WASP Software Engineering Assignment

Karim Khalil

August 21, 2025

1 Introduction

1.1 Research Area

The urgent need for effective tools and methodologies to assess and implement Distributed Denial of Service (DDoS) attack mitigation in 5G networks is a central concern in modern network security research. Much of the existing literature on anomaly detection in data traffic focuses on one dimension: evaluating machine learning models or static flow analysis techniques against offline benchmarks. While this approach is valuable for standardizing comparison, it overlooks a critical aspect of deployment feasibility and performance in real-world production environments.

Any traffic analysis and attack mitigation solution needs to be evaluated not only by its theoretical performance but also by its real-world effectiveness when integrated into live networks. Robust mitigation strategies must account for the dynamic nature of 5G infrastructure.

The primary goal of our research is to develop a test framework tailored to the needs of 5G environments. This framework will facilitate the generation of diverse attack scenarios, enabling the collection of attack data and testing of mitigation solutions. Our methodology aims to offer a holistic evaluation of solutions, where they are assessed not only by conventional benchmarking but also by their deployability, computational complexity, impact on network latency, resource consumption, and overall effectiveness in maintaining service continuity.

2 Lecture principles

2.1 Principle 1: Diversity-based testing

The principle is about deliberately constructing test suites that span a broad, meaningful space of inputs to expose blind spots, spurious correlations, and limitations of in ML solutions. This connects directly to the core aim of my research, as it builds a 5G attack-mitigation test framework that evaluates solutions beyond offline benchmarks and under realistic variability. For example, in 5G DDoS contexts, traffic distributions are dynamic and often non-stationary due to heterogeneous devices, slice configurations, and service-level requirements. Relying solely on single-source benchmarks underrepresents this variability and can overestimate deployability.

Diversity-based testing in this framework has three layers. First, generating diverse attack scenarios: volumetric floods, protocol-specific anomalies, low-and-slow patterns, and adaptive attacks that shift over time. Second, measuring diversity via representation over the dataset space dispersion. This is where metrics like pairwise distances in latent embeddings and compression-based similarity (e.g., NCD) can help quantify whether a test corpus truly spans distinct behavioral regions rather than clustered variants of the same pattern. Third, combining diversity generation with data augmentation and controlled permutations to stress generalization while maintaining semantic validity.

The test framework would support (a) comparative evaluations across public datasets versus locally synthesized traffic, (b) calibration of test adequacy criteria that go beyond accuracy and F1 to include latency impact, computational overhead, and resilience under distribution shift, and (c) reproducible test profiles that reflect real 5G constraints. By integrating diversity measurement and generation, the research closes the gap between static benchmarking and deployment-centric reliability.

2.2 Principle 2: Property-based and metamorphic testing

A second principle from the lectures is to test systems against explicit properties rather than enumerating examples: property-based testing (PBT), complemented by metamorphic testing and invariants. For ML-based traffic analysis and mitigation, absolute ground truth is often scarce at scale, while the input space is vast and adversarial behavior evolves. Properties and metamorphic relations offer a way to specify expected behavior under controlled transformations without requiring fully labeled standards for every case.

In the DDoS mitigation setting, useful properties include invariance to irrelevant feature variations (e.g., benign jitter in packet timing or header fields that should not alter classification), monotonicity under attack amplification (e.g., if attack intensity increases by a factor within plausible bounds, detection confidence should not decrease), stability under benign aggregation (e.g., batching of packets or minor reorderings should not flip decisions), and bounded degradation under load (e.g., mitigation action latency should remain below a target threshold under specified traffic rates). Metamorphic relations can encode equivalence classes and controlled transformations: duplicating non-informative padding, permuting order within a window, or applying noise consistent with radio conditions should preserve labels; conversely, injecting attack-signature features should monotonically increase detection likelihood.

3 Guest-Lecture Principles

3.1 Requirements Engineering and Goal Modeling

This principle is about understanding the problem and stakeholders before designing any solution. Applied to a 5G DDoS test framework, it keeps the work anchored in real operational needs rather than convenient benchmarks. The first step is to identify what matters to each stakeholder and express those aims as clear goals: providers care about service continuity, low latency, and SLA compliance; users care about reliability, responsiveness, and fairness; operators care about scalability and costs. These goals then become concrete, testable requirements. For example, if ultra-low latency is essential, the framework should specify target bounds for mitigation delay under defined traffic loads; if fairness is a goal, detection should not disproportionately flag legitimate heavy users; if robustness is required, detection confidence should not collapse when attackers vary patterns within realistic ranges.

