

NIST Internal Report
NIST IR 8579 ipd

Developing the NCCoE Chatbot

*Technical and Security Learnings from the Initial
Implementation*

Initial Public Draft

Harold Booth
Nakia Grayson
Michael Dimond
Michael Ekstrom
John Kent
Daniel Lee
Jasmine O'Hannon
Shahzad Rajput
Ken Sandlin
Theresa Suloway
Kevin Wacome

This publication is available free of charge from:
<https://doi.org/10.6028/NIST.IR.8579.ipd>

NIST Internal Report
NIST IR 8579 ipd

Developing the NCCoE Chatbot

Technical and Security Learnings from the Initial Implementation

Initial Public Draft

Harold Booth
*Computer Security Division
Information Technology Laboratory*

Nakia Grayson
*Applied Cybersecurity Division
Information Technology Laboratory*

Michael Dimond
Michael Ekstrom
John Kent
Daniel Lee
Jasmine O'Hannon
Shahzad Rajput*
Ken Sandlin
Theresa Suloway
Kevin Wacome*
The MITRE Corporation

**Former employee; all work for this publication was done while at employer.*

This publication is available free of charge from:
<https://doi.org/10.6028/NIST.IR.8579.ipd>

June 2025



U.S. Department of Commerce
Howard Lutnick, Secretary

National Institute of Standards and Technology
Craig Burkhardt, Acting Under Secretary of Commerce for Standards and Technology and Acting NIST Director

Certain equipment, instruments, software, or materials, commercial or non-commercial, are identified in this paper in order to specify the experimental procedure adequately. Such identification does not imply recommendation or endorsement of any product or service by NIST, nor does it imply that the materials or equipment identified are necessarily the best available for the purpose.

There may be references in this publication to other publications currently under development by NIST in accordance with its assigned statutory responsibilities. The information in this publication, including concepts and methodologies, may be used by federal agencies even before the completion of such companion publications. Thus, until each publication is completed, current requirements, guidelines, and procedures, where they exist, remain operative. For planning and transition purposes, federal agencies may wish to closely follow the development of these new publications by NIST.

Organizations are encouraged to review all draft publications during public comment periods and provide feedback to NIST. Many NIST cybersecurity publications, other than the ones noted above, are available at <https://csrc.nist.gov/publications>.

NIST Technical Series Policies

[Copyright, Use, and Licensing Statements](#)

[NIST Technical Series Publication Identifier Syntax](#)

How to Cite this NIST Technical Series Publication

Booth H, Grayson N, Dimond M, Ekstrom M, Kent J, Lee D, O'Hannon J, Rajput S, Sandlin K, Suloway T, Wacome K (2025) Developing the NCCoE Chatbot. (National Institute of Standards and Technology, Gaithersburg, MD), NIST IR (Internal Report) NIST IR 8579 ipd. <https://doi.org/10.6028/NIST.IR.8579.ipd>

Author ORCID iDs

Harold Booth: 0000-0003-0373-6219

Nakia Grayson: 0000-0003-1062-4338

Michael Dimond: 0009-0009-6775-4572

Michael Ekstrom: 0009-0007-1205-876X

John Kent: 0009-0001-9989-2277

Daniel Lee: 0009-0000-4662-6744

Jasmine O'Hannon: 0009-0000-5557-3218

Ken Sandlin: 0009-0000-3757-4858

Theresa Suloway: 0009-0007-5164-5182

Kevin Wacome: 0009-0007-7047-1341

Public Comment Period

June 18, 2025 – August 4, 2025

Submit Comments

nlp-nccoe@nist.gov

National Institute of Standards and Technology

Attn: Applied Cybersecurity Division, Information Technology Laboratory

100 Bureau Drive (Mail Stop 2000) Gaithersburg, MD 20899-2000

Additional Information

Additional information about this publication is available at <https://csrc.nist.gov/pubs/ir/8579/ipd>, including related content, potential updates, and document history.

All comments are subject to release under the Freedom of Information Act (FOIA).

Abstract

Chatbots are emerging as alternative interfaces for structured information retrieval and internal knowledge access. Chatbots can utilize the capabilities of large language models (LLMs) to help interpret user-provided input and provide responses to a variety of requests. This paper describes the development of an LLM chatbot by the National Cybersecurity Center of Excellence (NCCoE) at NIST to enable internal search across its published cybersecurity guidance. The paper provides a point-in-time examination of the tool's development process, including the architecture, the system configuration, and the NCCoE's approach to addressing cybersecurity challenges throughout the design and deployment lifecycle. Specific attention is given to threats such as prompt injection, hallucinations, data exposure, and unauthorized access. The paper also discusses the mitigations applied, including local deployment, access controls, and validation filters. This paper is not intended to serve as implementation guidance. Instead, it documents technical decisions, observed limitations, and risk-informed safeguards that shaped the prototype. It provides an overview of the chatbot and its supporting technologies so that other organizations might consider the benefits of their use.

