Contents

[Chapter 4: Analysis of Results and Discussion 2](#_Toc208251938)

[4.1 Introduction of the Analysis and Discussion 2](#_Toc208251939)

[4.2 Presentation of Findings 2](#_Toc208251940)

[4.2.1 Test Case 1: Broad Prompt 2](#_Toc208251941)

[4.2.2 Test case 2: Mid prompt 7](#_Toc208251942)

[4.2.3 Test Case 3: Specific Prompt 12](#_Toc208251943)

[4.3 Cross-Reference Analysis with Other Studies 17](#_Toc208251944)

[4.4 Overall Test Cases Assessment 20](#_Toc208251945)

[References 22](#_Toc208251946)

# Chapter 4: Analysis of Results and Discussion

## 4.1 Introduction of the Analysis and Discussion

This research hypothesised that large language models (LLMs) could be used to detect security misconfigurations in Cisco IOS configurations when assessed against the CIS Benchmarks. By employing a structured evaluation framework built on controlled datasets and carefully engineered prompts, the study aimed to measure both the accuracy and reliability of GPT-4o in identifying benchmark-aligned errors as well as common configuration mistakes. The primary objective was to determine whether prompt engineering could be used to guide the models toward more consistent and accurate compliance assessments. Additionally, the research aimed to examine how varying levels of prompt specificity (Broad, Mid and Specific) shaped the model’s performance, and to compare the outcomes across multiple protocol domains. In the following sections, the analysis will present the results of these evaluations, compare model performance across prompt types and protocols, and critically discuss the implications of these findings for the integration of LLMs into professional network auditing and compliance workflows.

## 4.2 Presentation of Findings

### 4.2.1 Test Case 1: Broad Prompt

The Broad prompt was designed as the baseline test, requiring the model to review configurations without explicit reference to the CIS Benchmarks. Its purpose was to measure GPT’s ability to apply general networking knowledge when identifying misconfigurations and Mistype errors. The results indicated limited effectiveness in detecting benchmark aligned issues, with stronger but still inconsistent performance in identifying simple Mistype errors.

**Figure 1**: Broad Prompt Misconfiguration Detection Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Protocol** | **Number of Errors** | **1 Error Detected** | **2 Errors Detected** | **3 Errors Detected** |
| AAA | 1 | 5/5 | - | - |
| 2 | 1/5 | 1/5 | - |
| 3 | 4/5 | 0/5 | 0/5 |
| EIGRP | 1 | 0/5 |  |  |
| 2 | 1/5 | 0/5 |  |
| 3 | 0/5 | 0/5 | 0/5 |
| OSPF | 1 | 1/5 |  |  |
| 2 | 2/5 | 3/5 |  |
| 3 | 1/5 | 2/5 | 1/5 |
| RIP | 1 | 0/5 |  |  |
| 2 | 2/5 | 0/5 |  |
| 3 | 1/5 | 0/5 | 0/5 |

**Table 1:** Broad Prompt Results (Configs with all errors detected / Total configs)

|  |  |  |
| --- | --- | --- |
| **Protocol** | **Amount of Errors Detected (Number of detected errors / Total number of errors)** | **PP Score** |
| AAA | 12/30 | 40% |
| EIGRP | 1/30 | 3% |
| OSPF | 17/30 | 57% |
| RIP | 3/30 | 10% |
|  | **Total PP Score** | 28% |

**Table 2:** Broad Prompt PP Scores

|  |  |  |
| --- | --- | --- |
| **Protocol** | **Mistypes Detected**  **(Number of Mistypes detected / Total number of mistypes)** | **PP Score** |
| AAA | 2/5 | 60% |
| EIGRP | 1/5 | 80% |
| OSPF | 1/5 | 80% |
| RIP | 1/5 | 80% |
|  | **Total PP Score** | 75% |

