Faculty of Engineering – Ain Shams University

Computer and Systems Engineering Department

zGate Gateway: A Zero Trust Database Access Proxy

Graduation Project Thesis

Supervisor:
Dr. Mohammed Sobh

Academic Year 2025–2026

Acknowledgments

We would like to thank...

Abstract

This project introduces a Zero Trust–based database access gateway (SecureDB Gateway)...

Contents

List of Figures

List of Tables

1. Team Information

1.1 Team Members

1.2 Roles & Responsibilities

2. Introduction & Problem Definition

2.1 Introduction

2.2 Problem Statement

2.3 Gap in Existing Solutions

2.4 Why Zero Trust for Databases is Different

3. Project Definition

3.1 Project Definition & Scope

3.2 Objectives

3.3 Expected Academic Contribution

4. Requirements Engineering

4.1 Functional Requirements

4.2 Non-Functional Requirements

4.3 Actors & Use Cases

4.4 Use Case Diagrams

4.5 User Stories

5. Proposed Solution

5.1 Overview of the Proposed Solution

5.2 Solution Architecture Layers

5.2.1 Authentication Layer

5.2.2 Proxy Layer

5.2.3 Policy Engine Layer

5.2.4 Data Protection Layer

5.2.5 Observability Layer

5.3 Term 1 vs Term 2 Features

5.4 Feature List

6. Alignment with International Standards

6.1 PCI DSS

6.2 HIPAA

6.3 GDPR

6.4 ISO 27001

7. Competitor & Market Analysis

7.1 Competitor Analysis

7.1.1 Comparison Table

7.1.2 What Competitors Lack

7.2 Market Research

7.2.1 Market Overview

7.2.2 Zero Trust Demand

7.2.3 Market Challenges & Needs

7.2.4 Regulatory Drivers

7.2.5 Trends & Opportunities

7.2.6 Landscape Summary

8. Scientific Research & Literature Review

8.1 Paper 1

8.2 Paper 2

8.3 Paper 3

8.4 Paper 4

8.5 Paper 5

8.6 Paper 6

8.7 Paper 7

9. Technical Background

10. System Architecture

10.1 High-Level Architecture Diagram

10.2 Main System Components

10.3 Component Communication Flow

10.4 Tech Stack Summary

11. Detailed Architecture of the Proxy

11.1 Connection Lifecycle

11.2 Authentication Flow

11.3 Query Filtering Flow

11.4 Policy Enforcement Flow

11.5 Session Monitoring Flow

11.6 User → Gateway → Database Diagram

12. High-Level Data Flow Diagrams

12.1 Authentication Flow Diagram

12.2 Query Filtering Diagram

12.3 Logging & Auditing Flow Diagram

13. Technology Justification

13.1 Why Go

13.2 Why Node.js / TS / React

13.3 Why mTLS (and why TCP is temporary)

13.4 Why SQLite / Internal Storage

13.5 Design Decision Summary

14. Prototype – Semester 1

14.1 Implemented Features

14.2 Screenshots (CLI & Dashboard)

14.3 What Works vs What Doesn’t

14.4 Technical Decisions Made

14.5 Implementation Challenges

15. Development Methodology

15.1 Agile Scrum Framework

To manage the complexity of the project and ensure continuous development, the team adopted the Agile Scrum methodology. We structured the development lifecycle into one-week sprints, enabling rapid iteration and frequent feedback cycles. This short sprint duration allowed us to demonstrate tangible progress weekly and incorporate supervisor feedback more frequently, ensuring alignment with project objectives throughout the development process.

15.2 Meeting Structure

Our workflow is organized around three distinct meeting types: the Weekly Kick-off, Daily Stand-ups, and the Sprint Review with stakeholders. This structured rhythm of recurring meetings ensures that team milestones and progress are consistently monitored and synchronized. Collectively, these meetings serve to define, review, and align all weekly tasks in accordance with agile best practices.