Keywords

artificial intelligence (AI); chatbot; cybersecurity; large language model (LLM); machine learning; retrieval-augmented generation (RAG).

Reports on Computer Systems Technology

The Information Technology Laboratory (ITL) at the National Institute of Standards and Technology (NIST) promotes the U.S. economy and public welfare by providing technical leadership for the Nation's measurement and standards infrastructure. ITL develops tests, test methods, reference data, proof of concept implementations, and technical analyses to advance the development and productive use of information technology. ITL's responsibilities include the development of management, administrative, technical, and physical standards and guidelines for the cost-effective security and privacy of other than national security-related information in federal information systems.

Note to Reviewers

We are seeking feedback on the following:

1. What additional information would be useful for you in understanding and applying the technologies discussed in this paper?
2. How do you expect to use these technologies in future practices and processes?
Alternatively, how have you used these or related technologies in your organization?
3. What additional applications of machine learning technologies would you like to see the NCCoE demonstrate?

Call for Patent Claims

This public review includes a call for information on essential patent claims (claims whose use would be required for compliance with the guidance or requirements in this Information Technology Laboratory (ITL) draft publication). Such guidance and/or requirements may be directly stated in this ITL Publication or by reference to another publication. This call also includes disclosure, where known, of the existence of pending U.S. or foreign patent applications relating to this ITL draft publication and of any relevant unexpired U.S. or foreign patents.

ITL may require from the patent holder, or a party authorized to make assurances on its behalf, in written or electronic form, either:

- a) assurance in the form of a general disclaimer to the effect that such party does not hold and does not currently intend holding any essential patent claim(s); or
- b) assurance that a license to such essential patent claim(s) will be made available to applicants desiring to utilize the license for the purpose of complying with the guidance or requirements in this ITL draft publication either:
 - i. under reasonable terms and conditions that are demonstrably free of any unfair discrimination; or
 - ii. without compensation and under reasonable terms and conditions that are demonstrably free of any unfair discrimination.

Such assurance shall indicate that the patent holder (or third party authorized to make assurances on its behalf) will include in any documents transferring ownership of patents subject to the assurance, provisions sufficient to ensure that the commitments in the assurance are binding on the transferee, and that the transferee will similarly include appropriate provisions in the event of future transfers with the goal of binding each successor-in-interest.

The assurance shall also indicate that it is intended to be binding on successors-in-interest regardless of whether such provisions are included in the relevant transfer documents.

Such statements should be addressed to: nlp-nccoe@nist.gov

63	Table of Contents	
64	1. Introduction.....	1
65	1.1. Project Overview.....	1
66	1.2. Related Work	2
67	2. Retrieval-Augmented Generation Technical Details.....	4
68	2.1. The Foundation Model.....	4
69	2.2. Preprocessing External Data	4
70	2.3. Creating an Index	4
71	2.4. Retrieving Relevant Information.....	5
72	2.5. Querying the LLM.....	5
73	3. Implementation Details and Considerations.....	7
74	3.1. Chatbot Configuration.....	7
75	3.1.1. Virtual/Physical Environment and Configuration.....	7
76	3.1.2. Preprocessing and Page-Level Citations.....	7
77	3.1.3. The Software Development Framework	7
78	3.1.4. The Embedding Function and Vector Database	8
79	3.1.5. The Foundation Model	8
80	3.2. Risk Mitigation and Discussion Around Threat Analysis	8
81	3.2.1. Hallucinations	8
82	3.2.2. Threat Analysis	9
83	3.3. Deployment.....	9
84	3.4. Testing and Evaluation	9
85	3.5. Risks and Limitations.....	10
86	4. Comparison Against COTS Tools.....	11
87	5. Future Considerations	13
88	5.1. Future Considerations for Testing and Evaluation.....	13
89	References.....	14
90	Appendix A. List of Symbols, Abbreviations, and Acronyms.....	15
91	List of Tables	
92	Table 1. Service Responses to Test Question	12
93	List of Figures	
94	Figure 1. NIST 1800-30C Active Scan Instructions	11

95 **Acknowledgments**

96 The National Institute of Standards and Technology (NIST) and the National Cybersecurity
97 Center of Excellence (NCCoE) would like to thank Cherilyn Pascoe of the NIST Applied
98 Cybersecurity Division; Alden Dima, Timothy Blattner, Guillaume Sousa Amaral, and Derek Juba
99 of the NIST Software and Systems Division; Ian Soboroff of the NIST Information Access
100 Division; and Michael Sharp of the NIST Smart Connected Systems Division for their
101 contributions to the paper and the ongoing work.

1. Introduction

In the digital age, internet search engines have been indispensable tools for finding information. They offer a vast array of data from across the internet, allowing users to access a wide range of topics with ease. Search engines primarily rely on keyword matching and ranking algorithms, which can sometimes lead to irrelevant or overwhelming results [\[1\]](#). Additionally, search engines provide a highlighted text blurb containing a potential answer and lead the reader to the highlighted document. The searcher is then left with the task of discovering the context of the answer within the document. In contrast, LLM-based chatbots can potentially provide more contextually relevant and precise responses by understanding the context of natural language queries, summarizing the results of a query, and potentially synthesizing a response from multiple documents.

