FUZZING ROUTING PROTOCOLS IN FLYING AD-HOC NETWORKS

Group_No: 50

Review -1

Guide: Dr. Vishnuvarthan R

SL NO.	Name	Roll No
1	Pavithraa S	CB.EN.U4CSE22343
2	Gayathri Sivakumar	CB.EN.U4CSE22117
3	Ratakonda Bhavitha	CB.EN.U4CSE22148
4	Srinidhi J	CB.EN.U4CSE22350

Mail Approval

Paste approve mail screen shot from Guide

Problem Statement

Flying Ad-Hoc Networks (FANETs) present significant communication and security challenges due to their dynamic topology, high node mobility, and three-dimensional environment. Existing validation methods includes formal verification and other models developed using network simulators. They face limitations in abstraction, insufficient coverage of dynamic behaviours, and a lack of systematic methods to detect unknown vulnerabilities. Through this project we aim to propose a fuzzing driven testing strategy for protocol vulnerabilities.

T. Kim, S. Lee, K. H. Kim, and Y.-I. Jo, "FANET Routing Protocol Analysis for Multi-UAV-Based Reconnaissance Mobility Models," *Drones*, vol. 7, no. 3, p. 161, Mar. 2023. doi: 10.3390/drones7030161

What is Fuzzing?

- Fuzzing, or fuzz testing, is a software testing method that helps identify bugs and security issues by providing a program with large volumes of input data.
- These inputs can range from malformed or unexpected data to valid inputs, which are used to test how the software handles different scenarios.
- It has been successful in finding hidden bugs in traditional software.
- Softwares like SQLite, Python, OpenSSL, BoringSSL, gRPC, WOFF2, LLVM (Clang, clang-format, libc++), TensorFlow, FFmpeg, Wireshark, QEMU use fuzzing to discover vulnerabilities.

Chen Chen, Baojiang Cui, Jinxin Ma, Runpu Wu, Jianchao Guo, Wenqian Liu, A systematic review of fuzzing techniques, Computers & Security, Volume 75, 2018, Pages 118-137, ISSN 0167-4048, https://doi.org/10.1016/j.cose.2018.02.002.

Types of Fuzzing

Factors	**Whitebox Fuzzing**	**Blackbox Fuzzing**	**Greybox Fuzzing**
Requirement of source code	Full knowledge	Not required	Partial knowledge
Effectiveness	Very high (deep code coverage, finds complex bugs)	Low to moderate (limited coverage, may miss deep bugs)	High (better than blackbox, close to whitebox with lesser effort)
Speed	Slow (requires symbolic execution, heavy computation)	Fast (no instrumentation or analysis)	Moderate to fast (uses lightweight instrumentation)
Scalability	Low (resource-intensive and hard to apply to large codebases)	High (easily scalable, minimal setup)	High (balance between feedback and resource use)
Suitability	Best for critical, high-assurance systems	Best for black-box, legacy, or closed-source systems	Best for practical open-source fuzzing with good bug discovery.

MALVIYA, Vikas Kumar; MINN, Wei; SHAR, Lwin Khin; and JIANG, Lingxiao. Fuzzing drones for anomaly detection: A systematic literature review. (2025). Computers and Security. 148, 1-45. Available at: https://ink.library.smu.edu.sg/sis_research/9910

Fuzzing Tools

Fuzzer	Type of Fuzzing	Advantages	Disadvantages
AFLNet	Greybox	Builds on AFL for stateful protocols. Uses code coverage and response codes.	Needs manual parsing logic for responses. Not plug and play for encrypted protocols
LibFuzzer	Greybox	Fast, coverage-guided, and integrates well with sanitizers for finding in-process bugs.	Requires source code with LLVM instrumentation and can't fuzz out-of-process targets.
StateAFL	Greybox	Combines memory and state tracking. Enhances AFL with multi-layer feedback.	High instrumentation complexity. Requires manual state mapping.
BLEEM	Blackbox	Targets BLE. Enforces structured protocol sequences	Lacks feedback-guided fuzzing .
WhisperFuzz	Whitebox(1st one with static analysis)	Automates finding side-channel vulnerabilities (e.g., timing leaks).	Niche focus; not a general-purpose bug finder. Can be complex to configure.
CIFuzz	AI Automated	Integrates fuzzing into CI/CD pipelines for continuous testing. Easy setup for OSS-Fuzz projects.	Effectiveness depends on the underlying engine. Best for standard build systems.