**Table 3:** Broad Prompt Mistype Detection Results

Figure 1 and Tables 1 and 2 summarise the results across all four protocols. Out of 120 benchmark-related errors, only 33 were detected, resulting in a total PP Score of 28%. Accuracy varied significantly by protocol. OSPF achieved the highest PP Score of 57%, followed by AAA at 40%, while RIP and especially EIGRP performed poorly, with scores of 10% and 3% respectively. By contrast, Mistype errors were detected resulting in a total PP Score of 75% as shown in Table 3. For the latter, the accuracy varied with EIGRP, OSPF and RIP all having PP Scores of 80% while AAA scored a PP Score of 60%. This indicated that GPT was more reliable at identifying basic syntactic mistakes than at enforcing benchmark compliance.

|  |  |  |  |
| --- | --- | --- | --- |
| **Protocol** | **Errors** | **Errors Detected (Amount of the error detected / Amount of total errors of that type)** | **Detection Rate** |
| AAA | service password-encryption | 3/3 | 100% |
| Enable secret | 1/1 | 100% |
| Line vty | 2/2 | 100% |
| aaa authorization exec | 2/3 | 66% |
| aaa accounting system | 3/5 | 60% |
| aaa authorization config-commands | 1/2 | 50% |
| aaa authorization reverse-access | 0/4 | 0% |
| aaa accounting commands 15 | 0/1 | 0% |
| aaa new-model | 0/2 | 0% |
| aaa authorization network | 0/2 | 0% |
| EIGRP | ip authentication key-chain eigrp | 1/9 | 11% |
| key-chain | 0/4 | 0% |
| key | 0/3 | 0% |
| key-string | 0/1 | 0% |
| ip authentication mode eigrp md5 | 0/4 | 0% |
| router eigrp | 0/2 | 0% |
| passive-interface | 0/7 | 0% |
| OSPF | area authentication message-digest | 5/9 | 56% |
| ip ospf message-digest-key | 10/19 | 52% |
| router ospf | 1/2 | 50% |
| RIP | key-string | 1/1 | 100% |
| ip rip authentication key-chain | 1/2 | 50% |
| key-chain | 1/3 | 33% |
| ip rip authentication mode md5 | 0/4 | 0% |
| version 2 | 0/2 | 0% |
| network | 0/4 | 0% |
| router rip | 0/4 | 0% |
| passive-interface | 0/2 | 0% |
| redistribute | 0/4 | 0% |
| maximum-paths | 0/3 | 0% |
| offset-list | 0/1 | 0% |

**Table 4:** Broad Error Detections

To better understand recurring strengths and weaknesses, the configurations were further analysed to identify specific requirements that were either consistently detected or frequently overlooked. Table 4 shows that certain fundamental commands, “service password-encryption”, “enable secret” and “line vty” in AAA were always detected and for RIP “key-string” which was always detected. Other configurations such as “ip ospf message-digest-key md5” and “area authentication message-digest” in OSPF were flagged the most when missing even though it still cannot be counted as reliable, suggesting that the model could recognise high-level authentication mechanisms and simple per-interface security features. Consequently, EIGRP remained the least reliable domain in this test, indicating that dependable detection likely requires benchmark excerpts or stricter, protocol-scoped prompting.

Another type of analysis that was done considered the references that GPT cited when justifying its outputs. To explore this, three configurations from each protocol were sampled (12 Total samples), and all references mentioned in the model’s reasoning were documented. As shown in Table 5, GPT frequently cited Cisco documentation, CIS Benchmarks and NIST guidelines, occasionally drawing on references such as NSA, RFC standards, DISA STIGs or real-world application examples. While the diversity of references suggests exposure to authoritative material, their use was inconsistent which means that each individual config was evaluated with different guidelines. Cisco, CIS and NIST were by far the most commonly used, indicating that GPT anchored its reasoning on broadly recognised security sources, but without the precision required for consistent compliance auditing.

|  |  |
| --- | --- |
| **Reference Source** | **Count** |
| Cisco References | 12 |
| CIS References | 11 |
| NIST References | 10 |
| Real-world applications Referenced | 5 |
| NSA References | 2 |
| RFC References | 2 |
| DISA STIG References | 1 |

