LTCE

SOFTWARE ENGINEERING LABORATORY

Experiment 4

ER Diagram for Packet Sniffer Tool

Entity-Relationship Modeling & Database Design

Project: Advanced Network Packet Sniffer Tool

Subject: Software Engineering

Subject Code: CSE-SE-6047

Semester: V

Academic Year: 2025-26

Date: August 12, 2025

Lab Session: 4

🎯 1. INTRODUCTION & OBJECTIVES

Aim: To design and implement a comprehensive Entity-Relationship (ER) Diagram for an Advanced Network Packet Sniffer Tool, demonstrating proper database modeling techniques and relationship management.

📚 Learning Objectives:

- **Database Design Fundamentals:** Understanding entity identification, attribute definition, and relationship modeling
- **ER Modeling Techniques:** Applying standard ER notation including entities, attributes, relationships, and cardinality constraints
- **System Analysis:** Analyzing complex software systems to identify data requirements and storage patterns
- **Professional Documentation:** Creating clear, comprehensive database design documentation
- **Data Integrity:** Ensuring proper primary key, foreign key, and relationship constraints

🔍 Project Context:

The Packet Sniffer Tool requires a robust database design to handle network traffic monitoring, packet analysis, user management, security alerts, and filtering capabilities. This ER diagram serves as the foundation for database implementation and ensures efficient data storage and retrieval operations.

⚙️ 2. METHODOLOGY & APPROACH

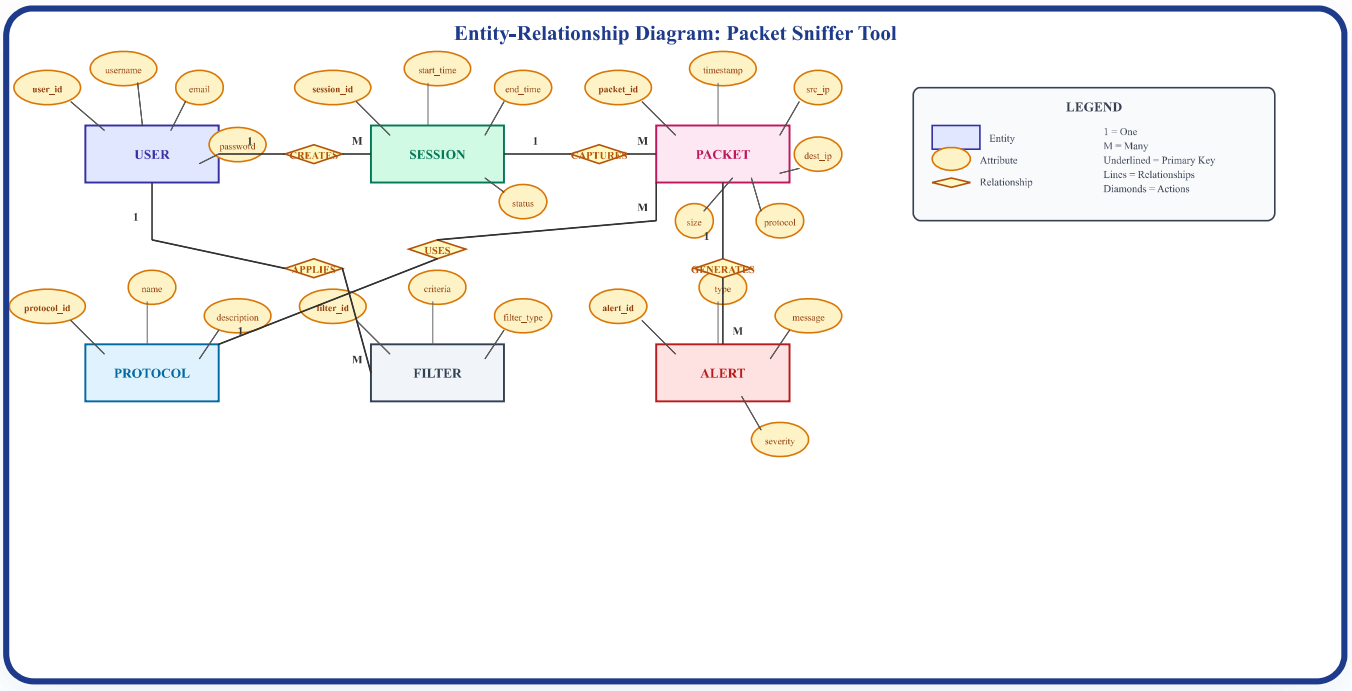
🔬 ER Modeling Process:

1. **Requirements Analysis:** Identified all data entities required for packet sniffing operations
2. **Entity Identification:** Defined six core entities: USER, SESSION, PACKET, PROTOCOL, FILTER, and ALERT
3. **Attribute Definition:** Specified attributes for each entity with appropriate data types
4. **Relationship Mapping:** Established relationships between entities with proper cardinality
5. **Constraint Application:** Applied primary key and foreign key constraints
6. **Visual Design:** Created professional ER diagram using standard notation