Xiaohan Zhang et al., "A Survey of Protocol Fuzzing", ACM Computing Surveys, Vol. 57, No. 2, Article 35, Oct 2024. https://doi.org/10.1145/3696788
Shihao Jiang et al., "A Survey of Network Protocol Fuzzing: Model, Techniques and Directions", arXiv:2402.17394v1, Feb 2024
Code Intelligence. (2020). CI Fuzz: Continuous Fuzzing for Modern Development. Available at: https://www.code-intelligence.com
P. Borkar *et al.*, "WhisperFuzz: White-Box Fuzzing for Detecting and Locating Timing Vulnerabilities in Processors," *arXiv preprint* arXiv:2402.03704, 2024. Accepted to USENIX Security Symposium 2024. arXivRahul Kande

Introduction to FANETs

- FANET refers to a group of mobile drones that continuously modify their network structure without relying on any permanent infrastructures.

- Applications of FANET :
  - disaster management .
  - Border Surveillance, Defence systems [7].
  - Agricultural monitoring
  - relaying networks.
  - traffic monitoring.

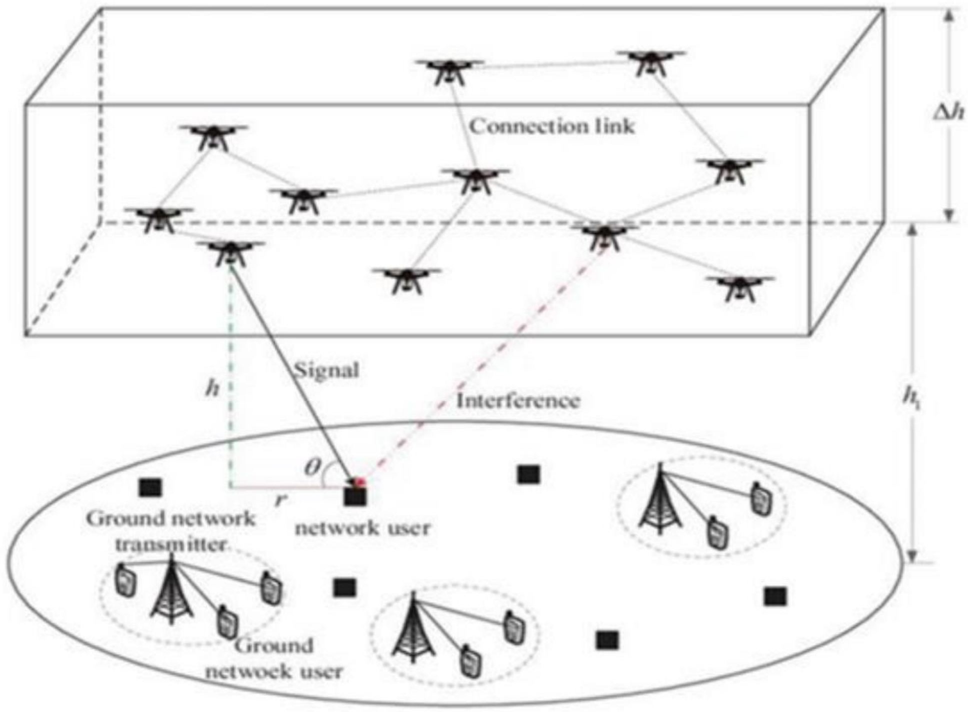


Fig. 1

[7] Z. Sun, P. Wang, M. C. Vuran, M. Al-Rodhaan, A. Al-Dhelaan, and I. F. Akyildiz, “BorderSense: border patrol through advanced wireless sensor networks,” Ad Hoc Netw., vol. 9, no. 3, pp. 468–477, May 2011, doi: 10.1016/j.adhoc.2010.09.008

Different Routing Protocols in FANETs

FANET routing protocols are generally classified into six categories:

1. **Topology-aware routing**: Uses network connectivity information to determine routes based on link states or tables.
2. **Position-aware routing**: Relies on GPS or location data to make routing decisions based on node positions.
3. **Cluster-based routing**: Organizes UAVs into clusters to improve scalability and reduce overhead.
4. **Delay-tolerant networks (DTN)**: Designed for intermittent connectivity, using store-and-forward techniques for reliable delivery.
5. **Heterogeneous-based routing**: Supports networks with different types of UAVs or communication technologies.
6. **Swarm-based routing**: Inspired by swarm intelligence (e.g., ants, bees) to enable adaptive and distributed routing.