**Table 5:** Most common references used by Broad Prompt

|  |
| --- |
|  |

**Figure 2:** Broad Prompt Output Example

The model frequently produced short and list-like responses as shown in Figure 2 rather than detailed checklists. While it often flagged certain recurring issues, it consistently failed to detect critical requirements in protocols as shown in Table 5 Overall, the Broad prompt demonstrated the weakness of unguided prompting. While capable of identifying obvious typos and occasional misconfigurations, its detection of benchmark-aligned errors was inconsistent and unreliable across protocols. These findings establish a baseline for comparison with the Mid and Specific prompts, where addition guidance was introduced.

### 4.2.2 Test case 2: Mid prompt

The Mid prompt introduced explicit reference to the CIS Benchmarks but did not provide excerpts of the standards themselves. Instead, the model was instructed to review the configurations “according to CIS Benchmarks”, requiring them to rely on any embedded knowledge of the framework gained during training. This test case therefore measured whether mentioning the CIS Benchmarks within the prompt alone improved detection accuracy compared to a purely unguided assessment.

**Figure 3**: Mid Prompt Misconfiguration Detection Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Protocol** | **Number of Errors** | **1 Error Detected** | **2 Errors Detected** | **3 Errors Detected** |
| AAA | 1 | 4/5 | - | - |
| 2 | 3/5 | 0/5 | - |
| 3 | 2/5 | 1/5 | 0/5 |
| EIGRP | 1 | 1/5 | - | - |
| 2 | 3/5 | 0/5 | - |
| 3 | 1/5 | 2/5 | 0/5 |
| OSPF | 1 | 1/5 | - | - |
| 2 | 2/5 | 2/5 | - |
| 3 | 1/5 | 1/5 | 2/5 |
| RIP | 1 | 1/5 | - | - |
| 2 | 2/5 | 1/5 | - |
| 3 | 0/5 | 0/5 | 0/5 |

**Table 6:** Protocol Mid Results (Configs with all errors detected / Total configs)

|  |  |  |
| --- | --- | --- |
| **Protocol** | **Amount of Errors Detected (Number of detected errors / Total number of errors)** | **PP Score** |
| AAA | 11/30 | 37% |
| EIGRP | 5/30 | 30% |
| OSPF | 16/30 | 53% |
| RIP | 5/30 | 17% |
|  | **Total PP Score** | 33% |

**Table 7:** Mid Prompt PP Scores

|  |  |  |  |
| --- | --- | --- | --- |
| **Protocol** | **Mistypes Detected**  **(Number of Mistypes detected / Total number of mistypes)** | | **PP Score** |
| AAA | 4/5 | | 80% |
| EIGRP | 5/5 | | 100% |
| OSPF | 4/5 | | 80% |
| RIP | 4/5 | | 80% |
|  | | **Total PP Score** | 85% |

**Table 8:** Mid Prompt Mistype Results

The results showed moderate gains in benchmark-related error detection as shown in Figure 3 and Tables 6 and 7. Out of 120 misconfigurations implemented, 39 were identified, resulting in a total PP Score of 33%. Compared to the Broad prompt, this represents a small overall improvement, although the distribution of accuracy across protocols shifted. OSPF achieving the highest PP Score of 53% slightly lower than the 57% achieved under Broad, owing to one additional missed error, AAA also recorded a lower accuracy at 37% compared to the 40% in Broad, as the Mid prompt overlooked a requirement previously detected, By contrast EIGRP demonstrated a significant increase from just 3% in Broad to 30%. RIP remained the weakest domain with only 17% compared to 10% in Broad, showing only minimal improvement. Mistype errors were handled more effectively than in the Broad test case, with a PP Score of 85% compared to 75% under Broad, suggesting that benchmark references encouraged more structured and attentive parsing of syntactic issues. Protocol-specific mistype scores ranged from 80% in AAA, OSPF and RIP to 100% in EIGRP, demonstrating consistency across domains.