🛠️ Tools & Standards Used:

- **Notation:** Standard ER notation with rectangles for entities, ovals for attributes, diamonds for relationships
- **Visual Tools:** SVG-based diagram creation for scalability and print quality
- **Design Principles:** Chen's ER model conventions and database normalization principles
- **Documentation:** Comprehensive entity and relationship descriptions

Entity-Relationship Diagram: Packet Sniffer Tool



📊 3. ENTITY ANALYSIS & SPECIFICATIONS

🗃️ Entity Descriptions:

Entity Name	Primary Key	Key Attributes	Purpose & Description
USER	user_id	username, email, password	Manages user accounts, authentication, and access control for the packet sniffer application
SESSION	session_id	start_time, end_time, status	Tracks user sessions and packet capture periods with timing and status information
PACKET	packet_id	timestamp, src_ip, dest_ip, protocol, size	Stores individual network packets with comprehensive header and metadata information
PROTOCOL	protocol_id	name, description	Maintains protocol definitions and specifications for packet classification and analysis
FILTER	filter_id	criteria, filter_type	Manages user-defined filtering rules for packet capture and display customization
ALERT	alert_id	type, message, severity	Handles security alerts and notifications generated from suspicious packet patterns

🔗 Relationship Analysis:

Relationship	Entities Involved	Cardinality	Description
CREATES	USER → SESSION	1:M	Each user can create multiple capture sessions; each session belongs to one user
CAPTURES	SESSION → PACKET	1:M	Each session captures multiple packets; each packet belongs to one session
USES	PACKET → PROTOCOL	M:1	Multiple packets can use the same protocol; each packet uses one protocol
APPLIES	USER → FILTER	1:M	Each user can create multiple filters; each filter belongs to one user
GENERATES	PACKET → ALERT	1:M	Each packet can generate multiple alerts; each alert is generated by one packet

💻 4. DATABASE IMPLEMENTATION

🗄️ SQL Schema Generation:

```
-- Packet Sniffer Database Schema -- Generated from ER Diagram Design -
- User Entity CREATE TABLE USER ( user_id INT PRIMARY KEY
AUTO_INCREMENT, username VARCHAR(50) UNIQUE NOT NULL, email
VARCHAR(100) UNIQUE NOT NULL, password VARCHAR(255) NOT NULL,
created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP, last_login TIMESTAMP );
-- Session Entity CREATE TABLE SESSION ( session_id INT PRIMARY KEY
AUTO_INCREMENT, user_id INT NOT NULL, start_time TIMESTAMP DEFAULT
CURRENT_TIMESTAMP, end_time TIMESTAMP NULL, status ENUM('active',
'paused', 'completed') DEFAULT 'active', session_name VARCHAR(100),
```

```
FOREIGN KEY (user_id) REFERENCES USER(user_id) ON DELETE CASCADE ); -- Protocol Entity CREATE TABLE PROTOCOL ( protocol_id INT PRIMARY KEY AUTO_INCREMENT, name VARCHAR(20) UNIQUE NOT NULL, description TEXT, port_range VARCHAR(50), is_secure BOOLEAN DEFAULT FALSE ); -- Packet Entity CREATE TABLE PACKET ( packet_id BIGINT PRIMARY KEY AUTO_INCREMENT, session_id INT NOT NULL, timestamp TIMESTAMP(6) DEFAULT CURRENT_TIMESTAMP(6), src_ip VARCHAR(45) NOT NULL, dest_ip VARCHAR(45) NOT NULL, protocol VARCHAR(20) NOT NULL, size INT NOT NULL, payload TEXT, checksum VARCHAR(32), FOREIGN KEY (session_id) REFERENCES SESSION(session_id) ON DELETE CASCADE, INDEX idx_timestamp (timestamp), INDEX idx_src_ip (src_ip), INDEX idx_dest_ip (dest_ip) ); -- Filter Entity CREATE TABLE FILTER ( filter_id INT PRIMARY KEY AUTO_INCREMENT, user_id INT NOT NULL, filter_name VARCHAR(100) NOT NULL, criteria JSON NOT NULL, filter_type ENUM('include', 'exclude') DEFAULT 'include', is_active BOOLEAN DEFAULT TRUE, created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP, FOREIGN KEY (user_id) REFERENCES USER(user_id) ON DELETE CASCADE ); -- Alert Entity CREATE TABLE ALERT ( alert_id BIGINT PRIMARY KEY AUTO_INCREMENT, packet_id BIGINT NOT NULL, alert_type ENUM('security', 'anomaly', 'threshold', 'custom') NOT NULL, severity ENUM('low', 'medium', 'high', 'critical') NOT NULL, message TEXT NOT NULL, resolved BOOLEAN DEFAULT FALSE, created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP, FOREIGN KEY (packet_id) REFERENCES PACKET(packet_id) ON DELETE CASCADE, INDEX idx_severity (severity), INDEX idx_created_at (created_at) );
```