[3] Bekmezci, Ilker, Özgür Koray Sahingöz, and Şamil Temel. "Flying Ad-Hoc Networks (FANETs): A Survey." *Ad Hoc Networks*, vol. 11, no. 3, May 2013, pp. 1254–1270. https://doi.org/10.1016/j.adhoc.2012.12.004
İlker Bekmezci, Ozgur Koray Sahingoz, Şamil Temel,Flying Ad-Hoc Networks (FANETs): A survey,Ad Hoc Networks,Volume 11, Issue 3, 2013,Pages 1254-1270,ISSN 1570-8705,https://doi.org/10.1016/j.adhoc.2012.12.004.

Different Routing Protocols in FANETs

- Specifically, within topology-based routing, protocols are divided into **static, proactive, reactive, and hybrid** types. Under FANET topology-based protocols, proactive routing protocols include OLSR, DSDV, BATMAN.
  1. Proactive Routing Protocols (PRPs) maintain up-to-date routing tables on each UAV for fast data transmission.
  2. Since paths are known in advance, proactive protocols significantly reduce message delivery latency compared to reactive approaches .
  3. Proactive protocols are better for real-time UAV tasks (e.g., collision avoidance, video streaming), where immediate communication is critical.
  4. Static routing is only useful for stable or predictable UAV missions, whereas proactive routing suits dynamic, mission-critical operations.

OLSR Protocol

1. Optimized Link State Routing protocol is a proactive routing protocol .
2. It **maintains routes** continuously using periodic control messages.
3. Uses **Multipoint Relays (MPRs)** to reduce the number of transmissions.
4. Relies on **HELLO** messages to detect neighbors and **TC messages** to share topology.
5. Offers l**ow-latency route availability**, suitable for dynamic networks.
6. Can be used in FANETs but may need adaptations due to high mobility and 3D movement.

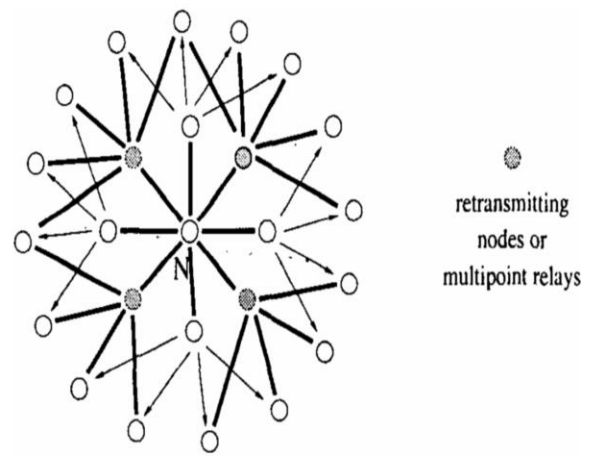


Fig. 2

https://datatracker.ietf.org/doc/html/rfc3626

P. Jacquet, P. Muhlethaler, T. Clausen, A. Laouiti, A. Qayyum and L. Viennot, "Optimized link state routing protocol for ad hoc networks," Proceedings. IEEE International Multi Topic Conference, 2001. IEEE INMIC 2001. Technology for the 21st Century., Lahore, Pakistan, 2001, pp. 62-68, doi: 10.1109/INMIC.2001.995315. keywords: {Routing protocols;Ad hoc networks;Wireless networks;Wireless application protocol;Mobile ad hoc networks;Network topology;Mobile computing;Communication system traffic control;Costs;Relays},

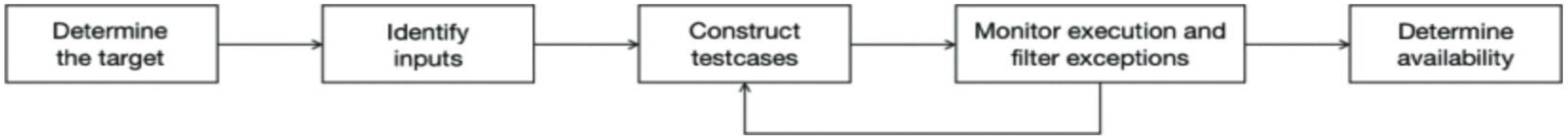
Why Fuzz FANETs ?

- FANETs are extremely vulnerable due to high susceptibility to insider threats due to their decentralized and dynamic nature.
- FANETs act with minimal human interaction which making them difficult candidates for proactive fuzzing.
- Fuzzing communication protocols such as MAVLink, RTPS, and proprietary UAV-ground control protocols.
- Firmware Logic & State Machine Fuzzing
- Interface/API Fuzzing

[1] Malviya, Vikas Kumar, et al. "Fuzzing Drones for Anomaly Detection: A Systematic Literature Review." *Computers & Security*, vol. 148, 2025, Article 104157, https://doi.org/10.1016/j.cose.2024.104157. [1] Malviya, Vikas Kumar, et al. "Fuzzing Drones for Anomaly Detection: A Systematic Literature Review." *Computers & Security*, vol. 148, 2025, Article 104157, https://doi.org/10.1016/j.cose.2024.104157.