As with the Broad case, further analysis was carried out to determine recurring strengths and weaknesses. Table 9 highlights commands that the Mid prompt reliably detected. These included “service password-encryption”, “enable secret”, “aaa authorization exec” and “line vty” in AAA with 3/3 detections, ,“ip rip authentication key-chain” and “key-string” in RIP. While EIGRP and OSPF had no constantly detected results to be mentioned. These results suggest that the introduction of benchmark framing improved recognition of certain authentication and encryption requirements, especially within

|  |  |  |  |
| --- | --- | --- | --- |
| **Protocol** | **Errors** | **Errors Detected (Amount of the error detected / Amount of total errors of that type)** | **Detection Rate:** |
| AAA | Service password-encryption | 3/3 | 100% |
| Enable secret | 1/1 | 100% |
| Aaa authorization exec | 3/3 | 100% |
| Line vty | 2/2 | 100% |
| Aaa authorization reverse-access | 3/4 | 75% |
| Aaa authorization config-commands | 1/2 | 50% |
| aaa accounting system | 0/5 | 0% |
| Aaa accounting commands 15 | 0/1 | 0% |
| Aaa new-model | 0/2 | 0% |
| Aaa authorization network | 0/2 | 0% |
| EIGRP | key | 2/3 | 66% |
| Ip authentication mode eigrp md5 | 2/4 | 50% |
| Ip authentication key-chain eigrp | 3/9 | 33% |
| Key-chain | 1/4 | 25% |
| Passive-interface | 1/7 | 14% |
| Router eigrp | 0/2 | 0% |
| Key-string | 0/1 | 0% |
| OSPF | Ip ospf message-digest-key | 11/19 | 58% |
| Area authentication message-digest | 5/9 | 56% |
| Router ospf | 0/2 | 0% |
| RIP | Ip rip authentication key-chain | 2/2 | 100% |
| Key-string | 1/1 | 100% |
| Key-chain | 1/3 | 33% |
| Ip rip authentication mode md5 | 1/4 | 25% |
| network | 0/4 | 0% |
| Passive-interface | 0/2 | 0% |
| redistribute | 0/4 | 0% |
| Maximum-paths | 0/3 | 0% |
| Offset-list | 0/1 | 0% |
| Router rip | 0/4 | 0% |
| Version 2 | 0/2 | 0% |

OSPF.**Table 9:** Mid Error Detections

However, as shown in Table 9, the model continued to overlook several critical requirements. For AAA, “aaa accounting system”, “aaa accounting commands 15”, “aaa new-model”, “aaa authorization network” were never flagged, EIGRP exhibited weak detection of “router eigrp” and “key-string. In OSPF, the omission of “router ospf” was never detected and in RIP, nearly all errors were not detected except for “ip authentication key-chain”, “Key-string”, “key-chain” and “ip rip authentication mode md5”. These gaps demonstrate that while Mid prompting increased structure and alignment with compliance concepts, reliability in detecting mandatory benchmark rules remained inconsistent

|  |
| --- |
|  |

**Figure 4**: Mid Prompt Output Example

The Mid prompt outputs were noticeably longer and more structured, often divided into categories such as “Critical Security Issues” and “Other notable misconfigurations”, as illustrated in Figure 4. The model not only listed errors but also explained the reasoning behind each detection and occasionally suggested corrective commands. This gave the outputs a more compliance-oriented character, reflecting attempts to align the findings with security standards. However, the responses remained list-like in presentation and frequently left out the final binary statement of whether the configuration was secure, despite explicit instruction to do so. In some cases, the model extended beyond benchmark requirements by including best-practice recommendations, which although relevant, were not mandated by CIS. These tendencies demonstrate that while the Mid prompt promoted more detail and contextualisation, the model still displayed inconsistencies in adhering strictly to the CIS Benchmarks.