Next, the problem space needs structure. On the attack side, define the relevant families: volumetric floods, protocol misuse, low-and-slow, and multi-vector blends, along with their intensities, durations, and evolution over time. On the network side, describe the context in which mitigation operates: core, edge, and RAN components. On the data side, specify what datasets are needed, how synthetic traffic is generated, how it compares to public benchmarks, and which validation checks must pass before training or testing. This leads to clear categories of requirements: data must be realistic and validated, scenarios must cover required attack mixes and load profiles, performance must meet bounds for detection delay, mitigation latency, CPU/memory overhead, and throughput impact.

4 Data Scientists versus Software Engineers

- Do you agree on the essential differences between data scientists and software engineers put forward in these chapters? Why or why not?
  - I actually disagree a little bit with the book hard split between data scientists and software engineers. Sure, there are real differences. The book talks about them in terms of preference and training, like how data scientists like doing modeling and exploring data, while software engineers usually enjoy building infrastructure and making things scale for users. But I don't think these are essential or always set in stone. Honestly, with enough drive and curiosity, one person can totally build at least a solid working skillset in both areas, even if they aren't a "superstar" in both. The book calls these folks "unicorns," like people who are rare or even mythical, but I don't buy that they're that rare, at least not on a functional level. Anyone who's motivated, works in a cross-disciplinary team, or just keeps learning on the job can pick up enough skills from both sides to be really valuable. In the end, most good products have to combine the strong points of both: good, well-tested models and robust production code. In real life, you don't need to be pure specialist, a competent generalist can still make great stuff, especially with help from domain experts, and other teammates to fill in gaps.
- Do you think these roles will evolve and specialise further or that "both sides" will need to learn many of the skills of "the other side" and that the roles somehow will merge? Explain your reasoning.

- I think, it's going to be a bit of both. On one hand, teams will always want deep specialists, especially when things get super technical like think hardcore ML research or scaling systems used by millions. There will always be jobs for people who just really love modeling, or people who only want to create massive distributed systems. But on the other side, I think more and more companies need hybrid people who can "speak both languages" and move around the team stack. Products move faster when there is someone who gets ML, software, and business needs at least well enough to connect the dots. The book point about generalists makes sense here, they really are the bridge, and more companies are probably going to want at least a couple of those on every team. But in general, I would say both sides will have to learn the basics of the other job, just to keep up with how things are done. You won't get hired as a "pure" data scientist these days if you can't put your model into production or at least work with those who do, and vice versa for software folks who want a foot in ML. Products are moving so fast, so being able to jump between the two is just going to get more valuable, not less. So yeah, specialization will get deeper for some jobs, but a lot of the day-to-day will blur together and the roles will definitely keep merging.

5 Paper analysis

5.1 Paper 1: Generating and Verifying Synthetic Datasets with Requirements Engineering, Vonderhaar et. al. 2025

5.1.1 Core Ideas

The core idea of the paper is to bring formal and traceable requirements engineering (RE) principles into the creation and verification of synthetic datasets generated by AI models, particularly for data augmentation. The approach treats synthetic data not just as random samples churned out for convenience, but as artifacts that must fulfill precise, stakeholder-driven specifications. Requirements are written for what the generated data must contain defined specification, these are translated directly into prompts for generative models, and then the outputs are verified both manually and with downstream detection models (here, YOLOv8). This RE-based rigor ensures that synthetic data actually supports intended model behaviors, making ML pipelines more trustworthy. In the context of software engineering for AI, this transforms the often ad hoc world of data augmentation into a process with validation, traceability, and clear rationale, which is especially important.

5.1.2 Relation to My Own Research

The paper's approach is directly relevant to my own research, which involves building frameworks for creating and testing synthetic datasets for network security scenarios like DDoS attack detection in 5G environments. Just as the paper argues for requirements-driven synthetic datasets in vision tasks, my framework could benefit from a formal process to specify exactly what types of attack traffic or anomalies need to be generated to supplement or match public datasets. For example, it would be useful to formally define the characteristics of real-world attacks (traffic rates, attack types, protocol features), then generate synthetic traffic that both augments and complements available public datasets to fill in gaps or representing underexplored attack vectors tailored for 5G. This approach also enhances the quality and consistency of data, making sure synthetic examples are not just more of the same, but targeted to address specific research and deployment needs.

5.1.3 Integration into a Larger AI-Intensive Project

Let's say there is a larger AI-based software project for autonomous vehicles (drones or cars) that relies heavily on visual recognition to operate safely. The system needs to recognize not just everyday cases, but also rare or dangerous edge cases—like unusual road hazards, bad weather, or adversarial images. This is an area where the paper's ideas can come to life, Instead of relying on the hope that generative models will randomly produce just the right edge cases, the project team would write explicit requirements for what the synthetic data should include (e.g., "images with obscured road signs at multiple angles). Generative AI models are then prompted and their outputs are checked systematically against these requirements, possibly leveraging detection systems for even more automated verification.

