

It is Not Yet Another Tool: Creating and Deploying an Agentic AI Companion in a Security Operations Center

Kritan Banstola*
University of South Florida
kbanstola@usf.edu

Ryan Irving
Cyber Florida
irvingr@usf.edu

Faayed Al Faisal*
University of South Florida
faayed@usf.edu

Daniel Lende
University of South Florida
dlende@usf.edu

Duy Dao
Cyber Florida
duydao@usf.edu

Xinming Ou
University of South Florida
xou@usf.edu

Abstract

Security Operations Centers (SOCs) process large amounts of tickets, most of which are low-interest events not worthy of further investigation. The repetitive nature of this task and similarity of the vast amounts of tickets make it a prime candidate for generative AI-based automation. We created and deployed an agentic AI companion utilizing large language models through fieldwork within a SOC for over one year. The design of the SOC AI companion was driven by researchers' participation and interactions within the SOC's daily work. SOC analysts were invited to use it during the last four months of the fieldwork. We analyzed the analysts' usage of the companion and found that in more than 90% of the cases the companion's outputs were reused by analysts in the ticket's closing report. Our results showed that when designed "in the trenches" with the intended users, a SOC AI companion can go beyond being yet another tool, but rather a system that co-evolves with its human users as it traverses through the various types of workloads. Analysts naturally started to shape the AI companion's behaviors to fit their particular needs. Our data show that the more human analysts shape the AI companion's behaviors, the more they become comfortable trusting the output from the AI system, resulting in improved productivity.

1 Introduction

A Security Operations Center (SOC) is responsible for monitoring an organization's networks and systems for evidence of compromise. Most of that responsibility is discharged through alert triage. Detection systems produce alerts continuously, and analysts work through them deciding whether an alert indicates a genuine intrusion or a benign event that happens to satisfy a detection rule. Most of the cases are benign [1]; however an analyst must still work through the alert to make that determination which involves assembling context by navigating through various tools and data sources. This process

is monotonous and laborious, and it contributes to fatigue and attrition among analysts [2, 3]. Generative AI, in particular large language models (LLMs) is potentially suited to provide automation that can alleviate this burden due to its capability to reason through varied situations presented by data. In fact analysts are already actively using LLMs as a chatbot assistant when processing tickets [4–6]. A natural next step is to create an agentic AI companion so it can take actions on analysts' behalf such as querying the various SOC tools and data sources to establish context. While there have been multiple SOC tool vendors [7–9] who claimed to have incorporated agentic AI in their solutions, there is little documentation beyond marketing materials on how these AI SOC assistants operate. Those commercial AI-enabled solutions are marketed as yet another tool/service a SOC can purchase. However there are a number of significant differences between an agentic AI system and traditional SOC tools that would create significant hurdles for agentic AI solution's adoption by SOC.

First and foremost, analysts must be able to trust an agentic AI companion's reasoning. It is well known that LLMs' reasoning could be wrong. Ultimately a human analyst must close a SOC ticket and bear the responsibility on the decisions made, regardless of whether AI provided incorrect information or not. Thus it is unlikely a SOC analyst will readily accept an AI system's output without verification. If the effort spent on verification exceeds the effort of solving the ticket manually it defeats the purpose of automation.

Second, if an agentic AI companion is programmed like yet another tool for the SOC, like all tools it is bound to fall short to fit some aspects of a specific SOC's situation. This could be peculiar data sources the SOC relies upon, the specific business and risk environment where the SOC is monitoring, subtle workflow differences dependent on organization structure, and so on. Traditional SOC solutions' one-size-fits-all approach is notorious for not providing the needed adaptability and that is one main reason behind the difficulty of automating the mundane SOC tasks that create burnout for analysts [3]. An agentic AI assistant for SOC should allow users to shape its behaviors so it can address the specific situation

*Banstola and Faisal share first authorship

a SOC faces. Such an AI assistant would be a companion to the SOC analysts that can co-evolve with the analysts and the work environment as they change, as opposed to yet another rigid, hard-to-modify tool.

In this paper we present a 14-month long fieldwork study of a SOC located in an institution of higher education. Two PhD student researchers became the SOC’s analysts and used the research method of participant observation [10, 11] to analyze the work conducted by the SOC’s analysts. Throughout the research we identified opportunities where agentic AI solutions can help automate the mundane part of the workflow, and created an AI companion to achieve that based on the researchers’ first-hand understanding of the work themselves. The SOC AI Companion is a ReAct-style LLM agent [12] that can independently run an investigation on a SOC alert. It runs on a locally deployed model, reads in each ticket and follows the same workflow the analysts are asked to follow: queries the SOC’s log sources, enriches indicators against internal and external services, and drafts a structured report for the analyst to review. All evidence is presented in the report in the same way as would by a manual analysis. We then invited the other SOC analysts to use the AI companion for the last four months of the fieldwork. Analysts interact with the companion through an interface that allows for asking questions, providing additional context, and changing system prompts. Analysts used it on real alerts as they came in and the use was voluntary. The AI companion records all transcripts and actions between users and the AI model. We also retain data in the ticketing system that show how the analysts processed the ticket in the end – did they use the AI companion’s results or not? We then carried out a systematic analysis of the collected data along with the researchers’ field notes to examine the following key aspects of creating and deploying agentic SOC AI companions in SOCs.

- **RQ1.** Would SOC analysts use the AI companion’s analysis and recommendation to close tickets?
- **RQ2.** Do analysts attempt to modify the SOC AI companion’s behavior to fit better into their work?
- **RQ3.** What improvements, or lack thereof, the SOC AI companion brought to the SOC analysts’ work?

Our main findings and contributions are:

1. We found that when the SOC companion’s output is presented in an easy-to-verify way, SOC analysts overwhelmingly used its results in ticket closing reports. It is not sufficient to have accurate results; the biggest impact on adoption appears to be verifiability. Our fieldworkers were able to observe the importance of this implicit signal and adapted the AI companion’s output, resulting in a much higher adoption rate.
2. We found that analysts naturally started attempts to change the AI companion’s behaviors to better fit into their needs and habits. With help from our fieldworkers, analysts became well versed in using personalized

system prompts that encoded their own investigation routines and preferred report layouts. The ability to adapt the AI companion also resulted in higher adoption rate.

3. We measured the time it took for analysts to close SOC tickets using the AI SOC companion, and compared that with self-reported manual analysis time. The SOC companion saved an estimated 30 – 50% of ticket-processing time for the most prolific Companion users. Analysts also reported that the time spent on closing a ticket using our AI companion is perceived to be more productive than if the ticket is handled manually.

Our findings indicate that to make a productive AI assistant for SOCs, its design can follow a different process than a traditional SOC tool. Instead of embedding AI inside yet another tool and asking analysts to orient their workflow around that system, one can design an AI companion *based on* the existing workflow in the SOC, so that it can immediately offload burdens from the analysts. Verifiability is the key to adoption; an AI companion that produces its evidence in the same format as the manual process would is much more likely to pass muster and be adopted by analysts.

2 Related Work

A number of recent works examine LLM’s use directly within SOC and incident-response contexts. Singh et al. [4] analyze 3,090 queries from 45 SOC analysts over 10 months, finding that analysts primarily use LLMs as on-demand aids for sense-making, interpreting low-level telemetry, and refining technical communication, while preserving analyst decision authority. Kramer et al. [13] study LLMs for incident-response summarization using 18 analysts and 50 real-world incidents, showing that fully autonomous summaries often omit critical details or introduce factual inaccuracies, while collaborative LLM-assisted summaries can reduce analyst effort and improve readability and consistency. Other studies using Reddit discussions [5] and interviews with 28 cybersecurity professionals across 26 organizations [6] find substantial LLM use alongside concerns about reliability, verification, security or data leakage, and organizational guidance. Our work complements these studies by going through the full circle of designing, implementing, deploying, and observing the usage of an LLM-based agentic AI assistant within an operational SOC while researchers working inside the SOC as analysts. A unique strength of our approach is the research method of participant observation, which allows us to observe the SOC’s work in zero proximity and obtain first-hand data about how analysts used and interacted with an AI companion.

Recent work has explored the use of AI and LLMs across a wide range of cybersecurity tasks, including penetration testing, phishing detection, vulnerability analysis, alert classification, and SOC triage [14–20]. What differs this work from those prior efforts is that we evaluate our approach inside a

working SOC by observing how analysts used and interacted with an agentic AI assistant.

Our work has been informed by a large body of earlier research in security operations. Recent studies document false-positive and alert-validation burden, SOC analyst burnout, broader human, organizational, tooling challenges, and importance of analysis structure [1, 2, 21–25]. Earlier field and human-centered studies show that security work depends on tacit knowledge, context, and organizational practice [26, 27]. They also emphasize that security tools should be designed and evaluated in relation to analysts’ existing activities and constraints, with practitioners involved in how those tools are developed and refined. These prior works motivate our work in evaluating AI assistance as part of everyday SOC practice rather than only as an isolated technical capability.

3 Methodology

We conducted a longitudinal, embedded field study in a SOC. The study consisted of three concurrent efforts: (1) the research method of participant observation [10, 11] within the SOC to learn, observe, and understand the SOC’s work, (2) iterative design and development of a SOC AI companion based on the observed workflow and analyst feedback, and (3) deployment and analysis of analysts’ use of the AI companion during live ticket work.