Finally, an examination of the references cited by GPT in its explanations in Table 10 provided further insight into how it justified its outputs. Three configurations per protocol were sampled (Total of 12 samples), and citations were documented. Cisco and CIS documentation were the most frequently invoked with them being used 12 times, following by NIST guidelines with 8 times, real-world application examples with 6 and NSA references with 4. The consistent appearance of Cisco and CIS sources suggest that the Mid prompt successfully anchored GPT’s reasoning to widely recognised compliance frameworks. However, the uneven use of supporting sources illustrates why detection accuracy remained inconsistent.

|  |  |
| --- | --- |
| **Reference Source** | **Count** |
| CIS References | 12 |
| Cisco References | 12 |
| NIST References | 8 |
| Real-world Applications Referenced | 6 |
| NSA References | 4 |

**Table 10:** Most common references used in Mid Prompt

Overall, the Mid Prompt demonstrated that referencing benchmarks within the instructions improves alignment with compliance-oriented reasoning and enhances detection of certain classes of misconfigurations. Nonetheless, accuracy gains were limited and several benchmark-critical omissions continued to be missed across all protocols.

### 4.2.3 Test Case 3: Specific Prompt

The Specific prompt represented the most constrained test case, limiting the model’s evaluation to a single protocol and attaching the corresponding section of the CIS Benchmark for reference. By providing explicit standards alongside the configuration, this test case measured the model’s ability to apply authoritative compliance rules within a defined scope.

**Figure 5**: Specific Prompt Misconfiguration Detection Results

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Protocol** | **Number of Errors** | **1 Error Detected** | **2 Errors Detected** | **3 Errors Detected** |
| AAA | 1 | 5/5 | - | - |
| 2 | 1/5 | 2/5 | - |
| 3 | 3/5 | 1/5 | 0/5 |
| EIGRP | 1 | 2/5 | - | - |
| 2 | 1/5 | 4/5 | - |
| 3 | 1/5 | 2/5 | 2/5 |
| OSPF | 1 | 3/5 | - | - |
| 2 | 3/5 | 2/5 | - |
| 3 | 0/5 | 2/5 | 3/5 |
| RIP | 1 | 2/5 | - | - |
| 2 | 4/5 | 1/5 | - |
| 3 | 3/5 | 1/5 | 0/5 |

**Table 11:** Protocol Specific Results (Configs with all errors detected / Total configs)

|  |  |  |
| --- | --- | --- |
| **Protocol** | **Amount of Errors Detected (Number of detected errors / Total number of errors)** | **PP Score** |
| AAA | 15/30 | 50% |
| EIGRP | 22/30 | 73% |
| OSPF | 23/30 | 77% |
| RIP | 13/30 | 43% |
|  | **Total PP Score** | 61% |

**Table 12:** Specific Prompt PP Scores

|  |  |  |
| --- | --- | --- |
| **Protocol** | **Mistypes Detected**  **(Number of Mistypes detected / Total number of mistypes)** | **PP Score** |
| AAA | 4 | 80% |
| EIGRP | 5 | 100% |
| OSPF | 5 | 100% |
| RIP | 3 | 60% |
|  | **Total PP Score** | 85% |

**Table 13:** Specific Prompt Mistype Results

The results shown in Tables 11 and 12 demonstrated considerably higher accuracy than the previous test cases. Out of 120 benchmark-related errors, 73 were detected, giving a total PP Score of 61% compared to 33% previously in the Mid test case. Protocol-level performance also showed notable improvements: , OSPF rose from 53% to 77%, EIGRP from 30% to 73%, AAA from 37% to 50% and RIP remaining the weakest domain, though still improved from 17% to 43%. Mistype detection was strong overall as shown in Table 13, with a combined PP Score of 85%, matching the Mid test case. However, while EIGRP and OSPF maintained perfect scores of 100% in mistype detection, RIP fell to 60% due to one error being missed. A per-protocol breakdown of misconfigurations detected is illustrated in Figure 5.