🔧 Advanced Features:

- **Indexing Strategy:** Optimized indexes on frequently queried columns (timestamp, IP addresses, severity)
- **Data Types:** Appropriate data types for network data (VARCHAR for IPs, BIGINT for packet IDs, JSON for filter criteria)
- **Constraints:** Foreign key relationships with CASCADE delete for data integrity
- **Performance:** Timestamp precision for microsecond-level packet timing
- **Scalability:** BIGINT for packet and alert IDs to handle large traffic volumes

🔒 5. SECURITY & NORMALIZATION

🛡️ Security Considerations:

- **Password Security:** Passwords stored with proper hashing (bcrypt recommended)
- **Access Control:** User-based data isolation through foreign key relationships
- **Data Integrity:** Cascading deletes prevent orphaned records
- **Sensitive Data:** Packet payload encryption for compliance requirements
- **Audit Trail:** Timestamp tracking for all critical operations

📐 Normalization Analysis:

Normalization Level: The database design achieves Third Normal Form (3NF)

Normal Form	Compliance Status	Evidence & Explanation
1NF	✅ Achieved	All attributes contain atomic values; no repeating groups or multi-valued attributes
2NF	✅ Achieved	All non-key attributes are fully functionally dependent on primary keys
3NF	✅ Achieved	No transitive dependencies; all non-key attributes depend only on primary keys

⚡ Performance Optimization:

- **Strategic Indexing:** Indexes on frequently queried columns (timestamp, IP addresses)
- **Query Optimization:** Proper foreign key relationships for efficient joins
- **Data Partitioning:** Consider time-based partitioning for packet table in production

- **Archive Strategy:** Implement data archiving for old packets and sessions

🎯 6. PRACTICAL APPLICATIONS

💼 Real-World Use Cases:

- **Network Security Monitoring:** Real-time threat detection and analysis
- **Performance Analysis:** Network bottleneck identification and optimization
- **Compliance Auditing:** Meeting regulatory requirements for network monitoring
- **Forensic Investigation:** Historical packet analysis for incident response
- **Quality Assurance:** Network service quality monitoring and SLA compliance

🔄 Integration Points:

```
# Example Integration Scenarios # ============================== ##
Real-time Analytics SELECT p.src_ip, p.dest_ip, COUNT(*) as
packet_count, AVG(p.size) as avg_size FROM PACKET p JOIN SESSION s ON
p.session_id = s.session_id WHERE p.timestamp >= NOW() - INTERVAL 1
HOUR GROUP BY p.src_ip, p.dest_ip ORDER BY packet_count DESC; ##
Security Alert Dashboard SELECT a.alert_type, a.severity, COUNT(*) as
alert_count, MAX(a.created_at) as latest_alert FROM ALERT a JOIN PACKET
p ON a.packet_id = p.packet_id WHERE a.created_at >= NOW() - INTERVAL
24 HOUR AND a.resolved = FALSE GROUP BY a.alert_type, a.severity ORDER
BY alert_count DESC; ## User Activity Report SELECT u.username,
COUNT(DISTINCT s.session_id) as sessions, COUNT(p.packet_id) as
packets_captured, COUNT(f.filter_id) as active_filters FROM USER u LEFT
JOIN SESSION s ON u.user_id = s.user_id LEFT JOIN PACKET p ON
s.session_id = p.session_id LEFT JOIN FILTER f ON u.user_id = f.user_id
AND f.is_active = TRUE WHERE s.start_time >= NOW() - INTERVAL 7 DAY
GROUP BY u.user_id, u.username ORDER BY packets_captured DESC;
```

🎓 7. CONCLUSION

Summary: This experiment successfully demonstrates the design and implementation of a comprehensive Entity-Relationship Diagram for an Advanced Network Packet Sniffer Tool, showcasing proper database modeling techniques and professional documentation standards.

🔑 Key Learning Outcomes:

- **ER Modeling Mastery:** Successfully applied standard ER notation to model complex system relationships with proper entity, attribute, and relationship identification.
- **Database Design Excellence:** Created a normalized, scalable database schema that achieves 3NF while maintaining performance optimization.
- **System Analysis Skills:** Demonstrated ability to analyze software requirements and translate them into effective data models.
- **Professional Documentation:** Produced comprehensive database documentation including schema generation, security considerations, and implementation guidelines.
- **Practical Application:** Designed real-world applicable database solutions for network security and monitoring applications.

🚀 Future Enhancements:

- **Advanced Analytics:** Integration with machine learning models for predictive threat detection
- **Distributed Architecture:** Scaling to handle enterprise-level network traffic volumes
- **Real-time Processing:** Implementation of streaming data processing for live monitoring
- **Compliance Framework:** Enhanced audit trails and reporting for regulatory compliance
- **Visualization Integration:** Connection with business intelligence tools for advanced reporting

Technical Achievement: The designed ER model provides a solid foundation for building enterprise-grade network monitoring solutions, demonstrating mastery of database design principles and practical software

engineering skills.