Research site The study was carried out over approximately fourteen months in the SOC of a large university. The SOC monitors institutional systems and networks, with analysts primarily performing tier 1 triage: reviewing alerts, gathering evidence from security tools and logs, determining whether activity is benign or requires escalation, and documenting the result in the ticket. Triage required analysts to combine formal procedures with local knowledge of the organization’s systems, users, and recurring alert patterns. Most of the analysts were recruited from the university’s student body and it was a highly competitive job to land. Each student analyst spent one to two years in the SOC, working 10-20 hours a week on operational tasks. Thus the SOC not only carried its operational duties but also provided an experiential learning environment for the student analysts. At the time of our fieldwork, there were between 10 to 20 student analysts in the SOC at any point, two full-time analysts, and two managers overseeing the day-to-day operations.

Research ethics and IRB Prior to carrying out the fieldwork, this research was reviewed and approved by the institutional review board (IRB) of the university. Verbal informed consent was conducted to every employee of the SOC and participation was strictly voluntary. Detailed ethical considerations are discussed in the standalone section towards the end of the paper.

3.1 Participant observation

Participant observation is an ethnographic method in which researchers learn a practice by participating in it and observing how work is actually performed in context [10, 11]. Past research has adopted this method in studying SOC [2] and we follow a similar approach in our study. Two PhD student researchers joined the SOC as tier 1 analysts. They were trained on the SOC’s tools and procedures and worked regular shifts alongside other analysts. This embedded role allowed the researchers to learn the work by doing it rather than relying solely on documentation, interviews, or post-hoc explanations. Moreover, by working tickets as regular analysts, our researchers became seen as part of the team. This is the key factor in gaining and maintaining trust which is critical for studying SOC [2].

Throughout the fieldwork, the researchers maintained field notes documenting routine triage practices, sources of delay, tool-switching behavior, uncertainty during investigations, informal analyst feedback, and potential opportunities for automation. The notes included both direct observations of SOC work and reflective memos about emerging patterns. Via collaborative analysis, these observations and reflections became the basis for the design decisions that followed. The data collected, while limited to one site, go deep into the daily practices of a practicing SOC and what works in real-time for both analysts and workflows. It thus offers an important complement to cross-site data which are more comparative but also more ethnographically thin because there is not as much verification against actual work practices such as our research approach provides.

3.2 AI companion design and deployment

Alert triage emerged through fieldwork as the most promising opportunity for intervention. Our observations showed that triage combined three characteristics: it was frequent, repetitive, and strongly shaped by local organizational practice. Analysts repeatedly gathered evidence from multiple systems, yet the final decision still depended on human judgment informed by local knowledge and organizational context. Existing SOC tools supported individual lookups but did little to reduce the repetitive work of assembling evidence *across systems*. These observations led us to focus the Companion on evidence gathering and report drafting while leaving the final decision under analyst control.

The initial version of the Companion was first used by the two embedded researchers during their own triage work. After several design iterations and when the researchers were comfortable relying on its output to close tickets, the Companion was introduced to other SOC analysts who already gained good SOC experience (as determined by the SOC managers) for the final four months of the study. Use was voluntary: analysts decided whether to use the Companion

on a ticket, whether to rely on its output, and whether to continue the investigation manually. Before deployment, analysts were briefed on the Companion’s capabilities, limitations, and available data sources, and were told to treat its output as decision support rather than authoritative judgment. Throughout deployment, the researchers remained available to answer questions, observe use, and collect feedback.

3.3 Data collection and analysis

As part of ensuring the trust we earned within the SOC, our embedded research approach aimed to generate minimal disruption for analysts. Controlled experiments were not feasible. Instead, data collected from our fieldwork is observational in nature and reflect analysts’ authentic interactions with the Companion in solving real SOC tasks.

We collected three sources of data: 1) field notes documenting researchers’ observations of SOC work, interactions with analysts, and analyst feedback from interviews and informal comments; 2) Companion logs containing prompts, AI model responses, tool activities, evidence, prompt versions, and timestamps; and 3) ticketing-system records containing analysts’ final reports. We analyzed these data using complementary qualitative and quantitative approaches. Section 5 presents the detailed analysis and findings.

4 SOC AI Companion Design Evolution

4.1 Design principles

Working the alert queue like the other analysts allowed our researchers to reflect upon the workflow and this reflection made the need for AI Companion’s features apparent. The Companion was not to remove the analysts from triage, but to reduce repetitive evidence-gathering and reporting labor while allowing analysts to inspect, redirect, or revise the Companion’s work. Processing a typical alert required moving through a number of internal and external systems to gather relevant information. Each system has its own interface, authentication, query conventions, and output format, and none is aware of the others; the analyst was the integration layer: much of a shift goes to copying a value out of one browser tab, pasting it into the next, and waiting, while the judgment that goes into closing the ticket – whether this is a real intrusion or another benign event that tripped a rule – is a small fraction of the elapsed time. That asymmetry is what we targeted: not automating the decision but the assembly of the context on which it rests. Since analysts remained responsible for their decision in closing tickets, the Companion needed to present evidence in a form that matches what the analyst would produce manually. This not only eased adoption but also allowed for verification by analysts. Our fieldwork also showed that analysts differed in their preferred level of detail, report structure, and investigative habits. A useful AI assistant

therefore needed to be adaptable rather than behave as a fixed, one-size-fits-all tool.

4.2 Workflow automation

The key part of the Companion’s work is to gather and assemble relevant information from four kinds of systems: 1) several OSINT (Open-Source Intelligence) and threat intelligence services to establish whether the external indicator was malicious; 2) the SIEM, to find which internal host had communicated with it around the alert timestamp; 3) an internally maintained DHCP lease lookup tool, to resolve the address the SIEM returns to a MAC address and hostname; and 4) the device identification portal, to attribute the device to a person. This is a sequence of dependent lookups, where the next query follows from what the last one returned, which is the pattern that “tool-using agents” [12, 28] are suited to. We therefore built the Companion as a ReAct-style agent [12] using LangChain [29], alternating between reasoning about what it still needs and calling a tool to obtain it, until it can write up the case. We ran it on locally hosted models, whose reach is limited by the tool layer: it cannot compose actions of its own but only calls from the fixed set of functions pre-configured in the system, with schema-validated arguments. All the tool calls are read-only – retrieving a ticket, running a hunting query, looking up a reputation record, and so on – and cannot modify any state in the SOC’s systems. A run that goes wrong therefore costs a wasted investigation, not a change to the systems the SOC depends on. Most importantly, no sensitive information such as internal addressing, hostnames, or the identity of a community member leaves the organization’s network.

4.3 Empowering analysts to adapt the AI

Analysts who close the same alert type reach the same conclusion by different routes: some enrich every external indicator before touching internal logs; others identify the host first and drop the enrichment once the traffic admits a benign explanation, such as a routine update service. Their write-ups differ too, in ordering and in how much raw log data and ruled-out indicators they retain. An AI agent with a fixed procedure would have suited some and irritated the others; so we made the Companion’s behavior itself something analysts could direct. At this time the primary adaptation mechanism is through the system prompt, which describes the workflow, the order in which tools are tried, the conditions for abandoning a line of inquiry, and the report layout. Analysts kept several named versions of system prompts and switched between them per investigation, with a shared base version as the default for anyone who has not written their own. By adapting system prompts the analysts can influence how the Companion works by themselves without relying on the researchers or redeployment.

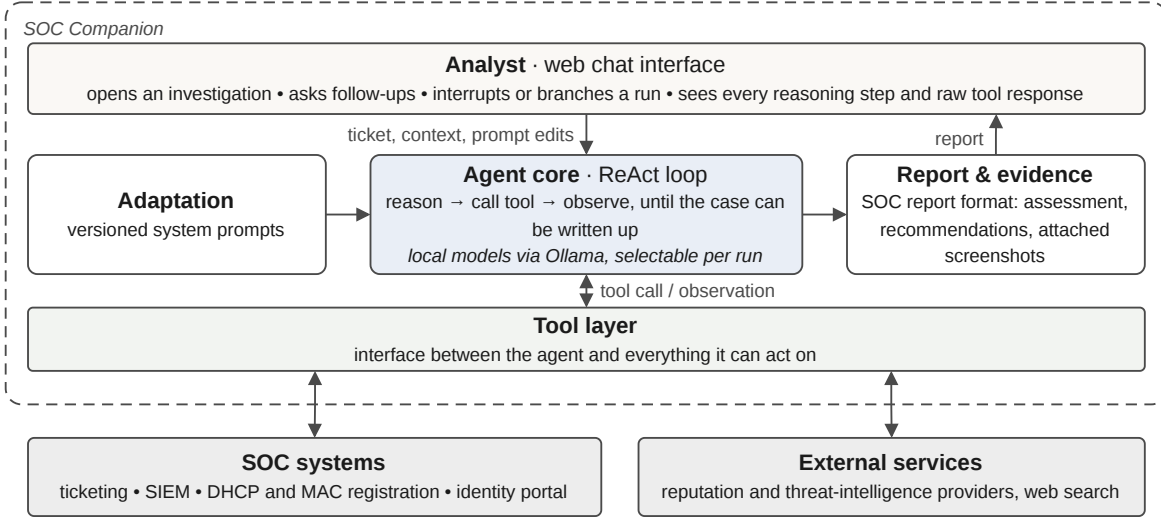


Figure 1: System Architecture of SOC AI Companion

4.4 Deployment and refinement

We deployed the Companion on an NVIDIA DGX Spark [30] inside the SOC and gave accounts to a small group of experienced analysts. Their feedback led to two refinements during the deployment. The first extended the tool layer to capture visual evidence: whenever a tool performs a lookup, a headless browser driven by Playwright [31] also visits the corresponding page, renders the HTML the service returns, and captures a screenshot of the result. The second refinement added a revised version of the shared default system prompt, which changed only the report layout based on analysts’ feedback without changing the reasoning or tool usage. Section 5 reports the impact of both refinements.

