



Enabling Consistent Stateful Security in Distributed Web Application Firewalls: A Framework for Scalable Cloud Environment

Puvipavan Palendrarajah
(Reg. No.: MS24020244)

A THESIS
SUBMITTED TO
SRI LANKA INSTITUTE OF INFORMATION TECHNOLOGY
IN PARTIAL FULFILMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE IN INFORMATION TECHNOLOGY (CYBER SECURITY)

December 2025

I certify that I have read this thesis and that in my opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Dr. Dharshana Kasthurirathna

Approved for MSc. Research Project:

MSc in IT(C8S) Programme Co-ordinator, SLIIT

Approved for MSc:

Head of Graduate Studies, FoC, SLIIT

DECLARATION

This is to certify that the work is entirely my own and not of any other person, unless explicitly acknowledged (including citation of published and unpublished sources). The work has not previously been submitted in any form to the Sri Lanka Institute of Information Technology or to any other institution for assessment for any other purpose.

Sign:

Puvipavan Palendrarajah

Date:

ABSTRACT

Enabling Consistent Stateful Security in Distributed Web Application Firewalls: A Framework for Scalable Cloud Environment

Puvipavan Palendrarajah

MSc. in Information Technology Specialization in Cyber Security

Supervisor: Dr Dharshana Kasthurirathna

December 2025

The rapid adoption of cloud-native infrastructures has highlighted a critical limitation in existing Web Application Firewalls (WAFs): their stateless design restricts consistent enforcement of security policies across distributed environments. This research addresses this gap by designing and evaluating a portable persistence module for open-source WAFs, enabling stateful security enforcement through integration with distributed data stores. Guided by the principles of design science research [1], the study develops a pluggable framework that supports both Redis and Memcached as backends. Redis is widely recognized for its durability and advanced data structures [2], while Memcached offers lightweight, in-memory caching optimized for speed [3]. By embedding the module within ModSecurity v3 [4] and deploying it on AWS cloud infrastructure, the research benchmarks the comparative performance of Redis and Memcached under simulated traffic and attack scenarios, including Distributed Denial of Service (DDoS) conditions [5]. Evaluation metrics include latency overhead, throughput, memory utilization, and resilience under node failures. Preliminary results indicate that Redis achieves superior consistency and resilience, albeit with higher memory consumption, while Memcached provides lower latency at the cost of weaker fault tolerance. Beyond technical performance, the research contributes a generalizable, portable framework that can be embedded into other open-source WAFs, expanding their applicability in distributed and multi-tenant environments. Both artifact and empirical evaluation contributions position the work as a step forward in bridging distributed systems and web security, while also providing a foundation for future enhancements such as adaptive, machine-learning-based intrusion prevention [6].

ACKNOWLEDGEMENT

During my time at Sri Lanka Institute of Information Technology, I have been fortunate to work under the guidance of dedicated advisors whose diverse perspectives greatly enriched my academic journey. I wish to express my deepest gratitude to my supervisor Dr. Dharshana Kasthurirathna for offering valuable insights, continuous constructive feedback, and mentorship throughout the development of this thesis. Their encouragement helped refine my ideas, while also allowing me the independence to explore new directions with confidence.

I also extend my appreciation to the faculty members, panel members and colleagues at SLIIT for fostering an environment of collaboration and support. Discussions with peers often inspired fresh approaches to complex problems, and their willingness to share knowledge was invaluable. The moral support, intellectual encouragement, and academic resources provided by the institution and community were essential for completing this work.

TABLE OF CONTENTS

DECLARATION	ii
ABSTRACT.....	iii
ACKNOWLEDGEMENT.....	iv
TABLE OF CONTENTS.....	v
List of Figures	xi
List of Tables	xii
Chapter 1 Introduction	1
1.1 Background	1
1.2 Research Problem	4
1.3 Research Objectives.....	7
1.3.1 Main Objective.....	7
1.3.2 Specific Objectives	8
1.3.3 Expected Outcomes	11
1.4 Significance of the Research	12
1.4.1 Academic and Scientific Significance	12
1.4.2 Practical and Industrial Significance	13
1.4.3 Strategic and Societal Relevance	14
1.4.4 Research Novelty and Contribution Summary.....	15
1.4.5 Expected Impact.....	15
1.4.6 Summary	16
1.5 Research Questions	16
1.5.1 Primary Research Question.....	16
1.5.2 Secondary Research Questions.....	17
1.5.3 Expected Insights from Research Questions.....	18
1.5.4 Summary	19
1.6 Thesis Organization.....	19
1.7 Summary	21
Chapter 2 Literature Review	22
2.1 Web Application Firewalls (WAFs).....	22
2.2 WAF Evasion and Limitations.....	23
2.3 Persistence in WAFs and Distributed Systems.....	23
2.4 Redis and Memcached in Distributed Environments	25
2.5 Cloud Security and Attack Landscape	26
2.6 Research Methodology: Design Science	27