LLMs are trained on extensive datasets, allowing them to generate human-like text responses. A consideration when using LLMs is that they typically have a knowledge cutoff date, largely due to the amount of time and resources needed for training, meaning they cannot answer questions about events occurring beyond that point. Additionally, LLMs may experience hallucinations, where they generate information that is incorrect and not based on the provided data. When interacting with LLMs, in addition to the intrinsic risk of hallucinations, security concerns such as prompt injection, data exposure, and unauthorized access should be considered.

Another limitation of LLMs is their inability to access or respond to questions about information contained within private document collections. Depending on the provider, interactions with some commercial LLMs may be stored and used for future analysis or training, raising privacy concerns.

1.1. Project Overview

The National Cybersecurity Center of Excellence (NCCoE), part of the National Institute of Standards and Technology (NIST), is a collaborative hub that brings together experts from industry, government, and academia to tackle real-world cybersecurity challenges.

Large language model (LLM) technology has shown promise in the development of specialized tools that could potentially increase the speed of certain tasks. To further enhance its mission, the NCCoE identified a potential application for a secure, internal-use chatbot. The development of the NCCoE chatbot aims to assist users in discovering and summarizing cybersecurity guidelines specific to their needs.

This was a significant motivation for the NCCoE when choosing to implement a locally available chatbot.

To mitigate these challenges, the NCCoE opted for a local installation of an LLM, ensuring that all interactions with the LLM remain within their secure network. To obtain more accurate responses from the chatbot regarding questions about NCCoE publications, the team employed RAG technology.

The chatbot was built using retrieval-augmented generation (RAG)-based LLM technology. This approach combines techniques from information retrieval and natural language generation, enabling the chatbot to provide more focused, contextually relevant responses by leveraging a repository of cybersecurity knowledge. Specifically, the chatbot is currently designed to search NIST publications exclusively, enabling users to receive information that is aligned with the NCCoE's guidelines and best practice documents.

RAG combines information retrieval with natural language generation, allowing the chatbot to search through a specific repository of documents and generate responses based on the retrieved information. This approach ensures that users receive relevant answers aligned with NCCoE's guidelines and best practice documents.

Moreover, the NCCoE chatbot has been customized to provide page-level citations for the source documents from which its responses are drawn. This feature not only allows for verification of the information provided but also enables users to explore the original publications for a deeper understanding. By adopting these and the approaches described below, the NCCoE chatbot offers an internal tool that is at the appropriate level of security, efficiency, and reliability risk for this context.

In addition to leveraging RAG, the NCCoE has taken measures to mitigate potential vulnerabilities such as prompt injections, which can manipulate the chatbot into generating unintended responses. To counteract this, the chatbot is designed with robust input validation and filtering mechanisms to provide guardrails around user inputs and responses. These precautions help maintain the integrity and reliability of the chatbot's interactions, safeguarding against potential misuse or exploitation.

The implications of using a RAG-supported chatbot is that users will be able to more rapidly find actionable cybersecurity guidelines. NCCoE publications contain large amounts of cybersecurity guidelines as well as precise details for their implementation. Using a RAG in this way will increase the searchability of cybersecurity guidelines for both NCCoE researchers as well as the wider community. Increasing the accessibility of organizational guidelines can be helpful to internal users, as it allows them to maintain awareness of past work when developing future guidelines. This has the potential to lead to more consistent documents, as well as more efficient use of workers' time.

Utilizing LLMs to generate content comes with the risk of misleading or fabricated information. For an internal audience, this risk is minimal. It is assumed that NIST researchers will be able to identify issues with the provided output, especially given the included citation to the relevant section of a document. However, there still exists the chance that such tools can mislead the user, especially when expanding access to a larger audience.

1.2. Related Work

Using a RAG in support of a chatbot is applicable across many use cases. These technologies have been utilized to support efforts such as Security Operations Center workflows [\[2\]](#), as just one example. The NCCoE's use case specifically considers the need to navigate a body of technical publications.

180 Other work on LLMs at NIST has conducted research in the manufacturing space to evaluate
181 bodies of literature to discover changes to relative topics over time [\[3\]](#). Furthermore, NIST is
182 actively considering the usage of AI tools in the field of information retrieval through the Text
183 Retrieval Conference [\[4\]](#).

184 Research has also been done by the National Library of Medicine at the National Institutes of
185 Health to evaluate improvements in searching for health-related information [\[5\]](#). This serves as
186 another example of how LLMs can be leveraged to assist in retrieving information from a broad
187 dataset. For such a use case, there is additional importance placed on the accuracy and
188 relevancy of responses.

189 There are specific considerations and implications for each use case of these technologies.

2. Retrieval-Augmented Generation Technical Details

Retrieval-augmented generation (RAG) can be used when context is needed that is not contained in the pre-trained foundation large language model (LLM). RAG allows the introduction of information like internal organization documents that was not part of the training of the foundation LLM.