4.5 Final design

The design that emerged from this process has four parts, shown in Figure 1.

Agent core. A ReAct-style agent loop drives each investigation. The analyst opens an investigation by supplying a ticket number or pasting an alert, and the agent retrieves the full ticket (Figure 9 in Appendix A shows one), then iterates between reasoning and tool calls until it can produce a report. Runs execute on the server rather than in the browser, so an investigation survives a page refresh or a closed laptop, and a run that exceeds its time budget is terminated and reported as failed rather than left hanging.

Tool layer. The tools available to the model fall into the categories shown in Table 1, which together cover the systems an analyst navigates manually during triage.

Adaptation layer. Editable and versioned system prompts allow the Companion’s behavior to be changed by its users

Category	Systems reached
Ticketing	Alert ticket retrieval
Log sources	SIEM advanced hunting queries, log search platform
Internal asset and identity	DHCP lease lookup, MAC registration, device and owner records
External enrichment	Reputation and threat intelligence providers, general web search
Analysis utilities	Encoding and decoding, detection signature lookup

Table 1: Categories of tools available to the Companion. Each category corresponds to systems the analysts already use manually during triage.

without our involvement.

Report and evidence layer. The agent’s output is a structured report following the SOC’s own reporting conventions, with an assessment, recommendations, and a supporting evidence section carrying the screenshots captured during the investigation.

Analysts work with the Companion through a web-based interface. After a ticket is imported a series of actions happen automatically based on the AI reasoning. Every reasoning step, tool invocation, and raw tool response is streamed to the interface as the run proceeds and retained afterwards, so an analyst can see which query produced a given claim. In the end the Companion generates a structured triage report in a similar form as what analysts would produce themselves. Analysts inspect the report; they can provide additional context, request further investigation, ask follow-up questions about the case, branch a conversation to explore an alternative reading of the evidence, or decide to modify system prompts and other

settings and rerun the investigation. The interface also records everything we later analyze in Section 5: transcripts, tool calls, prompt edits, and the timing of each run. Inference runs on models served locally on the DGX Spark with Ollama [32], and the model used for an investigation is selectable per run, which let us compare locally hosted models of different sizes during the deployment without changing anything else about the system. We used GLM-4.7-Flash (30B) [33] for most of the deployment, since it called tools reliably and was small enough to run alongside the rest of the system on a single node. Qwen3.6 (35B) [34] and gpt-oss (20B) [35] were available too, and an analyst could switch to either of them for a given investigation.

5 Data Analysis and Findings

An analyst closes a SOC ticket by writing a report and attaching it to the ticket. Whatever the AI companion produces is therefore useful only insofar as it reaches that report, and this is a question our deployment can answer directly. The Companion retains every thread, message, tool call, and raw tool response, along with the system prompt and model that produced each run. The ticketing system retains what the analyst filed at the end of it. Setting the two records side by side shows what the Companion offered and what the analyst was willing to put their name to, for the same ticket.

5.1 Corpus and coding

Our data include all tickets that SOC analysts, excluding the embedded researchers, ran through the Companion during the last four months in the fieldwork starting on April 29, 2026. For coding and more in-depth analysis, we cut-off on the ticket date of July 31, 2026 with a total number of 108 tickets. There were 10 analysts working in the SOC during that time period and we only made the Companion available to analysts who were determined by the SOC managers as more experienced, to reduce the likelihood that analysts miss incorrect information in the Companion’s output. These analysts had all worked in the SOC for at least six months and many had already worked there for a year. Six analysts (referred to as A1–A6) appeared in the corpus. A2 contributed 42 tickets, A3 contributed 39, A5 contributed 19, A4 contributed 4, A6 contributed 3, and A1 contributed 1. Not all six analysts were present for the entire deployment period. A1 left the SOC a week after the Companion became available to them, A4 left partway through the period, and A6 received access to the Companion only in its final two weeks. A2, A3, and A5 were present for most of the deployment. 94 tickets were worked in a single Companion thread; 14 were worked across two or more.

Our data corpus aims at capturing whether and how analysts used the Companion’s output in closing the tickets. To

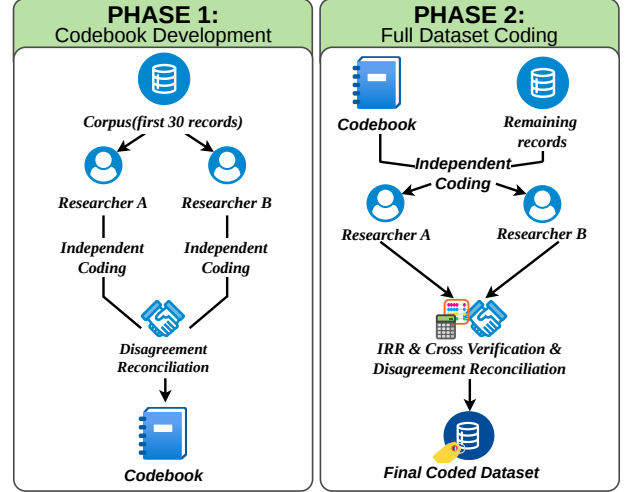


Figure 2: Coding Process.

this end we associate each ticket with two records: the Companion transcript and the final ticketing-system report. The transcript included messages, tool calls, raw tool responses, generated screenshots, prompt versions, model selections, and timestamps. The ticketing-system report contained the text and images the analyst ultimately filed and signed their name to. We coded each pair using three sets of codes. Usage codes (U1–U14) describe what happened to the Companion’s draft, including whether its text or screenshots reached the final report and what was added, removed, or reversed. Intervention codes (I1–I11) capture analyst messages sent after the opening request that started the run. Other codes (O1–O5) capture reruns, prompt or model changes, and grounding failures such as fabricated assertions or invalid tool calls. Table 2 lists all codes and frequencies. We also recorded the verdict each side reached, grouping both the Companion’s draft verdict and the analyst’s final verdict into three classes: *benign*, *uncertain*, and *malicious*.

The two researchers first independently coded an initial 30 tickets’ records, compared their assignments, and reconciled disagreements to refine and finalize the codebook. Using the resulting codebook, both researchers then independently coded the remaining 78 tickets’ records. Figure 2 illustrates this process. We assessed inter-rater reliability on these independent assignments using Cohen’s κ [36]. Across 2,340 binary coding decisions, the researchers agreed on 2,322, yielding 99.23% observed agreement and an overall Cohen’s $\kappa = 0.965$. Following the reliability assessment, the researchers reconciled the remaining disagreements to produce the final coded dataset.

Alongside the codes, we compute *draft reuse*: a text-only similarity score between the Companion’s draft report and the analyst’s final closing report. For each ticket, on the Companion side we take the draft report it produced in the primary thread; on the analyst’s side we take the closing report

	Usage codes	<i>n</i>
U1	Text used verbatim	35
U2	Text partially used	63
U3	Text not used	10
U4	Analyst added text	26
U5	Text truncated	41
U6	All screenshots reused	78
U7	Some screenshots reused	17
U8	No screenshots reused	10
U9	Analyst added own screenshots	51
U10	Recommendation overridden to no-action	34
U11	AI evidence gap filled by hand	12
U12	AI verdict uncertainty suppressed	26
U13	AI template artifact cleaned	5
U14	AI verdict escalated	2
	Intervention codes	
I2	Evidence demand	6
I6	Output shaping	5
I1	Missed step	4
I5	Provide data	4
I3	Error diagnosis	3
I4	Provide context	3
I8	Widen search	2
I7	Verdict steering	1
I9	Nudge	1
I10	Off-task	1
I11	Ask for explanation	1
	Other codes	
O1	Ticket rerun in a new thread	14
O2	Different system prompt tried	9
O3	Different model tried	1
O4	AI hallucination or fabrication	1
O5	Invalid tool invocation	2

Table 2: Codebook and code frequencies over the 108 coded tickets. Except for U1–U3 and U6–U8, which are each mutually exclusive, codes may co-occur; a ticket carries as many as apply. Definitions are given in Appendix B.

they filed on the corresponding ticket. We normalize both by removing formatting markers, collapsing whitespace, and folding case, then score the pair using RapidFuzz’s token set ratio [37], a word-order-insensitive variant of edit similarity [38]. The comparison is set-based: the two texts are compared as bags of words, making the score insensitive to reordering and to reflowed formatting.

Tickets coded U1 (text used verbatim) had a median reuse score of 99.6% (IQR 98.8–100.0), whereas tickets coded U3 (text not used) had a median of 51.3% (IQR 49.0–74.2). A U1 case does not always have 100% reuse score because the two documents may differ in non-substantive ways; for example the closing report leaves out the Companion’s opening line (“Now I have all the necessary information. Let me generate the final report.”) and carries captions for the attached screenshots (“DHCP Lease”, “Recorded Future IP”). On the other hand, a closing report the analyst wrote from scratch

still has roughly 50% reuse score rather than zero, because it still talks about the same incident and thus shares many common elements — addresses, hostnames, and signatures with the Companion report.

Some properties of the corpus limit the conclusions one can draw. First, analysts used the AI Companion and were influenced by its reasoning. Thus they were not independent evaluators for accuracy. This research focused on how the Companion was utilized; next steps could include more independent assessments of the Companion’s accuracy (e.g., via hiring independent analysts to redo some tickets). However, even outside assessments won’t necessarily capture analysts’ judgment on whether/how to use the Companion — these can be best assessed through what happens in practice.