|  |  |  |  |
| --- | --- | --- | --- |
| **Protocol** | **Errors** | **Errors Detected**  **(Amount of the error detected / Amount of total errors of that type)** | **Detection Rate** |
| AAA | Service password-encryption | 3/3 | 100% |
| Enable secret | 1/1 | 100% |
| Aaa accounting commands 15 | 1/1 | 100% |
| Aaa accounting system | 4/5 | 80% |
| Aaa authorization exec | 2/3 | 67% |
| Aaa new-model | 1/2 | 50% |
| Aaa authorization config-commands | 1/2 | 50% |
| Aaa authorization network | 1/2 | 50% |
| Line vty | 1/2 | 50% |
| Aaa authorization reverse-access | 0/4 | 0% |
| EIGRP | Key-chain | 4/4 | 100% |
| Ip authentication key-chain eigrp | 9/9 | 100% |
| Ip authentication mode eigrp md5 | 3/4 | 75% |
| Key | 2/3 | 67% |
| Passive-interface | 4/7 | 57% |
| Router eigrp | 0/2 | 0% |
| Key-string | 0/1 | 0% |
| OSPF | Ip ospf message-digest-key | 15/19 | 79% |
| Area authentication message-digest | 7/9 | 78% |
| Router ospf | 1/2 | 50% |
| RIP | Key-string | 1/1 | 100% |
| Passive-interface | 2/2 | 100% |
| Ip rip authentication mode md5 | 4/4 | 100% |
| Ip rip authentication key-chain | 2/2 | 100% |
| Key-chain | 2/3 | 67% |
| Router rip | 2/4 | 50% |
| Version 2 | 0/2 | 0% |
| network | 0/4 | 0% |
| redistribute | 0/4 | 0% |
| Maximum-paths | 0/3 | 0% |
| Offset-list | 0/1 | 0% |

**Table 14**: Error’s Found/Missed Total

Further inspection of detection patterns is summarised in Table 14, which highlights the key requirements that were consistently identified. For AAA “service password-encryption”, “enable secret” and “aaa accounting commands 15” in AAA were always detected, “ip authentication key-chain eigrp” and “key-chain” were always identified, there were no errors always detected in OSPF to be noted and “ip rip authentication mode md5”, “ip rip authentication key-chain”, “passive-interface” and “key-string” in RIP were always detected. These results confirm that providing benchmark excerpts helped the model anchor to certain critical CIS requirements more reliably.

Nevertheless, certain benchmark-mandated rules were still missed, as shown in Table 14. In AAA, “aaa authorization reverse-access” was overlooked in all 0/4 cases, “router eigrp” in EIGRP was constantly overlooked and RIP being particularly weak with “version 2”, “network”, “redistribute”, “maximum-paths” and “offset-list” being consistently missed. No consistently missed requirements were observed in OSPF. These results highlight that even under conditions when authoritative CIS excerpts were supplied, GPT was not uniformly consistent in applying all rules.

|  |
| --- |
|  |

**Figure 6**: Specific Prompt Output Example

The outputs generated under the specific prompt as shown in Figure 6, were more structured and protocol-focused, closely reflecting the authoritative CIS Benchmark excerpts that had been provided. Responses were generally presented in a systematic checklist style, with clear divisions between correctly configured elements, detected issues and suggested fixes. Each item was described with a brief explanation of the security implication and a remediation step, creating outputs that resembled the format of a professional compliance audit. Although the prompt explicitly instructed the model to provide a binary statement of overall compliance, this was not consistently included across all outputs. Nevertheless, the structure and tone of the responses were more precise, formal and directly tied to benchmark requirements, showing a stronger alignment with the expectations of a standards-driven assessment.