2.7 Research Gap	27
2.7.1 Lack of Portable Persistence Frameworks for WAFs	27
2.7.2 Limited Comparative Evaluations of Redis and Memcached in Security Contexts	28
2.7.3 Insufficient Focus on Persistence in Cloud-Native Security Architectures	28
2.7.4 Differentiation Between System Building and Design Science	28
2.7.5 Summary of Research Gap	29
Chapter 3 Research Methodology	30
3.1 Design Science Research (DSR) Framework.....	30
3.1.1 The DSR Process Model	30
3.2 Research Design	31
3.2.1 Approach.....	31
3.2.2 Research Paradigm	31
3.3 Artifact Design Process	31
3.3.1 Design Requirements	32
3.3.2 API Specification.....	32
3.3.3 Development Tools	33
3.4 Research Environment	33
3.4.1 Cloud Infrastructure.....	33
3.4.2 Software Stack	34
3.5 Data Collection and Analysis	34
3.5.1 Metrics	34
3.5.2 Data Collection Tools	34
3.5.3 Data Analysis Techniques.....	35
3.6 Validation and Reliability	35
3.6.1 Functional Validation	35
3.6.2 Reliability Assurance	35
3.7 Ethical Considerations.....	35
3.8 Methodological Limitations	36
3.9 Summary	36
Chapter 4 Implementation.....	36
4.1 Introduction	36
4.2 Architectural Overview	37
4.2.1 Design Goals.....	37
4.2.2 Layered Architecture.....	38
4.3 Persistence Abstraction Layer.....	39
4.3.1 Class Hierarchy and Interfaces.....	39

4.3.2 Data Model and Serialization	39
4.3.3 Transaction Lifecycle	40
4.4 LMDB Backend Implementation	40
4.4.1 LMDB Workflow	40
4.4.2 LMDB Characteristics	41
4.5 Redis Backend Implementation	41
4.5.1 Architectural Integration.....	41
4.5.2 Key Operations.....	41
4.5.3 Lua-Based Atomicity	42
4.5.4 Persistence and Recovery	42
4.6 Memcached Backend Implementation.....	42
4.6.1 Connection Setup.....	42
4.6.2 Characteristics.....	43
4.7 Configuration and Deployment	43
4.7.1 Configuration Parameters.....	43
4.7.2 Deployment Workflow	44
4.8 Fault Handling and Consistency Control	44
4.8.1 Failure Detection.....	44
4.8.2 Recovery Mechanisms	44
4.8.3 Lazy Expiration	44
4.9 Logging and Observability	44
4.10 Optimization and Thread Safety	45
4.11 Verification and Testing	45
4.12 Summary	46
Chapter 5 Results and Evaluation	46
5.1 Introduction	46
5.2 Benchmark Environment and Methodology.....	47
5.2.1 Rationale for AWS Selection	47
5.2.2 Detailed Topology Description.....	47
5.2.3 Measurement Infrastructure	48
5.3 Baseline Evaluation Using LMDB.....	48
5.3.1 Observed Metrics.....	48
5.3.2 Interpretation.....	49
5.4 Quantitative Performance Results.....	49
5.4.1 Latency Distribution	49
5.4.2 Throughput Analysis	50