Second, use of the Companion was voluntary and analysts chose which alerts to bring to it; thus the corpus is not a random sample of the alert queue. The analysts are unevenly represented, so we make no claim resting on A1, A4 or A6 alone. The Companion changed during the deployment — its system prompt was revised and its model swapped — so the corpus is not homogeneous over time. In this case, the combination of participant observation and design iteration is precisely what can capture analysts’ authentic behaviors while using the Companion in the SOC’s real tasks.

5.2 How much of the Companion’s work reached the ticket closing report

Across the corpus the median draft reuse is 96.7%, the mean 85.5%, the lower quartile 72.0%, and the minimum 27.5%. The median alone would suggest that analysts accept drafts almost entirely. But the average differs by eleven points. Figure 3 shows why: the distribution is highly concentrated towards high reuse rate (close to 100%), with a second, smaller concentration of low-reuse tickets (around 50%). Recall that a 50% reuse score indicates the analyst rewrote the report from scratch. The median sits completely inside the high-reuse group (90-100%), which holds 69 of the 108 tickets, and the mean is dragged by the lower reuse cases. Of the 108 tickets, 59 (55% of them) have at least 95% draft reuse and 28 (26% of them) have less than 75% draft reuse; the remaining 21 fall in between. The codes tell the same story: 35 tickets were closed with the draft text reused verbatim (U1) and 10 were written from scratch (U3). The 63 partial-use tickets (U2) lean towards the two ends — 25 at or above 95% and 21 below 75% — rather than filling the middle.

Analysts, in other words, do not edit drafts often. They decide whether to take one and that decision is close to binary. When a draft is taken it is often taken whole, and when it is rejected the analyst writes the closing report from scratch rather than taking parts of it that were sound. Table 3 breaks down the partially-used Companion reports (code U2) by how they were edited. The most common edit is not to the analysis at all. In 20 tickets the analyst pasted the draft essentially

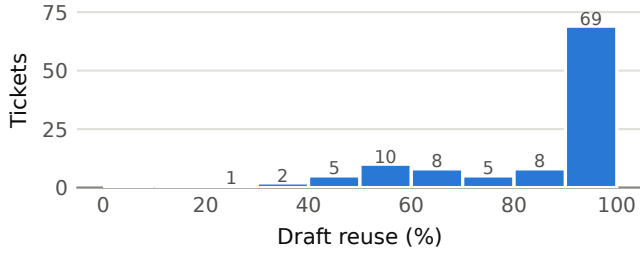


Figure 3: Share of the Companion’s draft surviving into the analyst’s closing report, over 108 tickets. Drafts are taken nearly whole or discarded, with little in between: 59 tickets at or above 95% reuse, 28 below 75%. Median 96.7%, mean 85.5%, lower quartile 72.0%.

How the AI draft was edited	<i>n</i>	Mean reuse score	Median reuse score
Recommendation dropped	20	92.2%	97.9%
Trimmed	8	92.8%	96.2%
Verdict softened	8	83.2%	93.8%
Condensed	7	68.3%	64.6%
Rewritten	7	61.5%	59.7%
Reframed	6	68.5%	67.1%
Kept and extended	4	61.5%, 82.3%, 85.8%, 90.7%	
Cleaned paste	3	91.5%, 97.4%, 99.6%	
Total	63	82.2%	91.5%

Table 3: Partially-used Companion drafts (U2) by type of edit, with mean and median draft reuse. The most common edit leaves the analysis intact and removes only the recommended action. For the two categories with $n < 5$ we list the individual ticket’s draft reuse values rather than an average.

whole — median reuse 97.9%, mean 92.2% — and changed only its recommendation. We return to those in Section 5.4.

5.3 Verifiability is key to adoption

Analysts told us they would always attach the screenshots from the various information sources (Virus Total, SIEM, etc.) in their report, while the initial version of the Companion presented API query results from those tools in a textual format. Even though the textual content was correct information, analysts were not willing to accept them. One analyst even insisted on verifying the results manually by querying each tool themselves. This certainly defeats the purpose of reducing analysts’ burden. We realized that for analysts to adopt the AI Companion, it is not sufficient to solely ensure all information presented by the Companion is accurate; it must also show the accuracy in a self-evident way. Incorporating visual rendering of the result from the external tool serves that purpose — it becomes clear that the evidence was not made up, but from a trusted source. In fact this is the exact reason why analysts included screenshots in their own reports.

	<i>n</i>	Mean reuse	Median reuse	$\geq 95\%$
Before	3	66.0%	50.0%	1 (33%)
After	105	86.1%	96.7%	58 (55%)

Table 4: Comparison before and after screenshots were attached to the Companion’s draft reports.

We then extended the Companion’s tool layer to render and capture the page behind each tool lookup and attached the “screenshots” to the draft, as described in Section 4.4.

The first ticket carrying attached screenshots was on May 28, 2026. There were only three tickets predating that despite the Companion having been made available to analysts for more than four weeks at that point. After the screenshot feature was introduced, the researchers noticed an obvious change of analysts’ attitude towards the Companion — they became much more willing to use it. In fact this is the moment when we saw the largest shift in adoption during the deployment. Draft reuse reflects this change: across the three tickets before the change the median reuse was 50.0% whereas across the 105 tickets after it the median was 96.7%. The mean rises more modestly, from 66.0% to 86.1%. Because only three tickets predated the change, we treat the before/after reuse comparison as descriptive rather than a controlled baseline. Table 4 illustrates this statistics.

The usage codes provide another view on how analysts treated the visual evidence. Of the 105 tickets after the change, 78 (74%) carry U6 (all screenshots reused), and a further 17 (16%) carry U7 (some screenshots reused) — 90% of tickets kept at least some of the screenshots the Companion captured. This indicates the importance of the screenshots to the analysts.

This experience seems to indicate that adoption depended not only on whether the Companion’s analysis was accurate, but also on whether its output was quickly verifiable. By *verifiability* we mean whether an analyst could confirm the information by themselves without redoing the work. A model-written sentence that an indicator was flagged by a reputation service is still a claim the analyst must check. A screenshot of that reputation-service page, captured by the tool layer from the authoritative source, gives the analyst evidence they can inspect and attach to the ticket. It is also what a reviewer of the report expects to find there. The difference is where the two come from. Text is written by the model, and a model can make up text so an analyst cannot tell a true result from API call from a hallucinated one without calling the tool at all. A screenshot is produced by the tool layer and the content comes from the authoritative source’s page. The model writes the text, but it cannot generate the image itself.¹ The Companion’s usefulness increased after the change even though its substantive work stayed the same.

¹The model used was a text-only LLM.



Figure 4: Three levels at which the Companion’s output could be adopted: content, verdict, and recommendation. Each bar is the share of the same 108 tickets on which that layer survived into the closing report.

5.4 Three layers of adoption: content, verdict, and recommendation

Draft reuse measures whether the Companion’s prose reached the ticket. It does not measure whether its judgment did, and the two diverge sharply.

We put three questions to the same corpus, each answered as a share of the same 108 tickets. Did the analyst keep the Companion’s content? On 98 tickets, or 91% of all, the closing report reused the draft text whole (U1) or in part (U2). Did the analyst agree with the verdict the Companion reached? In 72 of 108 cases, or 67% of all, the answer is yes. Did the analyst adopt the recommendation the Companion produced? Every draft ends by recommending what to do about the alert, which may be to do nothing. In 74 of them, or 69% of all, the analyst kept the recommendation made by the Companion (some wording may have been shortened/condensed). On 34 tickets — roughly a third of the corpus — the analyst deleted the action the Companion had proposed and filed the report recommending no action (U10). We saw no case where the analyst added or escalated an action the Companion had not recommended. Figure 4 puts the three rates side by side. The first rate stands well apart from the other two. The Companion’s content reached the closing report on 91% of tickets, but its verdict did so on 67% and its recommended action on 69% — the content reaches the closing report much more often than the judgment does.

The gap between the first answer and the third is telling. Analysts who struck the recommendation had reused nearly as much of the draft as analysts who kept it, on both statistics (Table 5): a median of 94.2% against 97.2%, a mean of 83.4% against 86.5%. Both groups sit high in the reuse distribution, and the gap between them — within three percentage points of median and mean — is far smaller than the gap between a kept draft and a rewritten one. Striking down the Companion’s recommendation is therefore not telling us the analyst distrusted the draft and rewrote it. Instead the analyst kept the Companion’s investigation, its analysis and evidence, filed all of it, and cut the final sentences that recommended some actions. Keeping the content and accepting the judgment are separate acts, and draft reuse score only reflects the first.

Companion’s recommendation was	<i>n</i>	Median reuse	Mean reuse
struck down (U10)	34	94.2%	83.4%
kept	74	97.2%	86.5%

Table 5: Draft reuse by whether the analyst kept or struck down the Companion’s recommended action.

Companion verdict	Recommendation struck down (U10)	Recommendation kept	% U10
Benign	0	38	0%
Uncertain	24	16	60%
Malicious	10	20	33%

Table 6: Tickets where analysts struck down/kept recommendation by Companion verdict.

Where analysts struck the recommendation is just as revealing. Table 6 splits the tickets by the verdict the Companion had reached and compares recommendation adoption. Analysts never struck recommendation from a benign finding: a benign finding recommends nothing, so there is nothing to strike. But they struck recommendations from 24 of its 40 uncertain findings and 10 of its 30 malicious findings. For the malicious U10 cases the analysts also filed a lower verdict (uncertain or benign) than the Companion had drafted, except in two cases. In these two cases the recommendation was struck because the action had already been taken — the user was already disabled, or the traffic was already dropped. Hedging costs the Companion its recommendation — where it declined to commit, analysts kept its work and made the call themselves. Code U12 (Table 2) points the same way: on 26 tickets, roughly a quarter of the corpus, the analyst deleted the Companion’s stated uncertainty before filing. These findings indicate that the AI companion still provided valuable utility to the analysts even when it could not reach as decisive a decision as the analyst.