However, analysis of GPT’s cited references shows that it did not exclusively rely on the CIS Benchmarks even when they were explicitly attached. As shown in Table 15, three configurations from each protocol were sampled (12 samples total), and GPT’s references recorded. CIS had been referenced in all 12 configurations and Cisco in 11 of them which were the most commonly referenced, followed by NIST with 8 references, real-world application sources with 4 and DIS STIG guidelines referenced 2 times. This demonstrates that although the prompt specifically instructed GPT to use the CIS Benchmarks, it continued to integrate information from a range of external sources. While Cisco and CIS remained dominant, the presence of other frameworks such as NIST and DISA STIG suggests that the model blended attached standards with embedded training data, which may partly explain its inconsistent application of certain rules.

|  |  |
| --- | --- |
| **Reference Source** | **Count** |
| CIS References | 12 |
| Cisco References | 11 |
| NIST References | 8 |
| Real-world Applications References | 4 |
| DISA STIG References | 2 |

**Table 15:** Most common references used in Specific Prompt

Overall, the Specific prompt provided the strongest results across all three test cases, demonstrating the benefits of supplying authoritative benchmarks within the prompt. While detection accuracy improved significantly, persistent inconsistencies and reliance on external references underscore the limitations of current LLMs in fully adhering to strict compliance requirements.

## 4.3 Cross-Reference Analysis with Other Studies

The findings of this study align closely with those reported by Sare and Debono [21], particularly in the effect that benchmark guidance has on LLM performance. In this research, the Broad prompt produced limited accuracy, with an overall PP Score of 28% for misconfiguration detection, whereas accuracy increased substantially when benchmark excerpts were introduced in the Specific case, reaching 61%. A similar trend was documented by Sare and Debono, who reported that “GPT-4’s response accuracy rate is 75%. . Their work also demonstrated that baseline zero-shot prompting without CIS reference produced accuracy rated of only 40.8% for GPT-4 and 26.3% for GPT-3.5, highlighting the same challenge observed in this study, where unguided prompts struggled to consistently detect benchmark-aligned errors [21]. Furthermore, Sare and Debono observed that when prompts explicitly referenced CIS benchmarks, performance in their “Cross-reference with CIS Benchmarks” category improved by around 10%[21], which is consistent with the improvements noted here when using benchmark-guided prompts. Together, these parallels reinforce that while LLMs demonstrate some capacity for detecting vulnerabilities under general prompts, their accuracy is markedly enhanced when grounded in explicit compliance standards.

Along with Sare’s study, the findings of Cao et al. [16] further support this dissertation’s argument that the level of guidance in prompt design directly affects how well LLMs perform in fault detection. Cao et al. observed that when ChatGPT was tasked with identifying faulty programs using a minimal prompt template, it successfully detected only 79% of cases, However, when supplied with a more informative and structured prompt, performance improved dramatically, reaching 100% detection rate. A similar pattern emerged in this dissertation’s evaluation of Cisco IOS configurations. Under the unguided Broad prompt, GPT identified 28% of misconfigurations. Introduction explicit references to CIS Benchmarks in the Mid prompt improved this slightly to 33%. The best results were achieved under the Specific prompt, which provided authoritative benchmark excerpts and narrowed the scope to a single protocol, allowing the model to identify 61% of misconfigurations. It was also reported that a similar pattern regarding the effect of prompt design on LLM performance was shown. It is shown that a basic, minimally guided prompt led ChatGPT to miss a meaningful portion of true errors and sometimes emphasized non-critical issues, whereas providing clearer task intention and context substantially improved performance and shifted the model towards a more functionally relevant direction. These results are consistent with this dissertations results. Unguided broad prompts tended to produce shorter, list-like outputs and occasional false positives, while benchmark-guided, protocol-scoped prompting yielded more structured compliance reasoning and higher accuracy in identifying materially important configuration issues.

