Network Security – Round 2

Network Security Tutor & Quiz Agent

Name: Hari Krishna Cherukuri

R#: R11978051

This system, titled Network Security Tutor & Quiz Agent, is designed to enhance users' understanding of core network security concepts through interactive learning and hands-on experimentation.
It operates entirely on a local machine, ensuring complete data privacy and zero external network interaction.

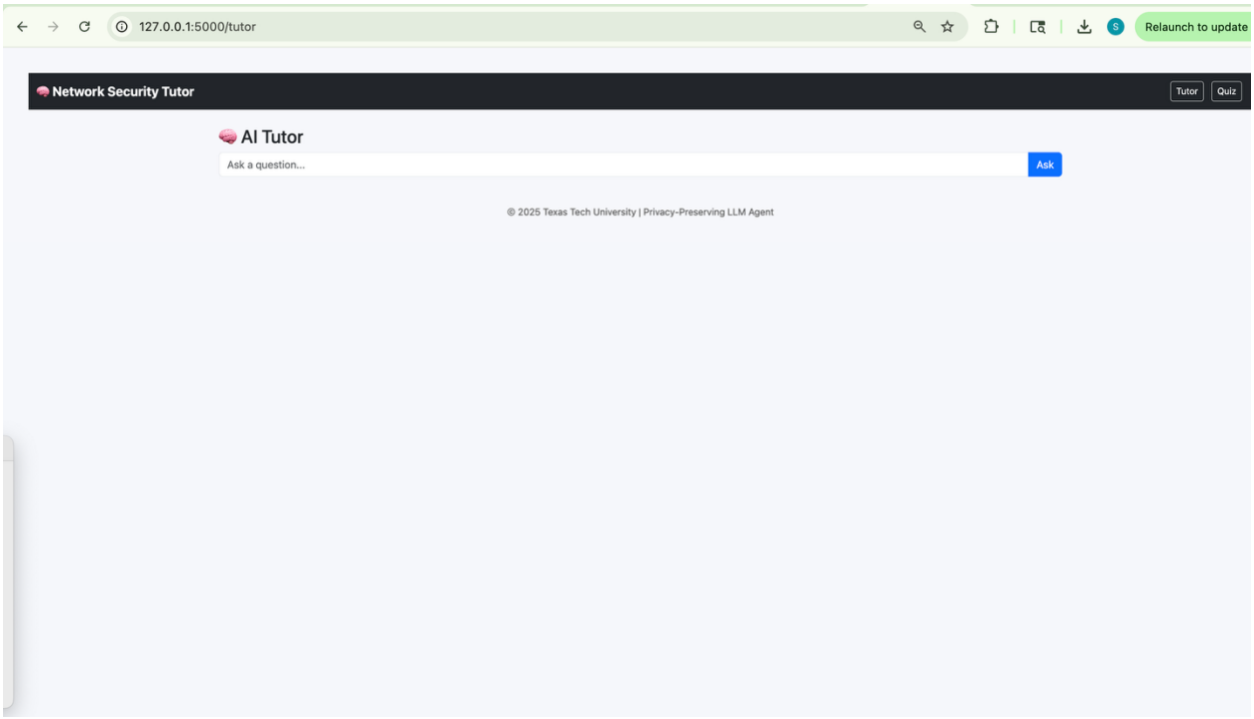
The system consists of two main components — the Network Security Tutor and the Quiz Agent.

- The Network Security Tutor retrieves relevant information from a Chroma vector database and uses a locally hosted Ollama large language model to generate accurate, context-aware responses.
- The Quiz Agent leverages the same model to dynamically create and evaluate quiz questions, allowing users to actively test and reinforce their understanding.

Users interact with the platform through a flask web interface (running on port 57235), while the backend logic is handled by a server (running on port 57246). When a user submits a question or quiz request, the frontend sends the query to the backend, which retrieves related data from the Chroma database and communicates with the Ollama model to generate the final response.

All communication between these components occurs locally over HTTP (127.0.0.1). This design ensures that every stage — from query submission to database retrieval to model inference and response delivery — remains securely contained within the host system.