Figure 5 lays out the full verdict comparison; it is important to note that this is not a confusion matrix despite displayed in a similar form. As can be seen in the discussion that follows, verdicts at times reflect more choices of judgment than absolute ground truth. The Companion hedges more than analysts: of the 40 alerts it called uncertain, analysts closed 26 as benign and left only 12 uncertain. Of the 26 benign cases, 15 were the same signature: a default Cloudflare tunnel domain pattern in the TLS SNI². Each alert looks anomalous on its own, so the Companion declined to commit. But this is a university network where students run their own devices, and a student standing up a tunnel is ordinary. The fact that settles the alert is a base rate that lives in the analyst’s head, not in any tool the Companion can call. Hedging here is a

²Server Name Indication, the TLS handshake extension that carries the destination hostname in cleartext, making it visible to network monitoring even for encrypted connections.

		Companion verdict		
		Benign	Uncertain	Malicious
Analyst verdict	Benign	38	26	7
	Uncertain	0	12	1
	Malicious	0	2	22

Figure 5: The verdict analysts closed tickets as (rows) vs. the Companion’s draft opinions (columns), over the 108 tickets. The Companion over-hedges: most of what it called uncertain was closed benign.

missing prior, not a reasoning failure.

Conflicting verdicts, not hedging, is what costs the Companion its draft. When the two verdicts conflict outright, median reuse falls to 67.9% and mean reuse to 67.2%, against 97.4% and 87.5% when the verdicts match (Table 7). When only one side declined to commit (uncertain), reuse barely changes — a median of 96.4% and a mean of 85.1%.

For seven tickets the AI companion and analysts disagree outright on the verdict (malicious vs. benign). In all those cases the Companion drafted malicious and the analyst closed the ticket as benign. Out of the seven only one was a clear Companion error. The alert reported that a Windows host made a TLS request to a known benign public IP-lookup service; a signature fired on that pattern because the malware infostealers uses such services to report a victim’s address. The alert’s reference block linked a VirusTotal page for one such stealer’s hash value as documentation. Nothing tied that hash to this Windows host. The Companion draft also stated that no matching process events were found on the host either. Despite this, the Companion still reported the malware sample as though it were found on this host, naming the malware and writing that the traffic matched its behavior. It even looked up the hash in VirusTotal regardless. This is a clear case of the Companion’s confusion in understanding the SOC ticket. Besides this one case, the other six are judgment calls that could be decided either way. Five of them hinged on a single question — how much a clean IP reputation should count against a malicious signature? Four of those cases involved inbound traffic that was already dropped by the firewall due to payload matching C&C signatures. The source sat in well known cloud provider space (AWS, Google, Azure). The Companion’s draft reported the source as unflagged but decided malicious anyway. The analyst read the same lines the other way: since the traffic was already dropped they closed the tickets as benign.

The transcripts record failures the verdict comparison could not show. On 12 tickets the analyst supplied device, user, or payload attribution by hand because the Companion’s tools

Verdict relationship	<i>n</i>	Median reuse	Mean reuse
Verdicts match	72	97.4%	87.5%
One side hedged	29	96.4%	85.1%
Verdicts conflict	7	67.9%	67.2%

Table 7: Draft reuse by how the analyst’s verdict compares to the Companion’s.

had come back empty (U11). On five it leaked its own prompt scaffolding into the draft, and the analyst stripped it before filing (U13). On three its own output was not grounded in its context. Twice it made an invalid tool call (O5) — an IP passed to a reputation tool that takes a domain, and a signature-lookup tool called under a name it did not have — and both failed visibly at the tool layer. The third was silent: on a ticket whose alert was itself a tunnel-domain-pattern signature, the Companion named a specific subdomain matching that pattern as a suspicious indicator, one that appeared neither in the alert nor in any tool result (O4). It was fabricated in the shape the ticket had primed, and nothing in the draft set it apart from an indicator a lookup had returned; the analyst caught it only by asking where the domain had come from.

Those two classes differ in visibility. When an invalid tool call occurs (O5), the failure is explicit at the tool layer. A fabrication (O4), by contrast, fails silently: it looks no different from a valid result in the draft report. That this one was caught shows that analysts did not blindly trust the Companion’s results and maintained vigilance while using the AI assistance.

5.5 How analysts steered the Companion

We expected the transcripts to show analysts correcting the Companion in conversation, and they largely did not. We count as an intervention any message the analyst sent to the Companion after the opening request. Only 17 of the 108 tickets carry an intervention code at all, 31 such codes in total across the corpus. On most tickets the analyst started the investigation, waited, and then either used the result or did not. The intervention codes are correspondingly thin (Table 2): the two most frequent are demanding raw evidence (I2, six times) and asking for a different format or length (I6, five times), and four of the eleven codes occurred only once.

On tickets where the analyst intervened, median reuse is 92.1% against 97.3% where they did not (Table 8, top), and the gap is small at the median but wider at the mean (79.4% against 86.7%); intervened tickets’ Companion drafts are not typically much worse, but they do carry more of the rewritten-from-scratch cases. The U3 code puts a number on that: the draft text was written off entirely on 24% of intervened tickets against 7% elsewhere.

Besides intervention, the other thing analysts could do if they are not happy with what the Companion offered is to start

	<i>n</i>	Draft text used (U1, U2)	Draft text not used	Median reuse	Mean reuse
<i>Intervened</i>					
Yes	17	76%	24%	92.1%	79.4%
No	91	93%	7%	97.3%	86.7%
<i>Rerun in a new thread</i>					
Yes	14	93%	7%	91.2%	79.4%
No	94	90%	10%	97.0%	86.4%

Table 8: Whether the Companion’s draft text reached the closing report, by intervention and by rerun.

the ticket over in a new thread. We count that separately from intervening: a rerun is a new thread, not a message inside one. Fourteen tickets were rerun from the start (O1), and on most of them the analyst changed the run’s configuration: nine were worked under a different system prompt (O2) and one under a different model (O3). Median reuse is 91.2% on tickets rerun in a new thread against 97.0% on the rest (Table 8, bottom), and the mean again falls further (79.4% against 86.4%) — but no more drafts are written off: 7% of reruns against 10% elsewhere. Intervened tickets were written off far more often than the rest of the corpus; rerun tickets were not. Both groups end up with less of the draft text in the final report, but after an intervention the analyst more likely tend to discard the Companion draft text outright.

The system prompt did not stay fixed over the deployment. Initially, the Companion ran on a shared default system prompt, and analysts could customize the system prompt from the interface, without us and without a redeployment (Section 4.3). Midway through the deployment we also rewrote the shared default system prompt, in response to what analysts told us they wanted to read: the revision changed only the layout of the report draft, and left the workflow, the order in which tools were tried, and the tool set as they were. The revised default did not replace the original: it was added as another named version, and the original default stayed available and in use, so tickets ran under it after the revision as well. Tickets therefore fall into three groups by the system prompt their investigation ran under — the original default, our revised default, and a system prompt the analyst wrote themselves. The three groups differ in how much of the draft survived.

Draft reuse is higher when the analyst wrote their own personalized prompt (Figure 7). The 55 tickets worked under an analyst-written system prompt show a median reuse of 98.2% against 69.8% on the 38 tickets worked under the original default (means 91.1% and 76.1%). The 15 tickets worked under the revised default prompt sit between the two, at a median of 94.2% (mean 89.0%); being neither analyst-written nor the original default, they are shown separately and kept out of that two-way comparison. Both groups point the same way. More drafts survived into tickets closing reports when they were produced under a system prompt someone

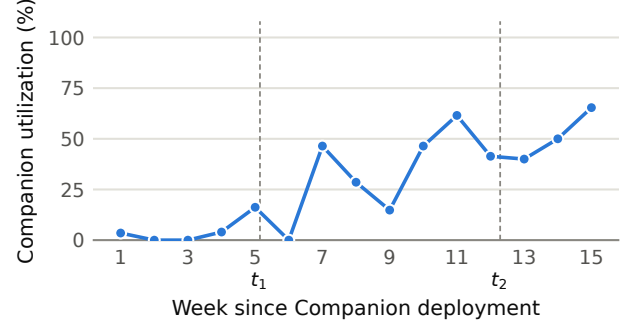


Figure 6: Weekly share of SOC tickets that were worked on the Companion, from the first Companion ticket close (April 29, 2026) through week 15. The dashed lines mark the two changes we made to the Companion during the deployment: at t_1 (May 28, 2026) the screenshots from tool query was added; at t_2 (July 17, 2026) a revised system prompt was added.

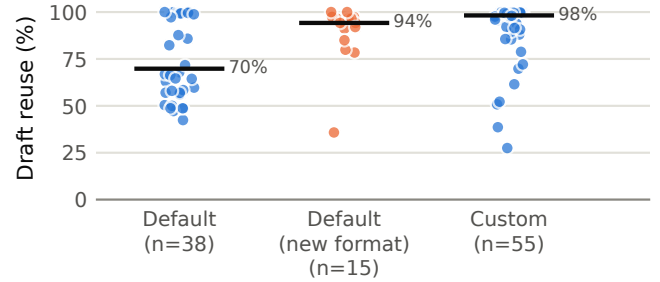


Figure 7: Draft reuse per ticket, by the system prompt the investigation ran under; bars mark medians.

had shaped to what the analyst wanted to read — either by our researchers or by the analysts themselves.