The selection of the foundation model informs the available options for configuring a RAG-based pipeline. Setting up a RAG-based pipeline then involves several key steps: preprocessing external data, creating an index from external data, retrieving relevant information based on the user's information need, querying the LLM with a prompt containing the user's query and the relevant information retrieved in the previous step as a context.

Since all our documents were PDF documents containing text, we focus only on the case where the external data is restricted to text documents.

2.1. The Foundation Model

Which foundation model to use is another variable in the RAG-based pipeline, as it determines the quality and cost of the final response. The model must be carefully evaluated for features like its complexity, speed, and maximum context window supported. This will dictate the acceptable parameters that can be used by the RAG.

A model with lower parameters is generally limited in its ability to generate a quality response. However, complex, higher-parameter foundation models may require a much powerful GPU and may be comparatively slower to generate a response.

Another factor to consider is the maximum context window supported by the model. This becomes of particular interest if the foundation model is a shared resource and is being provided as a service. In which case, the options provided need to not be evaluated by the value of the maximum context window the foundation model supports. Rather, it should be evaluated by the maximum context window offered by the service, since the service provider may choose to use context window size much lower than what the model supports.

Note that the same model when using a larger context window may be slower in generating its response compared to when using a smaller context window.

2.2. Preprocessing External Data

First, metadata such as the filename and authors, along with the clean text, is extracted from each PDF file. This metadata and text are then converted into a JSON object. Finally, each document's JSON object is stored on a separate line in a JSONL file.

2.3. Creating an Index

The JSONL file generated as part of the preprocessing step, containing extracted metadata and clean text from external source documents, is then used for chunking and indexing. Each

document is first split into smaller chunks which are then transformed into corresponding vector representations using an embedding function. These vector embeddings are used to represent meaning such that words closer in meaning should be closer in the vector spaces as compared to words representing more different meanings. A vector embedding is generally represented as a list of numbers, and the number of values in this list is referred to as *dimension*. Vectors with higher dimensions capture more information as compared to those with lower dimensions, but higher-dimensional vectors are harder to work with. There are several choices for embedding functions, and the one used dictates the maximum size of the chunks. Chunks from every document are converted to a vector representation using an embedding function, and these vectors are stored in the vector database. This process is called *indexing*.

There are several options available when choosing a vector database depending on the features offered. One of these features to consider is the availability of support for metadata tags and operations supported over these tags.

2.4. Retrieving Relevant Information

Vector databases help with mathematical operations over high-dimensional vectors. Most important among these operations is the search function, which is used to find vectors the most similar to a query vector. This comes in handy when one is interested in finding chunks of documents semantically the most similar to a given user query.

Upon receiving a query from the user, the embedding function is used, this time to convert the user query into a corresponding vector representation. The vector representation of the user query is then matched against all vectors, representing document chunks already in the vector database, to return a ranked list of chunks. The top chunk in the ranked list is the one that is the closest to the query in a semantic sense. Note that it is therefore important that the user's query is a sufficient representation of the information need.

Retrieval is the key component of RAG-based pipelines. A set number of the most closely matching chunks are selected. This process is called *top-k*.

2.5. Querying the LLM

The LLM processes each of the top-k chunks individually to respond to the user's query, using only that specific chunk as context. A key difference exists between how the first chunk and subsequent chunks are handled. When the first chunk is presented to the LLM, it responds to the user's query based solely on the content of that chunk. For subsequent chunks, the LLM is given the new chunk along with its previous response and the user's query. It then refines its earlier response if necessary, considering the new context provided by the additional chunk. When the LLM has seen all the top-k chunks, the answer given by the LLM is finally shown to the user.

It is important to note that when a RAG-based pipeline fails to respond to a user question, it is mostly due to retrieval failures. If the LLM is asked to restrict itself to a given context and to not

263 use any knowledge given at pretraining, but the context doesn't contain an answer, the LLM
264 stands no chance of giving a correct answer. This is why retrieval plays a key role in the success
265 of a RAG-based pipeline. Therefore, selecting the right value for top-k is critical.

3. Implementation Details and Considerations

Disclaimer: The latest technologies and best practices in the AI space are constantly evolving. The tools and technologies discussed below represent the latest deployment of the NCCoE chatbot as of April 2025. When implementing a solution, please look for the latest versions of software, as well as for advancements in AI models and supporting software.

3.1. Chatbot Configuration

3.1.1. Virtual/Physical Environment and Configuration

The NCCoE chatbot operates on a single Nvidia Deep GPU Xceleration (DGX) station, equipped with 4x Tesla V100 GPUs, providing the necessary computational power for running large-scale AI models.

The system runs on Ubuntu 20.04.6 LTS, utilizing a Python 3.12.4 virtual environment for dependency management. The chatbot leverages Ollama 0.3.6, an open-source tool for easily and efficiently running state-of-the-art LLMs locally, ensuring a controlled and reproducible setup. Ollama is straightforward to set up on most hardware, but other technologies such as SGLang or vLLM can be considered to better support multiple users.