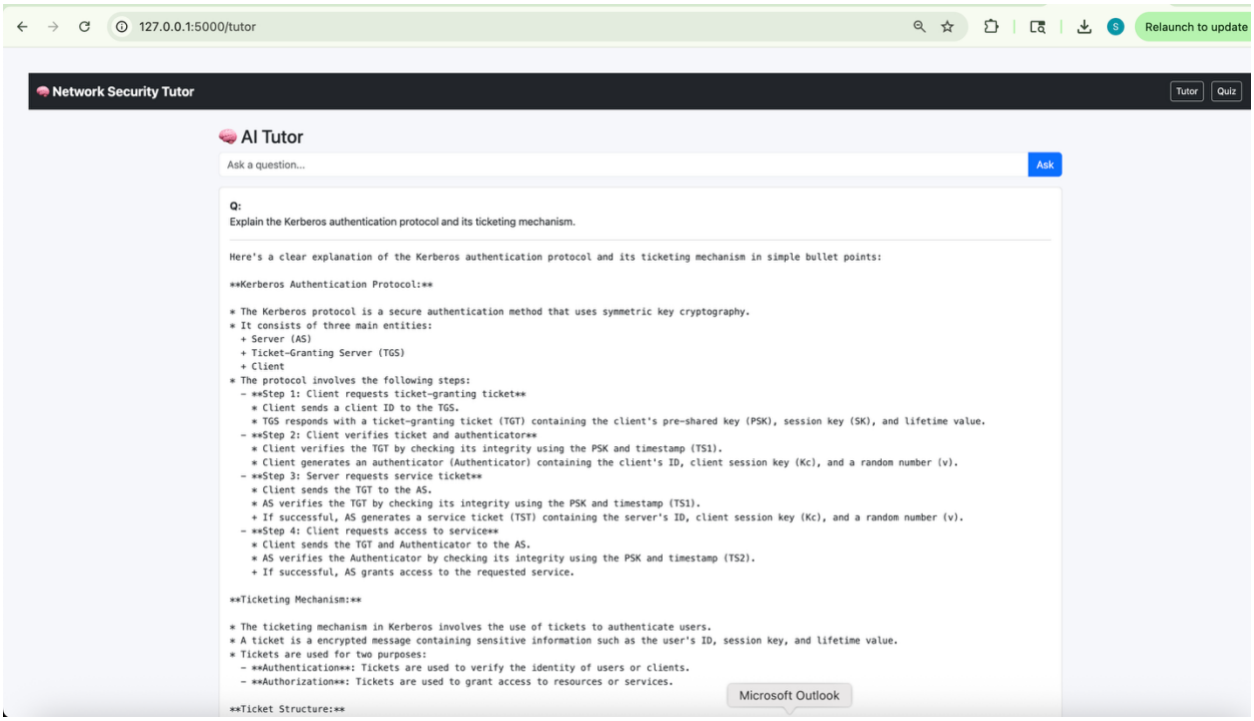
This report presents a Wireshark-based packet analysis of the system's operation, illustrating how data travels between the browser, backend, Chroma database, and Ollama model. By examining these packet exchanges, the report clearly demonstrates the end-to-end flow of information within the Network Security Tutor & Quiz Agent, validating its secure, efficient, and fully local architecture.



Prompt 1: Explain the Kerberos authentication protocol and its ticketing mechanism

Procedure:
When the user enters the query “Explain the Kerberos authentication protocol and its ticketing mechanism” in the Q&A interface of the Network Security Tutor, that request is sent from the client-side (flask frontend running on port 57235) to the backend running locally on port 57246. The backend receives the question, processes it through the Chroma vector database to fetch relevant knowledge, and then forwards the combined context to the Llama 3.2 model hosted on Ollama for answer generation.
Once the response is generated, it is returned to the frontend where it is displayed clearly in the tutor interface — outlining the Kerberos Authentication Protocol, its ticketing mechanism, and the ticket structure in well-formatted bullet points.

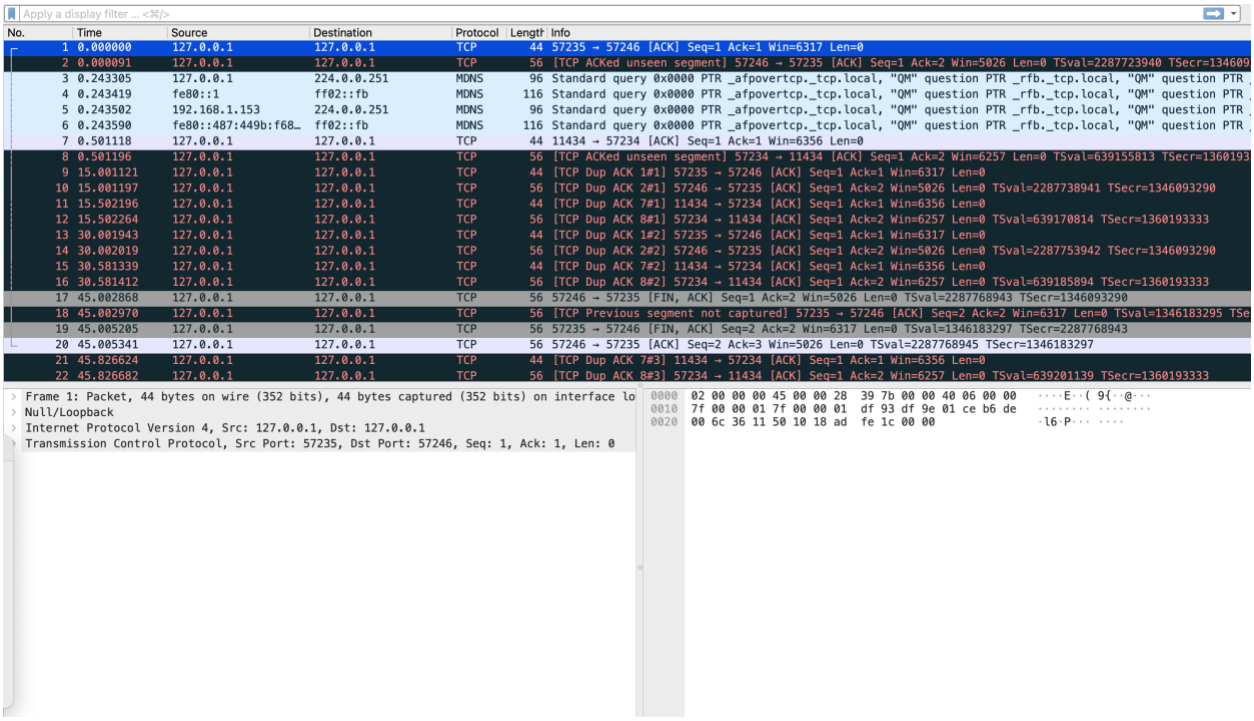
The captured browser view shows the user query at the top and the model’s structured explanation below, confirming successful frontend–backend communication and response rendering.