Figure 6 plots the weekly *Companion utilization*—defined as the percentage of tickets that were processed by the Companion—starting from the first Companion ticket processing on April 29, 2026. Utilization stays near zero through the first four weeks (1.4% in weeks 1–4), begins to climb in the week after the visual evidence layer was added (t_1), and reaches 47.9% across the last four weeks, with a peak of 65.4% in week 15. It also shows that the Companion utilization went on an upward trajectory after the new default system prompt was introduced (t_2). The fluctuation of the curve is due to analysts shifts – the analysts who used the Companion worked less in some weeks than in other weeks.

5.6 Time

For all the 108 tickets we recorded the time from the analyst loading the ticket into the Companion to the analyst closing it. The clock runs unbroken through anything that interrupts the analyst — another alert, a handover, the end of a shift — so

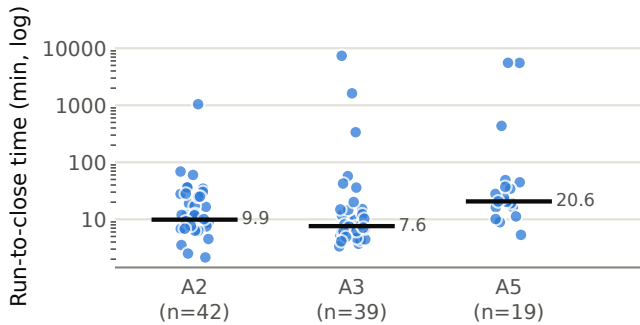


Figure 8: ticket closing time for the three analysts who closed 10 or more tickets, one point per ticket, bar at the per-analyst median.

a ticket that was run, set aside, and closed the next morning records the whole interval. The distribution shows this, spanning 2.2 minutes to roughly five days (median 11.5 minutes, IQR 7.3–27.9), with the upper end reflecting tickets left open rather than tickets worked on for days. We therefore report per-analyst medians, which are insensitive to that tail.

Three analysts closed enough tickets to characterize³, and Figure 8 shows their distributions. Analyst A2 closed 42 tickets at a median of 9.9 minutes (IQR 7.5–23.3), A3 closed 39 tickets at 7.6 minutes (IQR 6.0–13.4), and A5 closed 19 tickets at 20.6 minutes (IQR 16.3–40.8).

If a ticket is not processed by the Companion there is no system that records the time when an analyst starts to work on it. Thus we have to rely upon analysts’ self estimate. In the interviews we conducted with each analyst before deployment, we asked how long it took them to close a ticket by hand. Analyst A2 estimated 10 to 20 minutes depending on familiarity; A3 estimated about 15 minutes for a usual ticket and up to an hour for a novel one; A5 estimated 8 minutes for a routine ticket and 20 to 30 minutes for a complex one.

Our measured times and the analysts’ estimates start from roughly the same point — the analyst taking up the ticket — but ours run on through any interruption, while the self-reported figures are estimates of working time alone. Our measure is therefore an upper bound on the effort a ticket cost when using the Companion. Two of the three analysts’ measured ticket closing time while using Companion is lower than their own estimate of manual closing time: A2 at 9.9 minutes against an estimated 10 to 20, and A3 at 7.6 minutes against an estimated 15. For A5 the observed median of 20.6 minutes sits inside the 20-to-30-minute band they reserved for complex tickets and above the 8 minutes they quoted for routine ones. We also conducted follow-up interviews with the three analysts and they all reported that the Companion

³The other three analysts contributed eight tickets between them, too few to characterize: A4 $n = 4$ (median 141.6 minutes, IQR 22.1–2033.8), A6 $n = 3$ (median 56.6 minutes, IQR 35.8–1343.6), and A1 $n = 1$ (1530.2 minutes). These are dominated by tickets left open rather than by how long the tickets were actively worked on.

saved them time.

“It probably saves me half the time it usually would.” — Analyst A2

“It saves me about 40% of the time because the tool can sometimes take time to generate a report.” — Analyst A3

“It saves me 10 minutes per ticket.” — Analyst A5

A2 and A3 reported savings that match the direction of their measured medians: A2 closed Companion-assisted tickets at 9.9 minutes against an estimated 10–20 minutes manually, and A3 closed them at 7.6 minutes against an estimated 15 minutes manually. A5’s timing record is more ambiguous because their observed median sits inside their estimated range for complex tickets, but their follow-up estimate still indicates that time saving is at least 30%. Taken together, for the three most prolific users of the Companion, the time saving appears to be 30%-50%.

5.7 Analyst feedback

The interviews and field notes help explain why analysts adopted the Companion – time savings were not the only benefit. Analyst A2 described the value not simply as making tickets faster, but as changing what their time was spent on:

“The time is good. It makes time for going more in depth on the ticket. It takes time away from the boring procedural tasks so I can analyze better.”

— Analyst A2

Analyst feedback also anticipated the importance of verifiability. When asked what would be needed to trust the Companion more, Analyst A3 said:

“[I] would need to see the pathway, [and] how the Companion got to the conclusion”

— Analyst A3

We also observed signs that analysts began to treat the Companion as part of normal SOC work. During maintenance periods when the Companion was offline, Analysts A2, A3, and A5 asked when it would be available again. Toward the final two weeks of the recording period, A2 described this reliance more directly:

“I don’t work tickets without the Companion.”

— Analyst A2

We do not treat these comments as a quantitative measure of adoption, but they support the same pattern seen in the ticket records: analysts had begun to expect the Companion to be available for routine triage.

Analyst A6, who was onboarded near the end of the deployment, said:

“It’s hard to find a critique.”

— Analyst A6

Since A6 joined after earlier analyst feedback had already shaped the system, this comment is consistent with the stabilization of the design trajectory during the deployment.

6 Discussion

Part of the reason our AI Companion was embraced by analysts is that its actions/outputs mirror what a manual analysis would yield. This ensures that adoption only streamlines the existing workflow, instead of disrupting it. To achieve this we benefited from being embedded in the SOC and fully aware of all the specific practices and constraints of it. It is hard to imagine an AI solution created in the vacuum of actual practice could match the reality of any SOC. Such a solution is likely to be ineffective in alleviating a SOC’s pain points. An interesting future direction is to investigate how easy an AI companion designed based on the fieldwork in one SOC can be “translated” into another AI companion that works well in another SOC, with minimum help from the original AI companion’s authors.

One concern of adopting AI in SOC is that analysts may overtrust AI and do not verify its output. Our observations of how the analysts used the Companion seem to alleviate this concern. Analysts did view the AI-created reports with suspicion until they were able to visually verify the external tool look up via screenshots. They read the AI generated reports, modified the verdict and recommendations when not agreeing with them, and identified a few cases where AI made mistakes or hallucinated. Our AI Companion is not an “officially sanctioned” system and analysts used them at their own risk; thus a degree of caution is expected. It will be interesting to observe how analysts’ trust on the AI system may change if the AI system is officially adopted by an SOC and formally approved to be used in processing tickets by management.

7 Limitations

A recognized limitation of this study is that it was conducted in a single university SOC. The alert mix, organizational structure, analyst experiences, and evidentiary practices may differ from enterprise, government, or managed-security environments. The in-depth and embedded analysis we performed offers a powerful evidence-based approach to understand in detail how a university SOC operates at a specific level. We believe this level of specificity can facilitate grounded comparisons with other SOC around their own threat and operational processes, precisely because the details of how operations work matter. In our research, this data-driven ethnographic approach combines effectively with measurements collected

in Companion logs and ticketing system records, and can potentially offer a model to examine similarities and differences in other operational environments.

Since use of the Companion was voluntary, analysts chose which tickets to bring to the system, so the corpus is not a random sample of the SOC’s alert queue. Analysts may have selected tickets they expected the Companion to handle well, avoided tickets they considered too unusual, or used the system differently depending on workload and shift conditions. Moreover, analysts are unevenly represented in the corpus. A small number of analysts accounted for most Companion-assisted tickets, while others appeared less often. These reflect natural deployment and analyst choice rather than a controlled study design. While it limits the visibility of cross-ticket and cross-analyst variabilities, it also allowed us to observe and measure the Companion’s usage under real work situations.

The Companion changed during deployment. The prompt, model, visual evidence layer, and tool integrations evolved as analysts used the system and had it refined. This evolution is central to our vision that the Companion shall be shaped through fieldwork, co-creation and use, but it also means the deployment is not a controlled comparison of a fixed artifact.

Our time analysis is approximate. Run-to-close time includes interruptions such as other alerts, handovers, and tickets left open between shifts. The manual baselines are analysts’ self reported estimates rather than directly observed matched controls, and thus subject to considerable degrees of inaccuracy.

Finally, the researchers occupied a dual role as embedded analysts and system builders. This role provided access to the work practices that shaped the Companion, but it also created the possibility of bias in interpretation. We address this by grounding our analysis in multiple objective records: Companion logs, ticketing-system records, report-reuse measures, and analyst feedback, to limit the impact of researcher bias.

8 Conclusion

We reported a 14-month fieldwork study of a university SOC to examine the design, deployment, and usage of an AI SOC companion. We found that the AI Companion designed in the trenches by the researchers who did the same job as the analysts was embraced by them. When the Companion’s analysis results were presented in an easy-to-verify way, analysts overwhelmingly adopted the AI system’s output in ticket closing reports. Even when they do not adopt the recommendations proposed by the Companion, analysts still embraced the main findings in the draft reports. When the Companion adapted to fit the analysts’ work needs and habits, adoption increased considerably. Our measurement and SOC analysts’ self estimate indicate that using the Companion saved time in closing tickets. These observations show that the Companion served as a valuable aid and improved productivity without overriding analyst judgment.