The chatbot is hosted within the NCCoE AI Lab network, allowing access to NCCoE engineers for both testing and business use. Requests are routed through an NGINX reverse proxy, which forwards traffic to the chatbot's front-end UI, ensuring efficient request handling.

3.1.2. Preprocessing and Page-Level Citations

In order to support page-level citations, each PDF file is first split into pages, and each page is then stored as a separate JSON structure in the JSONL file for indexing. Each JSON structure is augmented to contain page-specific metadata along with clean text from the corresponding page. This page-specific metadata contains filename, URL, and page number. In the future, we plan to include the document title, document number, and other information in the metadata which will ultimately be used to build an auto-retrieval based chatbot [\[6\]](#).

3.1.3. The Software Development Framework

There are a number of software development frameworks that can be used to help speed up the process of building LLM-based applications, including LlamaIndex and LangChain [\[7\]\[8\]](#). LlamaIndex offers an easy-to-use setup for RAG. It provides flexible indexing methods suitable for various data types and structures, whereas LangChain offers high flexibility in customizing every aspect of the RAG pipeline but building a fully functional RAG pipeline with LangChain could be more complex compared to LlamaIndex. We chose LlamaIndex due to prior experience.

3.1.4. The Embedding Function and Vector Database

There are numerous embedding models available, primarily categorized into open-source and closed-source [9]. We chose open-source models to address privacy concerns. Among these, we selected models based on sentence-transformers, given that our queries were expected to be short. We were also limited to those embedding models that were supported by LlamaIndex. From the available options, we chose the all-mpnet-base-v2 model for its good performance, manageable size, and optimal embedding dimensions [10]. The selected embedding model maps sentences and paragraphs into a 768-dimensional dense vector space. These dimensions are neither too large to hinder performance on our hardware nor too small to compromise the complexity of concepts in the text. We set the max chunk size to 512 as dictated by the embedding model selected.

There are several vector databases to choose from, including but not limited to Chroma, Pinecone, Weaviate, Faiss, and Qdrant. We selected Chroma due to the features it offers. For this iteration of the chatbot, we used the top-k value of 3 as the optimal value for our use case.

3.1.5. The Foundation Model

Commercial LLMs may store any interactions with them and use them for future analysis or training, raising privacy concerns, which is why using a commercial LLM was not an option for us. Additionally, the expense associated with utilizing commercial LLMs was a significant concern. We chose Meta's open-source Llama models due to its performance and attempted to run its largest available model, Llama 3.1 with 405B parameters, but found that our Tesla V100 GPUs, with 64GB memory, were not powerful enough for this model to be able to respond in a timely manner. For this reason, Llama 3.1 with 70B parameters is presently being used as the foundation model. Note that a new version of Llama, Llama 3.3 with 70B parameters, was recently released, which claims to be on par with Llama 3.1 405B in terms of performance; we plan to switch to this model in the upcoming weeks.

3.2. Risk Mitigation and Discussion Around Threat Analysis

3.2.1. Hallucinations

When an LLM is asked to answer a user query restricting the response to a given context, the LLM may respond with, "Answer not found in the context of NCCoE Documents" if there were no answers to the question in the context. However, it may still respond with a made-up answer, even when answers to the question are found in the context. When it generates a made-up response, we refer to it as *hallucinating*. To make sure the responses displayed to the user are legitimate, we run the final response through a filter to make sure the response is supported by the document chunks seen by the LLM. The foundation model is used as this filter where the LLM is asked to determine if the response previously generated by the RAG-based chatbot is supported by the text provided to it as part of the context. The user is not shown this response if it is determined to be hallucinated.

3.2.2. Threat Analysis

Threat analysis for a RAG-based LLM application involves systematically identifying and evaluating potential security risks that could compromise the system's integrity, confidentiality, and availability. Given the hybrid nature of RAG systems, which combine LLMs with external data retrieval mechanisms, threats can arise from multiple vectors. Key concerns include data poisoning, where malicious actors manipulate the external data sources to influence the model's outputs adversely. Additionally, there is the risk of adversarial attacks that exploit vulnerabilities in the LLM's architecture. Unauthorized access to the system's API or data repositories could lead to data breaches or misuse of sensitive information.

To mitigate these risks, it is crucial to implement robust authentication protocols, continuous monitoring for anomalous activities, and regular updates to both the LLM and its data sources. Furthermore, incorporating explainability and transparency in the model's decision-making process can help in identifying and addressing potential inaccuracies in the generated content. It is also important to utilize trusted models. We are using open-source models downloaded directly from the producer of the model. The input and output are sanitized and filtered in the same way we mitigate hallucinations, as mentioned above, to reduce the likelihood of any successful prompt injection attack. In addition, only trusted users on the VPN can access the chatbot to manage unauthorized access risk. We are considering expanding the logging of security events in the next phase.

