB RTX 3060, extraction averaged 36 to 82 min per report and 100 to 215 Wh; on the 16 GB RTX 5080, 9 to 20 min and 45 to 109 Wh. Notably, on the 3060 the strongest local model (Ministral 3) was also the cheapest to run ($\sim$36 min, 100 Wh), so the quality ranking does not trade off against operating cost. The GPU stayed saturated throughout (87 to 96% utilization), so these timings reflect model inference rather than idle waiting.

5.3. Domain specialization

Table 3 isolates security-domain fine-tuning by comparing Llama 3.1 8B against two cybersecurity fine-tunes derived from it. Both variants raise Entity F1 over the base (97.8 and 96.8 vs. 95.6), confirming that domain adaptation sharpens deterministic field identification. The effect does not generalize: Foundation-Sec lowers per-vulnerability omission (2.7 vs. 3.2) but Primus raises it (6.7), and neither improves free-text fidelity (BERTScore) over the base. Foundation-Sec in particular collapses schema conformance (82.1 vs. 99.1), trading structural discipline for marginal field gains.

Table 3. Domain specialization: the generalist base (Llama 3.1 8B) vs. two cybersecurity fine-tunes derived from it (means, %). Bold marks the best per column; ↓ lower is better.

Model	BERTScore	Ent. F1	Omiss.↓	Schema
Llama 3.1 (base)	**85.5**	95.6	3.2	99.1
Foundation-Sec	78.8	**97.8**	**2.7**	82.1
Primus-Merged	85.2	96.8	6.7	**99.5**

The marginal value of cybersecurity-specific representations is thus limited: specialization helps deterministic field identification but not free-text quality or structural conformance, and for one variant it even degrades record coverage. Reading the numbers as hypotheses rather than findings, extraction rewards fidelity to the source rather than fluent domain prose, which would explain gains on deterministic fields alongside losses on free text; and schema conformance is instruction-following rather than domain knowledge, so Foundation-Sec's collapse points to format supervision eroded by domain adaptation. Since the two variants fail differently and we do not control their training recipes, we cannot isolate the cause; the practical remedy, however, lies at decoding time, via grammar-constrained decoding or schema validation with retry. The same limit appears at the evaluator, as we show next.

Robustness to the evaluation backbone. We re-scored every matched (prediction, reference) pair with SecureBERT [Aghaei et al. 2022], a RoBERTa model pretrained on security corpora, comparing both encoders on raw (non-rescaled) scores since SecureBERT lacks a rescaling baseline. The two agree closely: per-model scores differ by at most 0.21 pp across all ten models, with no change in ranking, and 336 of 360 field cells fall within 2 pp. A domain-specific evaluation backbone therefore adds no measurable signal here, mirroring the limited gains from domain-specific extraction models.

5.4. The privacy–quality–cost trade-off

The preceding axes resolve into a single decision for a CSIRT. The cloud is the most accurate model overall, but its margin over the best local model is narrow (1.2 pp on free-text fidelity, with structured fields and schema already solved locally) and confined to narrative fields that no model fills reliably. Against that small edge, every cloud report, a map of the organization's attack surface, leaves the premises for a third party, with the attendant confidentiality and regulatory exposure; what the cloud returns is operational simplicity and a low per-report fee (US$0.14).

The local option keeps every report on-premise and removes the per-call fee, at the cost of a consumer GPU and minutes-to-hours per report. Crucially, the choice within the local option matters more than the local-versus-cloud choice: a poorly chosen model (Gemma 4, 64.8% schema; DeepSeek-Coder-V2, 79.4 BERTScore) forfeits the on-premise route, whereas Ministral 3, also the fastest and most energy-efficient on the 12 GB card, captures nearly all of the cloud's quality with none of its exposure. For a CSIRT under confidentiality or regulatory constraints (e.g., LGPD), this makes on-premise extraction not merely viable but preferable; the cloud remains attractive only where neither confidentiality nor data residency is a concern.

6. Limitations

Our evaluation rests on three OpenVAS PDF baselines (208 vulnerabilities), narrower than the scanners, formats, and sizes of a production CSIRT, so generalization is an extrapolation; in particular, although the pipeline also ingests Tenable WAS reports, all three baselines come from OpenVAS, since it is open source and freely available for ground-truth construction, whereas Tenable WAS is commercial. Latency and energy, measured on two consumer GPUs assigned by memory requirement, are not comparable across cards; we report them per card. Decoding is greedy, yet residual run-to-run variation remains (mitigated, not removed, by five runs and confidence intervals), and since each model ran on one GPU, hardware-independence of quality is an assumption. Finally, BERTScore captures semantic similarity, not factual correctness: a field may score highly while holding a wrong detail (a swapped version, port, or recommendation); the SecureBERT check rules out backbone dependence but not this limitation of similarity-based evaluation.