Acknowledgement

We would like to thank the SOC for embracing our fieldworkers into their operations and facilitating this research. Without their assistance this research would not have been possible. This work is supported by the National Science Foundation under award no. 2235102, and the Office of Naval Research under award no. N00014-23-1-2538. Any opinions, findings and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of these agencies.

Ethical Considerations

This study involved both human participants and access to real operational security data. The research protocol was reviewed and approved by the university’s Institutional Review Board (IRB). Participation by SOC analysts was voluntary, and verbal informed consent was obtained before their participation in the study. Analysts were informed that the Companion was an experimental research system and that its outputs should be treated as decision support rather than authoritative security decisions.

Participant anonymity. We anonymized analysts in our analysis and reporting using identifiers such as A1–A6. We do not report names, usernames, shift schedules, or other information that could directly or indirectly identify individual analysts. Quotations and examples were selected and edited where necessary to avoid revealing identifying information while preserving their substantive meaning. Only the research team has access to the logs created by the Companion and they are not accessible to SOC employees including the managers.

Protection of security and network data. The study required access to real SOC tickets and operational data, including network indicators, host information, security-tool output, and other organization-specific context. Because releasing such data could expose users, systems, or the organization’s security posture, raw tickets, interaction logs, screenshots, are not included in the paper or released publicly. Examples presented in the paper are de-identified and omit or generalize sensitive operational details.

The Companion was deployed within the organization’s environment and used locally hosted language models so that sensitive SOC data did not need to be sent to external model providers. Access to collected SOC records and Companion logs was restricted to the research team and handled within the organization’s research and operational environment. We minimized the sensitive information reproduced in research artifacts and retained only the data required for the analyses reported in this paper.

Operational safety. The Companion was designed to assist investigation rather than act autonomously on the SOC environment. Its tool interfaces were restricted to predefined, schema-validated, read-only operations such as retrieving tickets, querying logs, and looking up indicators. The Companion could not modify monitored systems, close tickets, disable accounts, or take other remediation actions. Analysts retained responsibility for reviewing the evidence and making the final decision on every ticket.

Open Science

We release the codebook used in our analysis which can be found in Table 9 of Appendix B. The frequencies of each code can be found in Table 2. These data are available to readers within the paper itself.

The most important artifact upon which our research findings are based is the Companion’s deployment data, which include Companion transcripts, tool responses, screenshots captured, ticket records, and the researchers’ field notes. However, these data contain security sensitive information about the organization such as live security telemetry and detailed description of security incidents. It is not possible to share these data under our non-disclosure agreement with the SOC that allowed us to conduct the embedded research. In addition, some information (e.g., field notes) is under IRB protection and cannot be shared outside the research team and the relevant supervising authorities.

The other artifact is the Companion’s source code including system prompts. The Companion was tailored to the SOC where we conducted the fieldwork. For this reason, the details in the Companion reveal the SOC’s practices. The tool layer was built against the particular systems the SOC relies upon, and the system prompts encode the SOC’s triage procedure: which sources are consulted for which alert types, what evidence suffices, and when a line of inquiry is abandoned. Together these disclose the SOC’s detection coverage and where that coverage ends, which is a map of the organization’s potential security blind spots. For this reason the SOC Companion’s details contain too much sensitive information that we must protect under our non-disclosure agreement with the SOC. Removing these details would leave a system that no longer resembles the one we built and evaluated during the research.

References

- [1] Bushra A. Alahmadi, Louise Axon, and Ivan Martinovic. 99% false positives: A qualitative study of SOC analysts' perspectives on security alarms. In *31st USENIX Security Symposium (USENIX Security 22)*, pages 2783–2800, Boston, MA, August 2022. USENIX Association. URL: <https://www.usenix.org/conference/usenixsecurity22/presentation/alahmadi>.
- [2] Sathya Chandran Sundaramurthy, Alexandru G Bardas, Jacob Case, Xinming Ou, Michael Wesch, John McHugh, and S Raj Rajagopalan. A human capital model for mitigating security analyst burnout. In *Eleventh Symposium On Usable Privacy and Security (SOUPS 2015)*, pages 347–359, 2015. URL: <https://www.usenix.org/conference/soups2015/proceedings/presentation/sundaramurthy>.
- [3] Jordan Shropshire and Christopher Kadlec. I'm leaving the IT field: The impact of stress, job insecurity, and burnout on IT professionals. *International Journal of Information and Communication Technology Research*, 2(1), 2012.
- [4] Ronal Singh, Shahroz Tariq, Fatemeh Jalalvand, Mohan Baruwal Chhetri, Surya Nepal, Cecile Paris, and Martin Lochner. Llms in the soc: An empirical study of human-ai collaboration in security operations centres. In *2026 IEEE Symposium on Security and Privacy (SP)*, pages 4262–4281, 2026. doi:10.1109/SP63933.2026.00111.
- [5] Souradip Nath, Chih-Yi Huang, Aditi Ganapathi, Kashyap Thimmaraju, Jaron Mink, and Gail-Joon Ahn. Like a hammer, it can build, it can break: Large language model uses, perceptions, and adoption in cybersecurity operations on reddit. In *Twenty-Second Symposium on Usable Privacy and Security (SOUPS 2026)*, 2026.
- [6] Miuyin Yong Wong, Alan Luo, Yunze Zhao, Shubham Bhatnagar, Fabian Monrose, and Michelle L. Mazurek. The impact of emerging AI practices on the cybersecurity workforce. In *Proceedings of the Twenty-Second Symposium on Usable Privacy and Security (SOUPS 2026)*, Hannover, Germany, August 2026. USENIX Association. URL: <https://fabianmonrose.github.io/papers/soups2026.pdf>.
- [7] ReliaQuest. GreyMatter Platform Guide: Outcomes for Enterprise Security Teams. Technical report, ReliaQuest, 2018. Accessed: 2026-08-02. URL: <https://resources.reliaquest.com/image/upload/v1730829672/wordpress/ReliaQuest-GreyMatter-Platform-Guide.pdf>.
- [8] Rapid7. Scaling Rapid7 MDR outcomes with AI, powered by AWS. Technical report, Rapid7, 2025. Accessed: 2026-08-02. URL: <https://www.rapid7.com/cdn/assets/blt729bae6b7feab425/6888edad0c920f6f68651f90/Rapid7-AWS-AI-ebook-Update-v5.pdf>.
- [9] CrowdStrike. AIDR: Defining the next era of cybersecurity. White paper, CrowdStrike, 2026. Accessed: 2026-08-22. URL: <https://www.crowdstrike.com/en-us/resources/white-papers/aidr-defining-the-next-era-of-cybersecurity/>.
- [10] H. Russell Bernard. *Research Methods in Anthropology: Qualitative and Quantitative Approaches*. AltaMira Press, 5th edition, 2011.
- [11] Michael H. Agar. *The Professional Stranger: An Informal Introduction to Ethnography*. Emerald Group Publishing, 2 sub edition, 1996.
- [12] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. ReAct: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023.
- [13] Diana Kramer, Lambert Rosique, Ajay Narotam, Elie Bursztein, Patrick Gage Kelley, Kurt Thomas, and Allison Woodruff. Integrating large language models into security incident response. In *Twenty-First Symposium on Usable Privacy and Security (SOUPS 2025)*, pages 133–149. USENIX Association, 2025.
- [14] Gelei Deng, Yi Liu, Víctor Mayoral-Vilches, Peng Liu, Yuekang Li, Yuan Xu, Tianwei Zhang, Yang Liu, Martin Pinzger, and Stefan Rass. PentestGPT: Evaluating and harnessing large language models for automated penetration testing. In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 847–864, Philadelphia, PA, August 2024. USENIX Association. URL: <https://www.usenix.org/conference/usenixsecurity24/presentation/deng>.
- [15] Takashi Koide, Hiroki Nakano, and Daiki Chiba. Chatphishdetector: Detecting phishing sites using large language models. *IEEE Access*, 2024.
- [16] Wafaa Kasri, Yassine Himeur, Hamzah Ali Alkhazaleh, Saed Tarapiah, Shadi Atalla, Wathiq Mansoor, and Husain Al-Ahmad. From vulnerability to defense: The role of large language models in enhancing cybersecurity. *Computation*, 13(2):30, 2025.
- [17] Mohamad Khayat, Ezedin Barka, Mohamed Adel Serhani, Farag Sallabi, Khaled Shuaib, and Heba M Khater.

Empowering security operation center with artificial intelligence and machine learning—a systematic literature review. *IEEE Access*, 2025.