Protocol Fuzzing

The basic principle of network protocol fuzzing is to construct malformed packets by generation or mutation, then send malformed packets to the tested protocol entity through sockets and monitor the tested protocol entity to find vulnerabilities in network protocol implementations.



[2]Z. Hu and Z. Pan, "A Systematic Review of Network Protocol Fuzzing Techniques," *2021 IEEE 4th Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC)*, Chongqing, China, 2021, pp. Doi : doi: 10.1109/IMCEC51613.2021.9482063.

Why fuzz OLSR Protocol?

OLSR is currently an actively developing protocol and it is also an open-source protocol. This makes whitebox and greybox fuzzing viable for us. USB and WiFi fuzzing was unable to be done via whitebox fuzzing due to unavailability and access to its source code.

1. Routing instability and loop formation.
2. Lack of control message validation.
3. Inefficient MPR selection mechanism.
4. Vulnerability to topology-based attacks.
5. Overhead from frequent control messages.
6. Performance impact due to incorrect HELLO and TC timer configuration.
7. OLSR lacks authentication, allowing packet tampering and false topology updates.
8. Malicious nodes can exploit MPR roles to disrupt routing (e.g., wormhole attacks).

Clausen, T., & Jacquet, P. (Eds.). (2003). RFC 3626: Optimized Link State Routing Protocol (OLSR). IETF.
- "The OLSR.ORG story". Open-Mesh.org.
- Adar, E., & Lev, B. (2021). Security issues in the Optimized Link State Routing Protocol version 2 (OLSRv2). arXiv preprint arXiv:2105.07198.
- Vyas, N., & Sharma, A. (2013). A Study of Detour Attack on OLSR based Ad-hoc Networks. International Journal of Computer Applications, 75(13).

Fan Hong, Liang Hong and Cai Fu, "Secure OLSR," 19th International Conference on Advanced Information Networking and Applications (AINA'05) Volume 1 (AINA papers), Taipei, Taiwan, 2005, pp. 713-718 vol.1, doi: 10.1109/AINA.2005.305. keywords: {Routing protocols;Computer networks;Mobile ad hoc networks;Ad hoc networks;Communication system security;Information security;Educational institutions;Protection;Network topology;Wireless networks},

Existing Work: MAVLink

- The MAVLink protocol is an open source, point-to-point networking protocol used telemetry and to command and control many small unmanned aircraft.
- **Widespread Use in UAVs:**
  - MAVLink is an open-source protocol for telemetry and command/control in many small unmanned aircraft.
  - UAVs are rapidly increasing and often use unsecured communication.
- **Security Overlooked:**
  - Designed with focus on availability and safety, but security measures are lacking.
- **Known Vulnerabilities:**
  - No authentication or encryption in communication.
  - Susceptible to eavesdropping, spoofing, hijacking, message tampering, replay attacks, denial of service, and more.

K. Kyu-chang, K. Dong-hyun, C. Soo-bin, L. Joon-oh, K. Ji-soo, R. Hyung-ho, and C. Jin-sung, "Vulnerability analysis of PX4 autopilot's MAVLink module," in *Proc. 2021 Winter Conf. Korean Inf. Protection Soc. (CISC-W'21)*, 2021.