7. Conclusion and Future Work

We benchmarked nine local models against a DeepSeek cloud reference using three OpenVAS ground-truth baselines. The best-performing local model, deployable on a consumer-grade GPU, matches the cloud reference in structured extraction and schema conformance, while exhibiting only a 1.2 percentage-point gap in free-text fidelity. These results indicate that model selection has a greater impact than deployment mode, supporting on-premise LLMs as a practical and privacy-preserving alternative for CSIRTs operating under confidentiality and regulatory constraints.

Future work will extend the evaluation to additional vulnerability scanners, report formats, and larger-scale datasets. We also plan to investigate strategies specifically targeting narrative fields, including field-aware prompting, post-extraction validation, and factual-consistency evaluation, to further reduce the remaining gap between local and cloud-based models.

Acknowledgments

This work was partially supported by Capes, CNPq (Grant No. 409743/2025-9), FAPERGS (Grant Nos. 24/2551-0001368-7, 24/2551-0000726-1, and 25/2551-0002572-9), and FAPESP (Grant Nos. 2020/05183-0 and 2023/00816-2). The authors also acknowledge the use of Generative AI as a supporting tool for reviewing and improving the code and manuscript.

References

Aghaei, E., Niu, X., Shadid, W., and Al-Shaer, E. (2022). SecureBERT: A domain-specific language model for cybersecurity. arXiv preprint arXiv:2204.02685.

Chen, L. and Varoquaux, G. (2024). What is the role of small models in the LLM era: A survey. arXiv preprint arXiv:2409.06857.

Ferrag, M. A., Alwahedi, F., Battah, A., Cherif, B., Mechri, A., Tihanyi, N., Bisztray, T., and Debbah, M. (2025). Generative AI in cybersecurity: A comprehensive review of LLM applications and vulnerabilities. *Internet of Things and Cyber-Physical Systems*.

Keltek, M., Hu, R., Fani Sani, M., and Li, Z. (2024). Boosting cybersecurity vulnerability scanning based on LLM-supported static application security testing. *arXiv preprint arXiv:2409.15735*.

Li, G., Zhang, Y., Wang, Y., Yan, S., Wang, L., and Wei, T. (2025). PRIV-QA: Privacy-preserving question answering for cloud large language models. arXiv preprint arXiv:2502.13564.

Lu, Z., Li, X., Cai, D., Yi, R., Liu, F., Zhang, X., Lane, N. D., and Xu, M. (2024). Small language models: Survey, measurements, and insights. arXiv preprint arXiv:2409.15790.

Machado, B., Lautert, D., Kapelinski, C., and Kreutz, D. (2025). Structured extraction of vulnerabilities in openvas and tenable was reports using llms. In *Anais da XXII Escola Regional de Redes de Computadores*, pages 144–150, Porto Alegre, RS, Brasil. SBC.

Machado, B., Lautert, D., Kapelinski, C., Kreutz, D., Ferrão, I. G., and Bof, A. (2026). MulitaMiner: A multi-version evaluation of LLM-based vulnerability report extraction. In *Anais do XXVI Simpósio Brasileiro de Cibersegurança (SBSeg)*, Porto Alegre, RS, Brasil. SBC.

Rahman, F. I., Halim, S. M., Singhal, A., and Khan, L. (2024). ALERT: A framework for efficient extraction of attack techniques from cyber threat intelligence reports using active learning. In *Data and Applications Security and Privacy XXXVIII (DBSec)*. Springer.

Wiest, I. C., Ferber, D., Zhu, J., van Treeck, M., Meyer, S. K., Juglan, R., Carrero, Z. I., Paech, D., Kleesiek, J., Ebert, M. P., Truhn, D., and Kather, J. N. (2023). From text to tables: A local privacy preserving large language model for structured information retrieval from medical documents. *medRxiv*.

Yan, B., Li, K., Xu, M., Dong, Y., Zhang, Y., Ren, Z., and Cheng, X. (2024). On protecting the data privacy of large language models (LLMs): A survey. arXiv preprint arXiv:2403.05156.

Yang, L., Zhu, H., Mu, J., and Li, Q. (2026). KGAgent4CTI: Unlocking the power of LLM in threat intelligence. *Cybersecurity*, 9(76).

Zhang, T., Kishore, V., Wu, F., Weinberger, K. Q., and Artzi, Y. (2020). BERTScore: Evaluating text generation with BERT. In *International Conference on Learning Representations (ICLR)*.