3.3. Deployment

The end-to-end chatbot application was deployed on an NCCoE server running Ubuntu and restricted to the internal network. Nginx is used for HTTPS port forwarding, and the interface of the application was embedded using iFrame into a development instance of the NCCoE main webpage. The use of the chatbot was restricted to trusted users as opposed to the public to mitigate security concerns relating to various kinds of attacks.

3.4. Testing and Evaluation

We have conducted a small study to get a sense of the quality of responses generated by the NCCoE chatbot to guide us to improvements. The first part of this study entailed development of about 100 questions and corresponding ground truth answers. Due to time constraints, these 100 questions and corresponding answers were manually selected and further revised from a larger set of LLM-generated question-answer pairs. As part of this exercise, we found that the LLM-generated question-answer pairs were largely not up to the mark mainly due to limited specificity in the questions. We plan to revise the prompts in the future to help the LLM generate quality question-answer pairs in the next phase.

In the second part of this study, we used the chatbot to answer these 100 questions. These responses were then automatically assessed for correctness against ground truth using the evaluation tool Ragas which uses an LLM-as-a-Judge rather than human assessors [\[11\]](#). Only a very small subset was assessed manually due to limited resources. The Pearson product-

moment correlation coefficients between manual and automatic assessments of 0.7872 indicates a linear relation between the two assessments; however, human assessment for a much larger set is needed to draw any conclusion with statistical confidence. Failure analysis not only helped us with making informed decisions around the improvements in the chatbot, but it also gave us insights into the trustworthiness of using an automated tool like Ragas to replace human assessors. We found that Ragas frequently gave us a score of 0.2 for empty responses. This led us to conclude that automatic assessment systems can be useful for development purposes, but they cannot replace rigorous human assessment at this time.

3.5. Risks and Limitations

At this prototype stage, the NCCoE chatbot is configured as a question-answering bot. It lacks the ability to hold conversations with the user. We plan to address this in the next phase.

Additionally, despite the mitigations in place, there is still a chance that the response generated by the RAG-based model is a hallucinated one. Continued evaluation of the tool is needed to understand the level of this risk.

Furthermore, since we are not employing auto-retrieval, the chatbot's ability to answer some metadata-specific questions, like names of the authors of a publication, is limited. Active work is being conducted to address this limitation.

Another limitation of our chatbot is in responding to questions that touch upon answers found in the images contained in the PDF document. Addressing this limitation is considered out of scope of this work at present.

4. Comparison Against COTS Tools

Following is an example that highlights the benefits of using a RAG-based chatbot compared to commercial off-the-shelf (COTS) tools. We ran the following question against our chatbot, Meta AI, and OpenAI: “What are the steps involved in executing active scans as described in NIST SP 1800-30C: Securing Telehealth Remote Patient Monitoring Ecosystem?”

The screen-capture of the correct answer is shown in Fig. 1.