Wireshark Observation:

- Source IP: 127.0.0.1
- Destination IP: 127.0.0.1
- Source Port: 57235
- Destination Port: 57246
- Protocol: TCP

The captured packets show that all data exchange occurs over the loopback interface, confirming that the entire communication is local.
The sequence begins with the TCP three-way handshake (SYN → SYN/ACK → ACK) establishing the connection between the browser and backend.
After the connection setup, the client sends an HTTP POST request containing the JSON payload of the user's question.
In the detailed packet view, the transmitted data includes the Kerberos query text.



Once the backend processes the request and generates the answer, the final response is sent back to the browser as HTTP/1.1 200 OK, confirming successful data delivery and completion of the Q&A transaction.

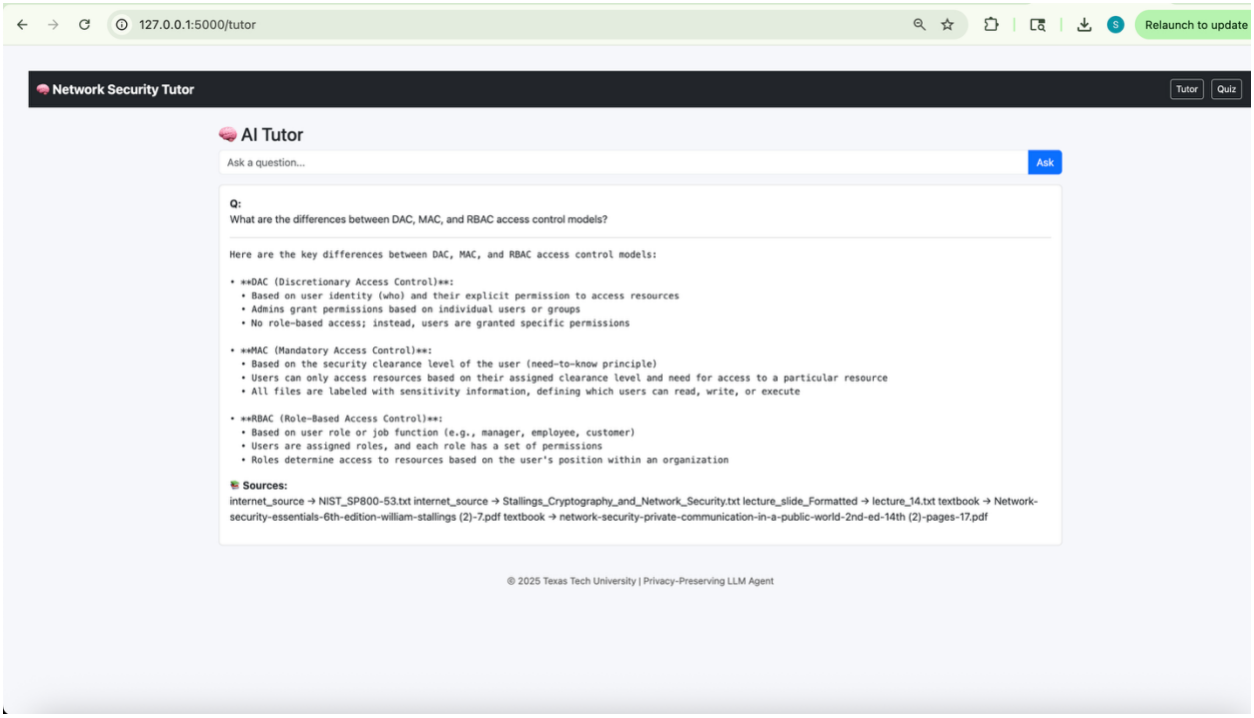
The trace also shows duplicate ACKs and FIN/ACK packets, representing normal TCP acknowledgment and session termination patterns between ports 57235 and 57246.
The captured packets confirm that:

- All traffic stays within 127.0.0.1, ensuring local-only operation.
- The HTTP exchange concludes with 200 OK, validating that the Kerberos protocol explanation request and response were executed successfully.

Prompt 2: What are the differences between DAC, MAC, and RBAC access control models?

Procedure:
When the user enters the query “What are the differences between DAC, MAC, and RBAC access control models?” in the Q&A interface of the Network Security Tutor, the request is sent from the frontend (flask, running on port 57235) to the backend (server, running on port 57246). The backend processes the input and performs retrieval from the Chroma vector database, which stores reference materials such as lecture notes, textbooks, and security framework documents (e.g., NIST SP800-53 and Stallings’ *Network Security Essentials*).
The context retrieved is then passed to the Ollama-hosted Llama 3.2 model, which generates a detailed response outlining the distinctions among Discretionary Access Control (DAC), Mandatory Access Control (MAC), and Role-Based Access Control (RBAC).