15.2.1 Weekly Kick-off Meeting

This meeting is held at the start of every sprint to align the team for the upcoming week. It encompasses three key components:

- **Retrospective:** We briefly analyze the previous sprint, discussing what went well and identifying bottlenecks (e.g., merge conflicts or unclear requirements). This reflection helps the team avoid repeating past mistakes and continuously improve our process.
- **Backlog Refinement:** We review upcoming tasks to ensure they are clearly defined and that all technical requirements are understood before assignment. This step reduces ambiguity and sets clear expectations.
- **Sprint Planning:** We select specific tasks from the refined backlog to be completed in the current week. Tasks are assigned to team members based on priority,

complexity, and estimated effort.

15.2.2 Daily Stand-up Meeting

This brief synchronization meeting is held daily to maintain continuous team alignment and identify blockers early.

- **Duration:** Limited to approximately 15 minutes total, with each member speaking for no more than 2 minutes. This time constraint ensures the meeting remains focused and efficient.
- **Format:** Each team member addresses two specific points: what they accomplished yesterday and what they plan to work on today. Any blockers or dependencies are also raised.
- **Objective:** This practice ensures that no team member works in isolation or is blocked by dependencies without the rest of the team being aware. It promotes transparency and enables rapid problem-solving.

15.2.3 Sprint Review (Weekly Supervisor Meeting)

At the conclusion of each sprint, a formal review is conducted with our project supervisor and mentors to validate progress and gather feedback.

- **Demonstration:** The team presents the functional features completed during the sprint, showcasing working software rather than just documentation or plans.
- **Validation:** Our supervisors provide immediate feedback on the implementation. This feedback is either approved for integration or converted into new change requests to be prioritized in the next sprint's backlog.

15.3 Collaboration Tools

To ensure accessibility and efficient collaboration across all phases of development, we employ an integrated tool stack that supports both synchronous and asynchronous communication:

Central Knowledge Base (Notion): Serves as the single source of truth for the project wiki. It is used for sprint planning, task and goal tracking, documenting new code features, and archiving meeting notes and recordings. All team members have real-time access to project documentation.

Version Control & Technical Documentation (GitHub): The primary repository for source code, version control, and code reviews. GitHub also hosts technical documentation, including the Proxy Installation Guide and API references.

Synchronous Communication: Discord, Microsoft Teams, and Google Meet are used for scheduled meetings and real-time voice/video collaboration. These platforms facilitate immediate discussion and decision-making.

Asynchronous Communication: WhatsApp is used for quick, urgent team notifications and simple coordination when immediate responses are needed outside of scheduled meetings.

Auxiliary Tools: We leverage AI-powered tools for generating meeting summaries and conclusions, creating immediate and searchable records of discussions. Excalidraw is used for creating collaborative diagrams, flowcharts, and architectural drawings during design discussions.

15.4 Documentation & Observability

We prioritize high observability by documenting all progress, regardless of scale. This comprehensive documentation approach ensures transparency and facilitates knowledge transfer within the team.

While day-to-day execution is tracked in Notion, high-level progress reporting is documented in a formal **Supervisor Meeting Log** maintained as a Word document. This log serves as an official record and tracks:

- **Attendance & Date:** A record of meeting participants and timestamps.
- **Retrospective:** Feedback on completed tasks, including what was delivered and any deviations from the plan.
- **Forward Planning:** Clearly defined goals and expected outcomes for the subsequent meeting, establishing accountability and measurable targets.

16. Task Tracking

16.1 Team Task Tracking (Actual Examples)

16.2 Supervisor Tracking Logs

16.3 Blockers, Risks & Resolution Notes

17. Milestones

17.1 Term 1 Milestone Roadmap

17.1.1 Milestone 1

17.1.2 Milestone 2

17.1.3 Milestone 3

17.1.4 Milestone 4

17.1.5 Milestone 5

17.2 Timeline Chart (Gantt-like)

18. Threat Model & Security Considerations

18.1 Threat Model

18.2 Risks & Attack Vectors

18.3 Mitigation Techniques

18.4 Why Zero Trust is Needed

19. Roadmap for Term 2

19.1 Remaining Features

19.2 Architecture Improvements

19.3 Performance Goals

19.4 Testing & Validation Plan

20. Team Contribution

20.1 Overview of Contribution Approach

20.2 Individual Contributions

21. Expected Outcomes

22. Conclusion

22.1 Restated Purpose

22.2 Summary of Achievements

22.3 Importance & Contribution

22.4 Transition to Next Semester

References

A. Glossary

B. Dashboard Mockups