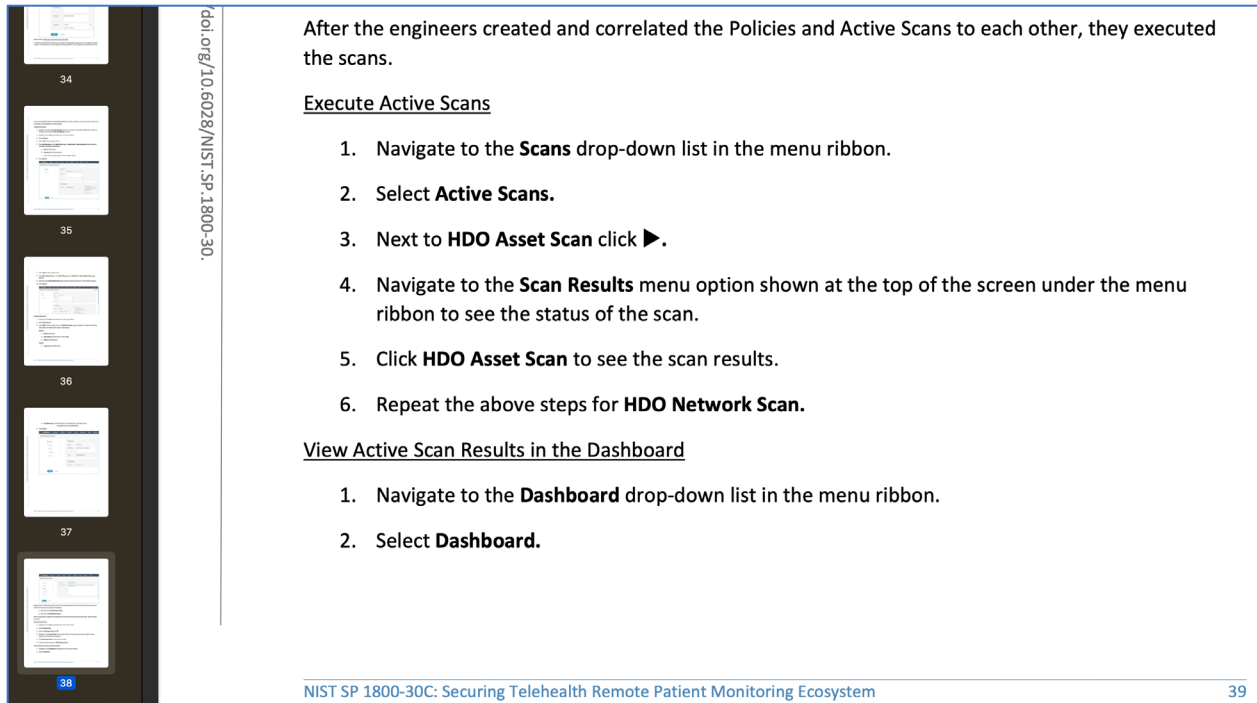


Figure 1. NIST 1800-30C Active Scan Instructions

Table 1 shows the results from asking the question to all three services.

404

Table 1. Service Responses to Test Question

NCCoE Chatbot (Llama)	Meta AI (Llama)	OpenAI (ChatGPT)
<p>To execute active scans as described in NIST SP 1800-30C: Security Telehealth Remote Patient Monitoring Ecosystems, follow these steps:</p> <ul style="list-style-type: none"> • Navigate to the Scans drop-down list in the menu ribbon • Select Active Scans • Next to HDD Asset Scan click. • Navigate to the Scan Results menu option shown at the top of the screen under the menu ribbon to see the status of the scan. • Click HDO Asset Scan to see the scan results. • Repeat the above steps for HDO Network Scan <p>Page number: 39 Source document URL: https://nccoe8fy2020dev.prod.acquia-sites.com/sites/default/files/2022-02/rpm-nist-sp1800-30c-final.pdf</p>	<p>Executing active scans as described in NIST SP 1800 30C involves several key steps:</p> <ul style="list-style-type: none"> • Configuring scans: This step involves setting up your scanning system to identify vulnerabilities in the telehealth remote patient monitoring ecosystem ^{1,2}. • Vulnerability Scanning: Use your configured scanning system to scan for endpoint vulnerabilities, including missing patches and outdated software ¹. • Linking a Scanner: Integrate a scanner, such as Nessus, into your vulnerability management system to streamline the scanning process ². • Repeating Scans: Regularly repeat scans, such as basic network scans, to ensure ongoing security and identify new vulnerabilities ³. • Analyzing Results: Review and analyze scan results to identify areas for improvement and implement necessary patches or updates. <p>By following these steps, you can effectively execute active scans and enhance the security of your telehealth remote patient monitoring ecosystem, as outlines in NIST SP 1800-30C.</p>	<p>Executing scans in accordance with NIST SP 1800-30C for securing the Telehealth Remote Patient Monitoring Ecosystem involves the following steps:</p> <ol style="list-style-type: none"> 1. Preparation: <ul style="list-style-type: none"> • Define the scope of the scan, including systems, networks, and devices to be tested. • Ensure that you have the necessary permissions to perform the scans to avoid legal issues. 2. Select Scanning Tools: <ul style="list-style-type: none"> • Choose appropriate active scanning tools that can identify vulnerabilities and misconfigurations to the telehealth environment. 3. Configuration of Scanning Tools: <ul style="list-style-type: none"> • Configure the scanning tools according to the specific requirements of the telehealth systems. • Set parameters such as scan depth, types of vulnerabilities to look for, and authentication methods if needed. 4. Conduct Scans: <ul style="list-style-type: none"> • Execute the active scans on the defined scope. • Monitor the process to address any issues that may arise during scanning (e.g. system performance impacts) 5. Analysis of Results: <ul style="list-style-type: none"> • Review the scan results to identify vulnerabilities and security issues. • Prioritize findings based on severity and potential impact on the telehealth ecosystem. 6. Reporting: <ul style="list-style-type: none"> • Document the findings in a comprehensive report, detailing vulnerabilities, affected systems, and suggested remediation steps. • Ensure the report is understandable for technical and non-technical stakeholders 7. Remediation: <ul style="list-style-type: none"> • Collaborate with relevant teams to address identified vulnerabilities. • Implement patches, configuration changes, or other mitigations as necessary. 8. Verification: <ul style="list-style-type: none"> • Conduct follow-up scans to verify that vulnerabilities have been effectively remediated. • Ensure that no new vulnerabilities were introduced during the remediation process. 9. Continuous Monitoring: <ul style="list-style-type: none"> • Establish a process for ongoing vulnerability scanning and assessment to maintain security in the telehealth environment. • By following these steps, organization can effectively execute active scans to secure their telehealth remote patient monitoring ecosystems. <p>By following these steps, organization can effectively execute active scans to secure their telehealth remote patient monitoring ecosystems.</p>

405 Our RAG-based chatbot provided an informed response and also provided the page-level
406 citation back to the source document.