5.4.3 CPU and Memory Utilization.....	50
5.4.4 Scalability Across Auto-Scaling Nodes	50
5.5 Reliability and Consistency Under Failure.....	50
5.5.1 Redis Behavior.....	51
5.5.2 Memcached Behavior	51
5.5.3 Observational Logs.....	51
5.6 Attack-Response Validation	51
5.6.1 Brute-Force Scenario.....	52
5.6.2 DDoS Flood Scenario	52
5.7 Qualitative Observations	52
5.7.1 Log Consistency.....	52
5.7.2 Operational Stability	53
5.8 Statistical Reliability and Uncertainty Analysis	53
5.9 Synthesis and Interpretation	53
5.10 Extended Discussion of Theoretical Implications	54
5.11 Alignment with Design Science Research Evaluation	54
5.12 Limitations and Future Extensions.....	55
5.13 Summary	55
Chapter 6 Discussion.....	56
6.1 Introduction	56
6.2 Reflection on Research Problem and Objectives	56
6.3 Theoretical Implications.....	57
6.3.1 The CAP Theorem and Security Enforcement Consistency	58
6.3.2 Design Science Research Evaluation	59
6.4 Technical and Engineering Implications.....	59
6.4.1 Consistency and Fault Tolerance	59
6.4.2 Performance and Scalability	60
6.4.3 CPU and Resource Utilization Trade-offs.....	60
6.4.4 Observability and Maintainability.....	60
6.5 Practical and Industrial Implications.....	61
6.5.1 Adoption in Cloud-Native Security.....	61
6.5.2 Security Operations and Policy Uniformity.....	61
6.5.3 Cost and Performance Considerations	61
6.6 Comparison with Prior Work	62
6.7 Lessons Learned	62
6.7.1 Balancing Simplicity and Generalization	62

6.7.2 Importance of Observability	62
6.7.3 Redis vs Memcached as Complementary Systems	63
6.8 Broader Research and Theoretical Contributions.....	63
6.9 Limitations and Threats to Validity	63
6.10 Future Directions.	64
6.10.1 Multi-Backend Federation	64
6.10.2 Integration with Machine Learning.....	64
6.10.3 Formal Consistency Verification	64
6.10.4 Multi-Cloud and Edge Expansion	64
6.11 Summary	64
Chapter 7 Conclusion and Future Work.....	65
7.1 Introduction	65
7.2 Summary of the Research Process.....	65
7.3 Summary of Key Findings.....	66
7.3.1 Achievement of Research Objectives	66
7.3.2 Redis vs. Memcached - Comparative Insights.....	67
7.3.3 Security and Enforcement Consistency	68
7.3.4 Performance Stability and Resource Efficiency	68
7.4 Theoretical Contributions	68
7.5 Practical and Industrial Contributions	69
7.5.1 Framework Generalization Across Open-Source WAFs.....	69
7.5.2 Enhanced Cloud Deployments	70
7.5.3 Operational Improvements.....	70
7.6 Lessons Learned and Research Reflections	70
7.6.1 The Hidden Complexity of Consistency	70
7.6.2 Balancing Performance and Generality	70
7.6.3 Observability as a Critical Requirement.....	71
7.6.4 Design Science in Practice.....	71
7.7 Limitations.....	71
7.8 Recommendations for Future Work	71
7.8.1 Multi-Cluster Replication and Geo-Distribution	71
7.8.2 Hybrid Persistence Model.....	72
7.8.3 Machine Learning Integration.....	72
7.8.4 Formal Consistency Verification	72
7.8.5 Broader Application Domains	72
7.9 Implications for Academia and Industry	72

7.9.1 Academic Impact.....	72
7.9.2 Industrial and Societal Impact	73
7.10 Conclusion.....	73
7.11 Closing Remarks	73
Chapter 8 References.....	74
Appendix	75
Appendix 1: Source Code Listings	75

List of Figures

Figure 1.1 Generalized Persistence Framework for Distributed Web Application Firewalls.....	4
---	---

List of Tables

Table 1.1: 1.4.4 Research Novelty and Contribution Summary.....	15
Table 2.1 Comparison of WAF Models and Limitations.....	23
Table 2.2 Feature Comparison of Redis vs Memcached.....	26
Table 2.3 Summary of Research Gaps vs Proposed Contributions.....	30
Table 3.1: Data collection metrics	34
Table 4.1: Architecture design goals.....	38
Table 4.2: LMDB Characteristics	41
Table 4.3: Redis Core Operations.....	42
Table 4.4: Memcached Core Operations	43
Table 4.5: Memcached Characteristics.....	43
Table 4.6: Framework verification and testing phases.....	46
Table 5.1: Baseline metrics with LMDB.....	49
Table 5.2: Attack response validation for brute-force scenario	52
Table 6.1: Research Problem and Objectives Outcomes	57
Table 6.2: CAP Theorem and WAF Trade-offs	58
Table 7.1: Summary of the Research Process.....	66
Table 7.2: Achievement of Research Objectives.....	67
Table 7.3: Redis vs Memcached Comparison Metrics.....	68