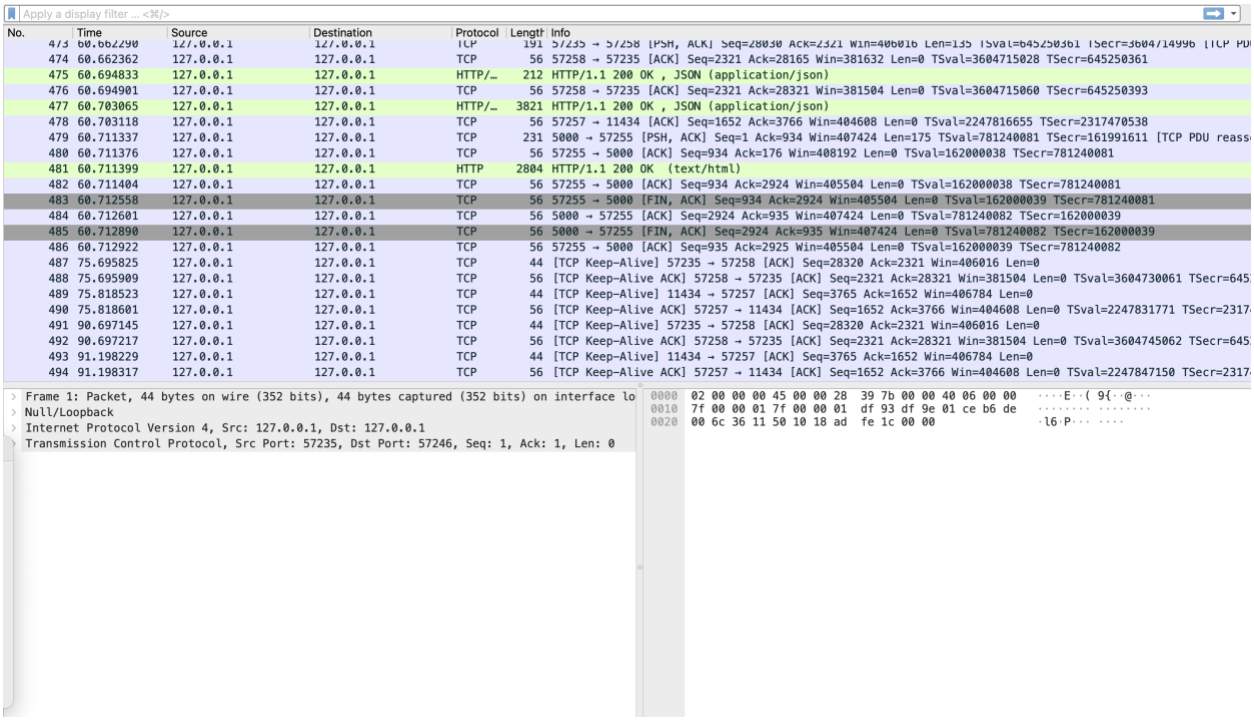
The frontend then displays the formatted explanation with bullet points for each model, including key attributes and decision principles, followed by source citations confirming that the response was built using validated local materials.



Wireshark Observation:

- Source IP: 127.0.0.1
- Destination IP: 127.0.0.1
- Source Port: 57235
- Destination Port: 57246
- Protocol: TCP / HTTP

The packet trace clearly shows all communication confined to the loopback interface, verifying that the data never leaves the local system.
The capture sequence begins with a TCP handshake (SYN → SYN/ACK → ACK) establishing the connection between the browser and backend server.
Subsequent HTTP POST packets contain the user's query payload, followed by multiple HTTP/1.1 200 OK responses indicating successful backend processing and result delivery.



The detailed capture also reveals normal network flow patterns such as ACK, FIN/ACK, and Keep-Alive messages exchanged between the same local endpoints, ensuring persistent connectivity during model inference.
Packet details confirm:

- The frontend (port 57235) communicates with the backend (port 57246).
- The server responds with JSON and HTML packets tagged as application/json and text/html.
- The transaction ends with HTTP/1.1 200 OK, confirming successful completion of the access control comparison request.

Thus, the Wireshark session validates that:

- All packets remain within 127.0.0.1, confirming full local operation.
- The request–response cycle completed successfully, delivering the DAC vs. MAC vs. RBAC explanation to the user interface without any network errors.

Prompt 3: How does two-factor authentication enhance security?