5. Future Considerations

The following are future work areas being considered:

- Incorporate secure logging practices.
- Establish pipelines which monitor the logs continuously.
- Store common questions and answers to improve user interactions and performance on repeat questions.
- Implement tools for the LLM that better serve concurrent users.
- Upgrade the tool to hold a conversation with the user.
- Deploy the chatbot to a dedicated domain.
- Enhance the guardrail systems in place and separately perform rigorous validation to continue to address security concerns such as prompt injection [\[12\]](#).
- Continue to improve the user interface for a better user experience, potentially involving a user study to gauge chatbot usability.
- Conduct testing and evaluation before public release.

5.1. Future Considerations for Testing and Evaluation

Evaluating the proper function of an LLM is an area of active interest to the research community. As we continue to iterate on the chatbot, the testing and evaluation process will evolve as well. There are two research areas of particular interest as we move forward:

- **Perturbation testing** utilizes variations in input to test model performance. For instance, introducing typos or slight variations to the prompts that are given to an LLM can lead to variations in output. Given the expectations that a variety of users may use a chatbot, it is useful to analyze how these variations impact the performance of the tool. Furthermore, it is useful to understand how a model will perform when it is asked to conduct a task that is slightly outside of an area it is most reliable in, such as asking it slight variations to common logic puzzles.
- The concept of **topic modeling** may also be useful for informing further development of the chatbot. Topic modeling can be used to understand the primary pillars of the vector database that is utilized by the RAG. Utilizing this information, automated testing can be implemented to understand how far a query to the chat is from the central topic pillars. This can be used to inform quality control for responses, such as providing a disclaimer when a question strays outside of the most well-supported topics.

References

- [1] Karpukhin, V, Oguz, B, Min, S, Lewis, P, Wu, L, Edunov, S, Chen, D, & Yih, W-T (2020) Dense passage retrieval for open-domain question answering. *Conference on Empirical Methods in Natural Language Processing* (Association for Computational Linguistics, Online), pp 6769-6781 <https://aclanthology.org/2020.emnlp-main.550.pdf>
- [2] Paduraru C, Patilea C, Stefanescu A (2024) Cyberguardian: An Interactive Assistant for Cybersecurity Specialists Using Large Language Models. *19th International Conference on Software Technologies* (Science and Technology Publications, Dijon, France), pp 442-449. <https://www.scitepress.org/Papers/2024/128117/128117.pdf>
- [3] Sexton, R, Brundage, M, Dima, A, Sharp, M (2020) MSEC: A QUANTITATIVE RETROSPECTIVE. *MSEC: Manufacturing Science and Engineering Conference* (NIST, Cincinnati, Ohio, US), Available at <https://doi.org/10.1115/MSEC2020-8440>
- [4] National Institute of Standards and Technology (2025) *The Text Retrieval Conference*. Available at <https://trec.nist.gov/>
- [5] Demner-Fushman D, Mrabet Y, Abacha A (2020) Consumer health information and question answering: helping consumers find answers to their health-related information needs. *Journal of the American Medical Informatics Association* 27(2):194-201 <https://doi.org/10.1093/jamia/ocz152>
- [6] LlamaIndex (2025). *Auto-Retrieval from a Vector Database*. Available at https://docs.llamaindex.ai/en/stable/examples/vector_stores/chroma_auto_retriever.html
- [7] LlamaIndex (2025). *Welcome to LlamaIndex*. Available at <https://docs.llamaindex.ai/en/stable/>
- [8] LangChain (2025). *The platform for reliable agents*. Available at <https://www.langchain.com/>
- [9] Hugging Face (2025). *MTEB Leaderboard – a Hugging Face space by MTEB*. Available at <https://huggingface.co/spaces/mteb/leaderboard>
- [10] Hugging Face (2025). *sentence transformers/all-mpnet-base-v2*. Available at <https://huggingface.co/sentence-transformers/all-mpnet-base-v2>
- [11] Ragas (2025). *Introduction*. Available at <https://docs.ragas.io/en/latest/>
- [12] Debendetti E, Shumailov I, Fan T, Hayes J, Carlini N, Fabian D, Kern C, Shi C, Terzis A, Tramèr F (2025) Defeating Prompt Injections by Design. *arXiv preprint*. <https://arxiv.org/pdf/2503.18813v1>

471 **Appendix A. List of Symbols, Abbreviations, and Acronyms**

472 **AI**

473 Artificial Intelligence

474 **COTS**

475 Consumer Off-The-Shelf

476 **DGX**

477 Deep GPU Xceleration

478 **GPU**

479 Graphics Processing Unit

480 **JSON**

481 JavaScript Object Notation

482 **JSONL**

483 JavaScript Object Notation Lines

484 **LLM**

485 Large Language Model

486 **NCCoE**

487 National Cybersecurity Center of Excellence

488 **NIST**

489 National Institute of Standards and Technology

490 **RAG**

491 Retrieval-Augmented Generation

492 **SP**

493 Special Publication

494 **VPN**

495 Virtual Private Network