My own WASP research could fit in here as a supporting module, for example, anomaly detection tools that flag images whose distribution or properties are suspicious, such as abnormally generated content inserted by a hacker trying to fool the autonomous system. This would add a layer of real-time monitoring and safety to the AI pipeline by ensuring that not only the training data, but also inputs received in the wild, are validated.

5.1.4 Adaptation of your research

Long-term, to better support the challenges outlined in the paper, I'd tweak my project to build in more formal requirements engineering from the start. For instance, I would search the literature to ensure I add scenarios and attack data that previous work has identified as relevant for building such a traffic anomaly model. These requirements would directly inform the generation process, and every generated dataset would be validated against them before use.

One challenge, though, is the use of GANs or diffusion models for generating "new" data. If these generative models are trained mainly on public datasets, the synthetic samples may not add much genuine novelty, they may just remix what's already there. So, I'd be a bit skeptical, and would want to add measures to analyze the true diversity and utility of generated data compared to existing sets, not just in appearance but also in terms of how much they help downstream models generalize or spot uncommon attacks.

5.2 Paper 2: Is Your Anomaly Detector Ready for Change? Adapting AIOps Solutions to the Real World, Poenaru-Olaru, et. al. 2024

5.2.1 Core Ideas

This paper tackles the maintenance of anomaly detection models in AIOps, especially under data that changes over time . It compares retraining strategies (full-history training vs. sliding window) and evaluates how different methods affect anomaly detection performance. The paper treats ML models as evolving software artifacts that require continuous maintenance, testing, and adaptation.

5.2.2 Relation to My Own Research

We have written a paper with the title "Resilient Automatic Model Selection for Mobility Prediction". published in Cluster Computing Journal later this year, which addresses the same core idea of maintaining the prediction quality of mobility prediction models in 5G networks. The focus is on mobility predictions under adversarial conditions. Both papers recognize the need to adapt ML in deployment, but while the AIOps paper looks at retraining signals and frequency for anomaly detection, our work examines retraining strategies for robustness under data poisoning. Both highlight that real-world ML needs robust and flexible updating, not just initial accuracy.

5.2.3 Integration into a Larger AI-Intensive Project

We can think of a large-scale telcom analytics platform that integrates operational anomaly detection (AIOps) and predictive mobility analytics (like my paper's focus). This paper's ideas can inform the anomaly monitoring/alerting subsystem: by adopting dynamic retraining (drift-aware or periodic), the system can stay resilient to changes in network or usage patterns. Our research would plug into the mobility prediction subsystem, providing robust, AutoML-driven predictions even as adversaries attempt to poison inputs—helping the whole platform maintain reliable performance as data and threats evolve.

5.2.4 Adaptation of your research

To better support the AIOps, I could explicitly include performance monitoring in my AutoML retraining schedule, not just react to accuracy but actually monitor for distribution shift or data quality drops. This could mean combining feature drift detectors with AutoML for integrating drift signals as triggers for model retraining. This can combine our current adversarial-robust retraining logic with the system maintenance mindset promoted in the AIOps anomaly paper.

6 Research Ethics & Synthesis Reflection

- (Search and screening process) How you searched for and screened/selected the papers?

  - I went to https://ieeexplore.ieee.org/xpl/conhome/1842224/all-proceedings, and searched in the proceedings per year, going from latest.

- (Pitfalls and mitigations), Any misleading titles/abstracts you encountered? How you adjusted your search and screening when initial results were not effective?

  - The paper "Generating and Verifying Synthetic Datasets with Requirements Engineering" in the title and abstract sound like they're talking about a general method for testing synthetic datasets, but once you get into the paper, it's pretty much all about image datasets made with diffusion models or GANs. They make it seem like their approach is universal, but the process is really just: write a requirement, hand it to a generative model, and hope it spits out an image that fits. There's not much applicability to non-image or non-GAN datasets, so that's a little misleading. The "verification" is also pretty basic, just checking if YOLO or a person thinks the image matches the prompt. So, yeah, the title oversells the generality compared to what's actually delivered.

- Ethical considerations (Steps taken to ensure originality (no copying from LLMs or sources))?

  - I usually either use grammarly or preplexity where I write my understanding on the topic or question, and what I want to write, but in a very sloppy way, as I feel it is the fastest way to write. Then I ask the LLM to make my text clearer and coherent.