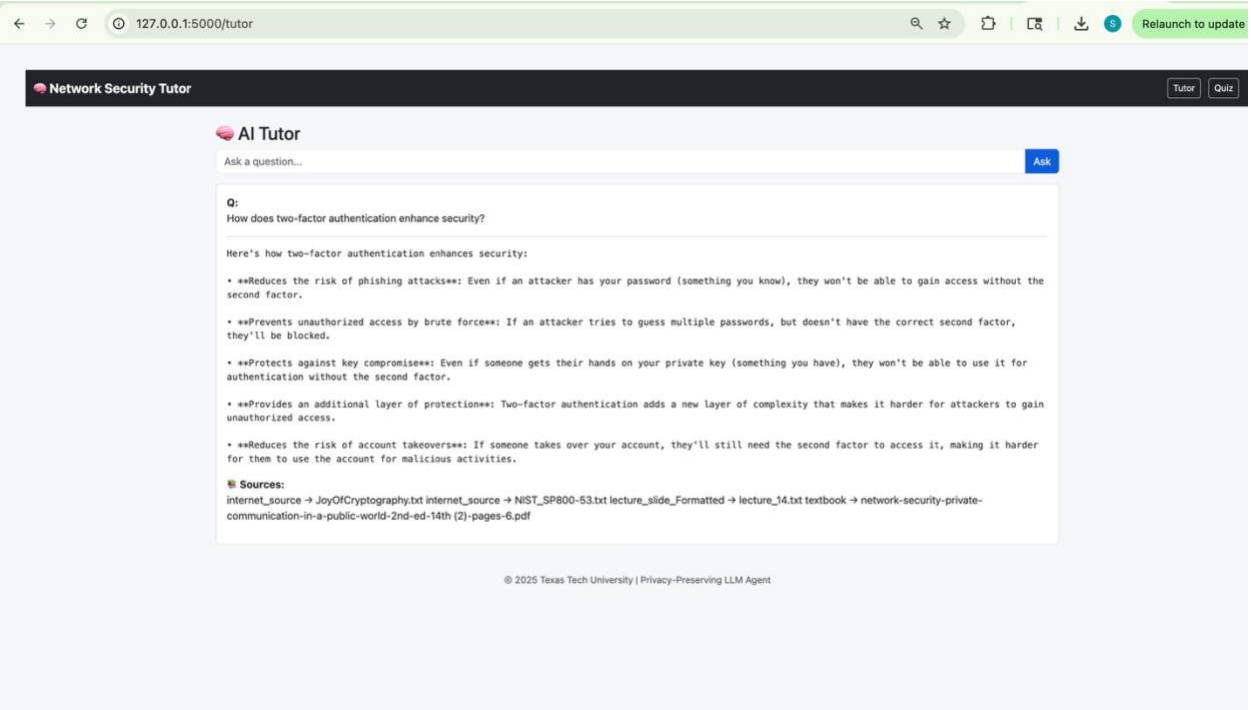
Procedure:
When the user submits the query “How does two-factor authentication enhance security?” in the AI Tutor interface of the Network Security Tutor application, the question is transmitted from the frontend (flask, port 57235) to the backend (port 57246) over the local host interface.
The backend processes the query by fetching relevant context from the Chroma vector database, which includes materials like security frameworks (e.g., *NIST SP800-53*), lecture notes, and standard textbooks such as *Network Security: Private Communication in a Public World*.

The retrieved context is then passed to the Ollama-based Llama 3.2 model for response generation.

The resulting answer displayed in the browser explains how two-factor authentication (2FA) enhances security by:

- Reducing risks of phishing and brute-force attacks,
- Preventing unauthorized access even if passwords or keys are compromised,
- Adding additional layers of defense through possession and knowledge factors, and
- Minimizing account takeover incidents.

The structured output includes clear bullet points under each benefit, along with verified source citations confirming the explanation was locally generated using the internal database and model inference.



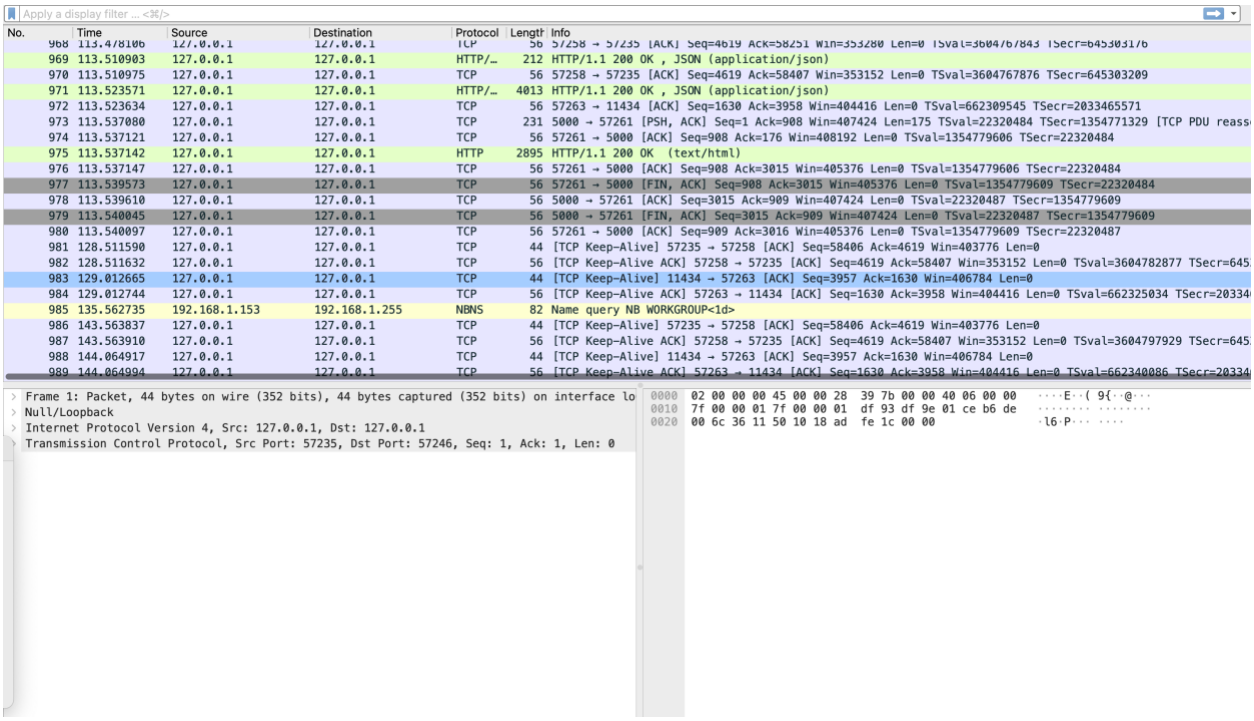
Wireshark Observation:

- Source IP: 127.0.0.1
- Destination IP: 127.0.0.1
- Source Port: 57235
- Destination Port: 57246
- Protocol: TCP / HTTP

The packet trace confirms that all network communication occurred exclusively within the loopback interface, maintaining complete local execution.
The session begins with a standard TCP three-way handshake (SYN → SYN/ACK → ACK)

between the client and backend server.
Subsequently, HTTP POST packets carry the user's question payload, while multiple HTTP/1.1 200 OK responses confirm successful data exchange and content delivery between the frontend, backend, and Llama 3.2 model.