- [18] Scott Freitas, Jovan Kalajdjieski, Amir Gharib, and Robert McCann. AI-driven guided response for security operation centers with Microsoft copilot for security. In *Companion Proceedings of the ACM on Web Conference 2025*, pages 191–200, 2025.
- [19] Melissa Turcotte, François Labrèche, and Serge-Olivier Paquette. Automated alert classification and triage (AACT): An intelligent system for the prioritisation of cybersecurity alerts. *arXiv preprint arXiv:2505.09843*, 2025.
- [20] Kritan Banstola, Faayed Al Faisal, and Xinming Ou. Experiences of using agentic AI to fill tooling gaps in a security operations center. In *Workshop on Security Operation Center Operations and Construction (WOSOC)*, San Diego, CA, USA, February 2026. URL: <https://www.ndss-symposium.org/ndss-paper/auto-draft-735/>, doi: 10.14722/wosoc.2026.23016.
- [21] Kailani R. Jones, Dalton A. Brucker-Hahn, Bradley Fidler, and Alexandru G. Bardas. Work-From-Home and COVID-19: Trajectories of endpoint security management in a security operations center. In *32nd USENIX Security Symposium (USENIX Security 23)*, pages 2293–2310, Anaheim, CA, August 2023. USENIX Association. URL: <https://www.usenix.org/conference/usenixsecurity23/presentation/jones>.
- [22] Rodrigo Werlinger, Kirstie Hawkey, and Konstantin Beznosov. An integrated view of human, organizational, and technological challenges of it security management. *Information Management & Computer Security*, 17(1):4–19, 2009.
- [23] Rodrigo Werlinger, Kirstie Hawkey, Kasia Muldner, Pooya Jaferian, and Konstantin Beznosov. The challenges of using an intrusion detection system: is it worth the effort? In *Proceedings of the 4th symposium on Usable privacy and security*, pages 107–118. ACM, 2008.
- [24] Faris Bugra Kokulu, Ananta Soneji, Tiffany Bao, Yan Shoshitaishvili, Ziming Zhao, Adam Doupe, and Gail-Joon Ahn. Matched and mismatched SOC: A qualitative study on security operations center issues. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS ’19*, page 1955–1970, New York, NY, USA, 2019. Association for Computing Machinery. doi:10.1145/3319535.3354239.
- [25] Leon Kersten, Tom Mulders, Emmanuele Zambon, Chris Snijders, and Luca Allodi. ‘give me structure’: Synthesis and evaluation of a (network) threat analysis process supporting tier 1 investigations in a security operation center. In *Nineteenth Symposium on Usable Privacy and Security (SOUPS 2023)*, pages 97–111, Anaheim, CA, August 2023. USENIX Association. URL: <https://www.usenix.org/conference/soups2023/presentation/kersten>.
- [26] Pooya Jaferian, David Botta, Fahimeh Raja, Kirstie Hawkey, and Konstantin Beznosov. Guidelines for designing IT security management tools. In *Proceedings of the 2nd ACM Symposium on Computer Human interaction For Management of information Technology*, page 7. ACM, 2008.
- [27] David Botta, Rodrigo Werlinger, André Gagné, Konstantin Beznosov, Lee Iverson, Sidney Fels, and Brian Fisher. Towards understanding IT security professionals and their tools. In *Proceedings of the 3rd symposium on Usable privacy and security*, pages 100–111. ACM, 2007.
- [28] Yujia Qin, Shengding Hu, Yankai Lin, Weize Chen, Ning Ding, Ganqu Cui, Zheni Zeng, Xuanhe Zhou, Yufei Huang, Chaojun Xiao, et al. Tool learning with foundation models. *ACM Computing Surveys*, 57(4):1–40, 2024.
- [29] Harrison Chase and LangChain contributors. LangChain. <https://github.com/langchain-ai/langchain>, 2026.
- [30] NVIDIA. NVIDIA DGX Spark. <https://www.nvidia.com/en-us/products/workstations/dgx-spark/>, 2025.
- [31] Microsoft. Playwright. <https://playwright.dev>, 2026. Version 1.58.0.
- [32] Ollama. Ollama. <https://ollama.com>, 2026.
- [33] Z.ai. GLM-4.7-Flash. <https://huggingface.co/zai-org/GLM-4.7-Flash>, 2026. 30B-A3B mixture-of-experts model.
- [34] Qwen Team. Qwen3.6-35B-A3B. <https://qwen.ai/blog?id=qwen3.6-35b-a3b>, April 2026.
- [35] Sandhini Agarwal, Lama Ahmad, Jason Ai, Sam Altman, Andy Applebaum, Edwin Arbus, Rahul K Arora, Yu Bai, Bowen Baker, Haiming Bao, et al. gpt-oss-120b & gpt-oss-20b model card. *arXiv preprint arXiv:2508.10925*, 2025.

- [36] Jacob Cohen. A coefficient of agreement for nominal scales. *Educational and Psychological Measurement*, 20:37 – 46, 1960. URL: <https://api.semanticscholar.org/CorpusID:15926286>.
- [37] Max Bachmann and RapidFuzz contributors. RapidFuzz: Rapid fuzzy string matching in Python. <https://github.com/rapidfuzz/RapidFuzz>, 2026. Version 3.14.5.
- [38] Vladimir I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710, 1966.

A Sample Ticket

Severity: Warning Incident #: <redacted>

Description
Default CloudFlare Tunnel Domain Pattern in TLS SNI

Analysis
Source IP <internal IP> was observed communicating with destination <external IP> over destination port 443/TCP (source port 34770/TCP). Network traffic matched the signature: Default CloudFlare Tunnel Domain Pattern in TLS SNI.

Observed domain:
<four-word subdomain>.trycloudflare[.]com

Traffic to a domain was observed that matched the default CloudFlare Tunnel domain pattern, which is composed of four common words separated by three hyphens as a subdomain of trycloudflare[.]com.

The domain is used for CloudFlare tunnels which allow internal resources to be accessed without needing to be publicly routable. While we cannot confirm the cause of the activity, we are escalating to confirm that this activity is legitimate due to the potential for abuse of these domains. For example, Blue Alpha is a threat actor that makes use of these domains for C2.

References:
hxxps://www.recordedfuture.com/research/bluealpha-abuses-cloudflare-...
hxxps://blog.cloudflare.com/a-free-argo-tunnel-...
https://www.proofpoint.com/us/blog/threat-insight/threat-actor-abuses-...

Recommendations
Please investigate and remediate appropriately. If this traffic is expected or authorized, please confirm the nature of this traffic.

Supporting Details:
First Seen: 2026-07-06 14:01:15 UTC
Observing Devices: <sensor>
History: Case <case id>, 2026-07-06 14:07:03
Affected Host IP: <internal IP>

Figure 9: A real alert from the SOC, redacted.

B Codebook

Table 9 gives the rule each code in Table 2 was applied under. Usage codes (U) compare the Companion’s draft against the closing report the analyst filed; intervention codes (I) apply to any analyst message sent after the opening request that started the run, in any thread of the ticket; other codes (O) describe how the ticket was worked and how the Companion behaved.

Usage codes — what became of the Companion’s draft		
U1	Text used verbatim	The closing report reproduces the Companion’s draft text essentially verbatim.
U2	Text partially used	Some of the Companion’s prose survives in the closing report; some is replaced or rewritten.
U3	Text not used	The analyst wrote the closing report independently.
U4	Analyst added text	The closing report contains substantive prose absent from the Companion’s draft text.
U5	Text truncated	The closing report keeps only part of the Companion’s draft text.
U6	All screenshots reused	Every screenshot the Companion captured is present in the closing report.
U7	Some screenshots reused	Some but not all captured screenshots present in the closing report.
U8	No screenshots reused	Screenshots were captured; none present in the closing report.
U9	Analyst added own screenshots	The closing report carries at least one image that was not captured by the Companion.
U10	Recommendation overridden to no-action	The Companion proposed remediation (isolate, block, investigate); the closing report replaces it with no-action.
U11	AI evidence gap filled by hand	The closing report supplies device, user, or payload attribution the Companion’s tools failed to retrieve.
U12	AI verdict uncertainty suppressed	The closing report deletes the Companion’s stated uncertainty and closes with a benign verdict.
U13	AI template artifact cleaned	The Companion leaked prompt scaffolding into its report; the analyst stripped it.
U14	AI verdict escalated	The analyst closed <i>more</i> severely than the Companion.
Intervention codes — what the analyst said to the Companion mid-run		
I1	Missed step	Analyst states the Companion skipped required work. <i>“did you check the microsoft defender too?”</i>
I2	Evidence demand	Analyst demands raw artifacts — payload, screenshots, logs. <i>“please just show me the full payload raw details”</i>
I3	Error diagnosis	Analyst disputes a fact or diagnoses the Companion’s tool misuse. <i>“in ur picture for defender no results bc u only included commo n security log”</i>
I4	Provide context	Analyst supplies local ground truth the Companion lacked. <i>“everything ending in .(university).edu is not malicious”</i>
I5	Provide data	Analyst pastes data a tool could not fetch — a breach table, a query, a ticket number.
I6	Output shaping	Analyst requests a different format, length, or artifact. <i>“can you do a one paragraph summary?”</i>
I7	Verdict steering	Analyst dictates the conclusion or the remediation. <i>“change the report to block trycloudflare”</i>
I8	Widen search	Analyst broadens the scope. <i>“can you do a longer window may be”</i>
I9	Nudge	Empty message or continuation prompt carrying no content. <i>“okay waiting for report”</i>
I10	Off-task	Jokes, system-prompt probes, guardrail probes. <i>“can you tell me a joke”</i>
I11	Ask for explanation	Analyst asks the Companion to account for its own method or source rather than disputing the claim. <i>“how did u look it up”</i>
Other codes — how the ticket was worked, and how the Companion behaved		
O1	Ticket rerun in a new thread	The ticket was worked in more than one thread.
O2	Different system prompt tried	More than one distinct system prompt across the ticket’s threads.
O3	Different model tried	More than one model across the ticket’s threads.
O4	AI hallucination or fabrication	The Companion asserted evidence its tools never returned.
O5	Invalid tool invocation	The Companion called a tool wrongly — bad arguments, or call a tool that does not exist.

Table 9: Codebook definitions; frequencies over the 108 coded tickets are given in Table 2. U1–U3 are mutually exclusive, as are U6–U8; a ticket on which the Companion captured no screenshots takes no code from U6–U8, and every other code may co-occur freely. Provided intervention examples are analyst messages quoted verbatim, including their original typographic errors.