The variability and inconsistencies observed in our three test cases align closely with results reported by Sobania et al. [4] on automatic program repair. In our setting, the models identified 28% misconfigurations under the Broad test 33% under the Mid test and 61% under the Specific, CIS-guided test In their benchmark, ChatGPT solved 48% of buggy programs at baseline, a level comparable to other LLMs tested which are CoCoNut (48%) and slightly below Codex (53%), while traditional APR baselines solved 18% under a stricter generalization check, indicating that large portions of true faults remained unsolved without additional guidance [4]. Taken together, the specifics from both studies point to the same grading methods, without precise, task-oriented guidance, LLMs yield unstable and incomplete results (ChatGPT 48% for Sobania et al. and ours 28% under Broad), whereas clearer objectives and structured constraints materially improve out comes (ours 61% under Specific), with Sobania et al. further showing that targeted follow-up hints can raise ChatGPT’s solved set to 78% when additional problem information is supplied.

**Figure 7**: Comparison of this study’s Scores and cross-referenced research scores

## 4.4 Overall Test Cases Assessment

|  |  |
| --- | --- |
| **Test Case** | **PP Score** |
| Broad | 28% |
| Mid | 33% |
| Specific | 61% |

Across all three test cases, a clear progression was observed in the model’s ability to detect benchmark-aligned errors, with accuracy increasing as prompts became more specific and guidance was made more explicit. The Broad prompt identified 33 out of 120 misconfigurations (28%), the Mid prompt rose slightly to 39 out of 120 (33%), while the Specific prompt achieved 73 out of 120 (61%). This steady improvement demonstrated that prompt engineering plays a pivotal role in shaping the reasoning process of large language models, validating the central hypothesis of this study that the precision and structure of prompts directly influence compliance assessment performance.**Table 16:** Results Summary

At the protocol level, results highlighted persistent patterns of strength and weakness. OSPF consistently achieved the highest scores across all prompt types, suggesting that the models possess stronger embedded knowledge and rule application capacity for this protocol compared to others. By contract, EIGRP and RIP were the weakest domains throughout, with frequent failures to detect critical commands. These discrepancies underscore the uneven distribution of protocol knowledge within the models and highlight the importance of scope-specific evaluation.

As the prompts progressed, the model’s outputs shifted from short, vague and list-like responses to longer, compliance-oriented explanations and finally to structured checklist-like reasoning anchored in CIS standards. However, even when benchmark excerpts were explicitly attached in the Specific case, the models continued to reference external sources, including Cisco documentation, NIST guidelines and real-world practices. This behaviour indicated that while LLMs can be directed toward authoritative standards, they cannot be fully constrained to them, raising concerns for professional use cases where strict adherence is a must.

Taken together, these findings demonstrate both the potential and the limitations of LLMs in network configuration auditing. Prompt engineering was shown to markedly improve accuracy, yet detection remained inconsistent and heavily protocol-dependent, with many critical requirements still overlooked. While the Specific prompt achieved the highest accuracy, its 61% detection rate remains insufficient for production environments where full compliance is non-negotiable. These outcomes confirm that LLMs cannot yet replace formal auditing tools, but they do provide evidence that with carefully constructed prompts and controlled datasets, they can be leveraged as support tools in compliance checking workflows.

## References

[4] D. Sobania, M. Briesch, C. Hanna, and J. Petke, “An Analysis of the Automatic Bug Fixing Performance of ChatGPT,” in *2023 IEEE/ACM International Workshop on Automated Program Repair (APR)*, May 2023, pp. 23–30. doi: [10.1109/APR59189.2023.00012](https://doi.org/10.1109/APR59189.2023.00012).

[16] J. Cao, M. Li, M. Wen, and S. Cheung, ‘A study on Prompt Design, Advantages and Limitations of ChatGPT for Deep Learning Program Repair’, Apr. 17, 2023, *arXiv*: arXiv:2304.08191. doi: [10.48550/arXiv.2304.08191](https://doi.org/10.48550/arXiv.2304.08191).

[21] A. Sare and D. Debono, “The Dual-Edged Sword: The Impact of Large Language Models in Network Infrastructure Security,” Institute of Information and Communication Technology, Malta College of Arts, Science and Technology, 2025. Accessed: Aug. 30, 2025. [Online]. Available: <https://www.scitepress.org/PersonProfile.aspx?PersonAccountID=SPMqmL3tsi8=&t=1>

‌