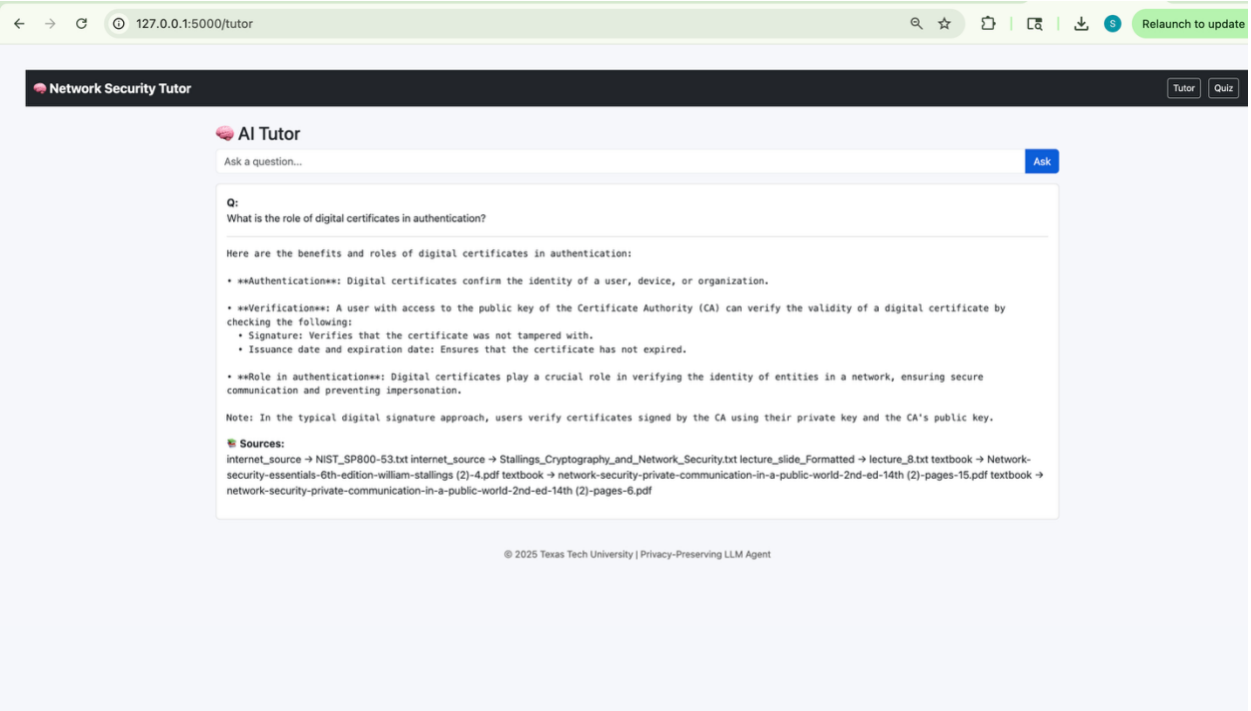
The captured frames also include a normal series of ACK, FIN/ACK, and Keep-Alive packets between ports 57235 and 57246 showing a stable TCP connection maintained throughout the inference and rendering process.
Each 200 OK response signifies that the backend responded correctly with JSON or text content, validating successful local communication.



Prompt 4: What is the role of digital certificates in authentication?

Procedure:
When the user enters the query "What is the role of digital certificates in authentication?" in the AI Tutor interface of the Network Security Tutor application, the request is transmitted from the flask frontend (port 57235) to the backend, which in this instance is operating locally on port 57246.
The backend retrieves contextual information from the Chroma vector database, which contains references from verified learning materials such as *NIST SP800-53*, *Stallings' Cryptography and Network Security*, and *Network Security Essentials (6th Edition)*.
The relevant context is then processed by the Ollama-hosted Llama 3.2 model, which generates a structured explanation describing the authentication, verification, and identity validation roles of digital certificates.

The response shown in the tutor interface highlights how digital certificates confirm the identity of entities, verify integrity through signatures, and ensure authenticity through issuance and expiration validation. The result includes a concise note about certificate verification mechanisms and lists the academic sources used.

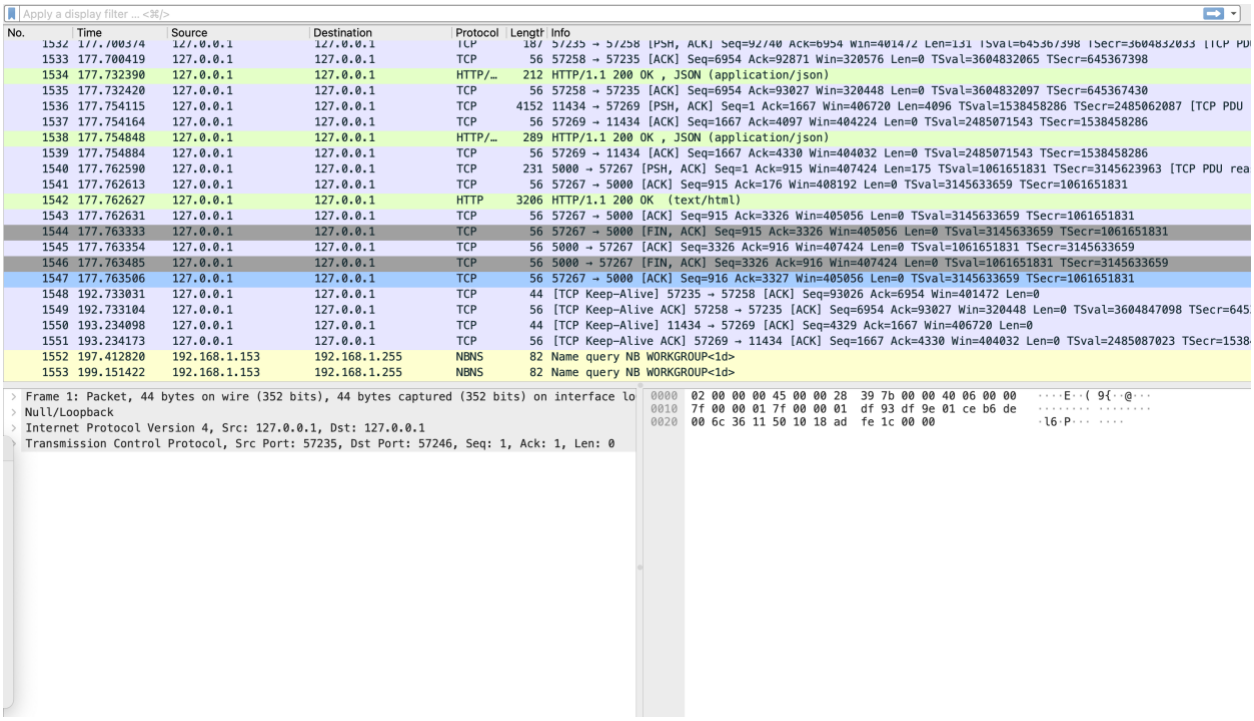


Wireshark Observation:

- Source IP: 127.0.0.1
- Destination IP: 127.0.0.1
- Source Port: 57235
- Destination Port: 57267
- Protocol: TCP / HTTP

The Wireshark capture confirms that all communication occurs locally through the loopback interface, ensuring complete data isolation within the system.
The packet flow begins with a TCP three-way handshake (SYN → SYN/ACK → ACK) establishing the connection between the frontend client and backend server.
Next, the browser sends an HTTP POST request containing the user's question in JSON format. The backend processes it, interacts with the model, and responds with multiple HTTP/1.1 200 OK packets containing the JSON and HTML results.

Throughout the capture, the traffic between port 57235 (client) and port 57267 (server) includes normal ACK, FIN/ACK, and Keep-Alive packets that maintain stable communication until session termination.
Each response concludes successfully with an HTTP 200 OK status, indicating the smooth exchange of requests and results between the local components.



This confirms that:

- All data transmission is confined to 127.0.0.1, validating local execution and privacy.
- The connection sequence completed without packet loss or retransmission.
- The digital certificate explanation was fully processed and displayed within a secure, self-contained local environment.

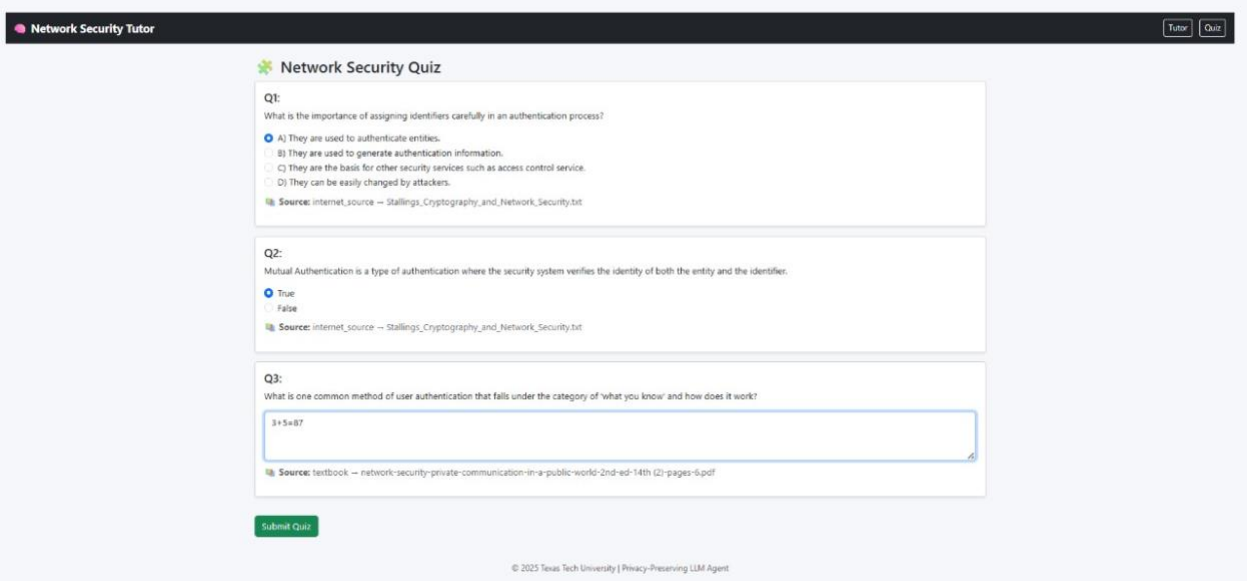
Prompt 5: Quiz Lab

Procedure:
When the user selects the Quiz tab in the Network Security Tutor & Quiz Agent, the system presents configuration options such as Quiz Type, Topic, and Number of Questions.
In this session, the user selected the topic “Authentication Mechanisms” and requested 3 questions. Once the Generate Quiz button was pressed, the frontend (flask, running on port 49938) sent the quiz generation request to the backend (running on port 49951).

The backend fetched relevant materials from the Chroma vector database, which contained lecture content and textbook references related to authentication concepts, and passed this data to the locally hosted Ollama model on the Ollama server. The model then created the quiz questions, formatted them, and sent them back to the frontend for display.

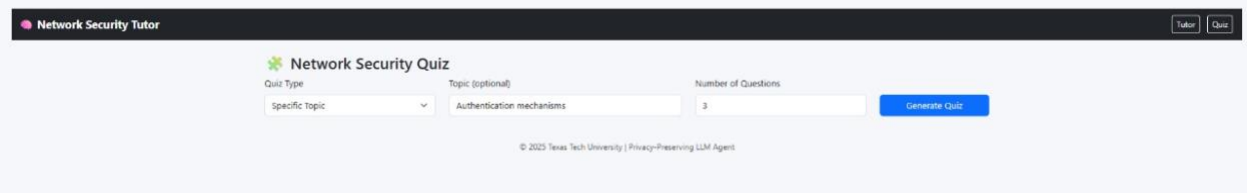
During the quiz, the user answered questions covering authentication identifiers, mutual authentication, and user credential categories. Upon submission, the backend evaluated the answers and returned detailed feedback showing the correct answers, explanations, and final score (1/3).

The response page included color-coded feedback and clear justifications for each answer, confirming that the system's evaluation and grading features worked correctly.



Wireshark Observation:

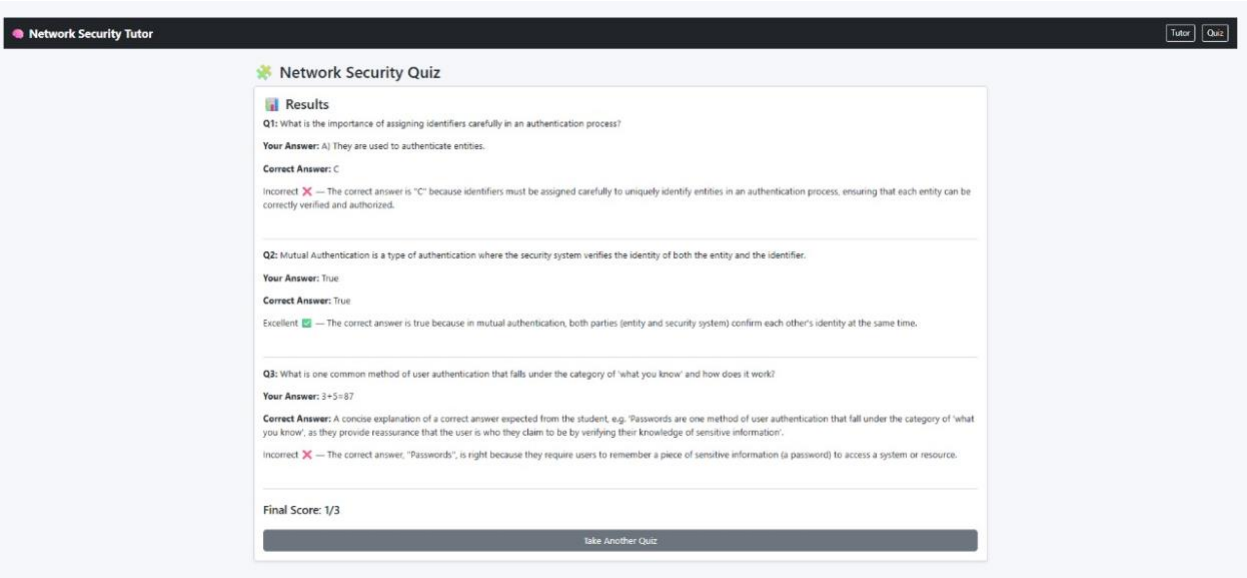
- Source IP: 127.0.0.1
- Destination IP: 127.0.0.1
- Source Port: 49938
- Destination Port: 49951
- Protocol: TCP / HTTP / IGMPv2



The Wireshark trace shows that all communication occurs entirely over the loopback interface (127.0.0.1). The trace begins with a TCP handshake (SYN → SYN/ACK → ACK) to establish communication between the frontend and backend.
Following that, a series of HTTP GET and POST packets capture the full quiz lifecycle:

- Quiz Generation Request: Frontend → Backend via HTTP GET `/quiz`
- Question Retrieval: Backend → Frontend via HTTP 200 OK
- Answer Submission: HTTP POST `/api/quiz/grade`
- Result Delivery: HTTP 200 OK confirming successful grading

The presence of ACK, FIN/ACK, and Keep-Alive packets between ports 49938 and 49951 shows a stable and consistent TCP connection. Additionally, IGMPv2 Membership Report packets are visible, which are standard local multicast maintenance messages unrelated to the quiz communication.



The capture confirms that:

- All network activity was local to 127.0.0.1.
- The quiz generation, submission, and result retrieval processes completed successfully.
- The HTTP sessions consistently returned 200 OK responses, validating full local operation and no data leakage beyond the host system.