Literature Survey

S.No	Title	Year	Main Focus	Key Challenges Addressed	Proposed Solutions	Inference
1	MAVSec: Securing the MAVLink Protocol for Ardupilot/PX4 UAVs	~2017–2018	Fuzzing MAVLink in UAV systems to find communication vulnerabilities.	Insecure MAVLink protocol and lack of dedicated fuzzing tools.	4D-Fuzzer: a Python tool for fuzzing MAVLink over serial/UDP.	MAVLink is vulnerable; fuzzing reveals real-world security flaws.
2	Secure OLSR	~2005	Securing OLSR routing protocol in Mobile Ad Hoc Networks (MANETs) against wormhole and packet tampering attacks	- Wormhole attacks during neighbor discovery - Lack of authentication for MPR selection - No integrity or source verification for routing packets	- Secure OLSR (SOLSR) protocol - Wormhole detection using propagation delay analysis - Identity authentication using digital signatures and certificates - Secure routing packet structure with hash chains and signatures	Provides a robust authentication and wormhole detection scheme for securing routing in MANETs, especially when broadcast-based protocols like OLSR are used
3	Fuzzing Drones for Anomaly Detection: A Systematic Literature Review	2025	Systematic review of fuzzing techniques applied to UAVs, identifying gaps and challenges.	- Immature fuzzing techniques for drones- Lack of scenario-aware fuzzing- Over-reliance on open-source simulators- Limited attack scope beyond control unit	- UAV-specific scenario-aware fuzzing- Multi-level fuzzing (network, application, scenario)- Adoption of IoT fuzzers like PortFuzzer	Urges development of holistic, scenario-aware fuzzers for UAVs and expansion of targets beyond control components.
4	Cyber-Physical Fuzzing Framework for Extracting Butterfly Effect in UAV Systems	2023–2024 (approx.)	Proposes a fuzzing framework for UAVs based on the butterfly effect to identify vulnerabilities in flight control.	- Increasing UAV cyber threats- Need for cyber threat response- Difficulty in identifying impact chains	- Butterfly-effect-based cyber-physical fuzzing- Impact vector analysis- MAVLink protocol + PX4/Ardupilot testbed	Proactively detects vulnerabilities through subtle flight behavior changes to enhance UAV cybersecurity.
5	Analysis of Routing and Security Issues in OLSR Protocol for Video Streaming over MANET	2022	Reviewing enhancements in OLSR with emphasis on routing and security challenges, particularly under video-streaming traffic over mobile ad-hoc networks	Dynamic topology in MANETs for video streaming - Security vulnerabilities in OLSR (e.g. routing misuse, packet tampering) - QoS under mobility and video traffic	Survey of existing OLSR enhancements and security mechanisms; identification of gaps for future enhancements	Highlights the need for protocol improvements to ensure both routing efficiency and security for real-time media in MANETs

Literature Survey

S.No	Title	Year	Main Focus	Key Challenges Addressed	Proposed Solutions	Inference
6	DetecVFuzz: Enhancing Security in Consumer Electronic Devices Through Scalable Vulnerability Testing of Virtual Devices	2023 (approx.)	Introduces DetecVFuzz, a scalable fuzzing framework for virtual PCI devices using QEMU-Qtest.	- Manual stub coding difficulty- Inefficiency in structured fuzzing- Complex nested data structures- VM startup and input randomness	- Seed pool generation & coverage feedback- Static analysis of hypervisor- Multi-seed AFL fuzzing- Fuzzing proxy interface	Automates vulnerability testing of virtual devices, reducing manual effort and finding real-world bugs effectively.
7	Systematic Exploration of Fuzzing in IoT: Techniques, Vulnerabilities, and Open Challenges	2025	Holistic survey on IoT fuzzing: methods, vulnerabilities, and future research needs.	- Complexity of IoT architecture- Lack of universal protocols and evaluation metrics- Proprietary systems limit fuzzing- AI integration barriers	- Categorized fuzzing (technique, target, objective)- AI-powered protocol translation & federated learning- Hybrid edge-cloud models- Blockchain + lightweight crypto- Standardized benchmarks	Advocates AI-augmented, cross-platform, and scalable fuzzing solutions for IoT, with unified standards and hybrid frameworks.
8	Fuzzing the Internet of Things: A Review on the Techniques and Challenges for Efficient Vulnerability Discovery in Embedded Systems	2021	Review of fuzzing for embedded/IoT systems, emphasizing tailored techniques and future directions.	- Embedded resource constraints- OS-less testing challenges- Lack of feedback mechanisms- No standard evaluation or benchmarks	- Constraint-aware fuzzers- Feedback via physical response monitoring- AI-integrated generators- SUT-independent architectures	Emphasizes the need for intelligent, feedback-driven fuzzers tailored to embedded IoT constraints and challenges.
9	GREYHOUND: Directed Greybox Wi-Fi Fuzzing	2020-2022	Directed greybox fuzzing of Wi-Fi client implementations (smartphones, IoT) to find crashes and protocol non-compliance	- Fuzzing stateful network protocols without source code. - Poorly documented device implementations. - Existing tools only find shallow bugs or crashes.	- Holistic protocol state-machine model. - Directed fuzzing with optimized mutation probabilities (using PSO). - Sending well-formed but state-inappropriate packets.	Demonstrates model-based fuzzing finds complex Wi-Fi flaws missed by other tools.
10	A Survey of Protocol Fuzzing (ACM Computing Surveys)	2024	End-to-end guide to protocol fuzzing—challenges, solutions, and future trends	- High communication complexity - Strict semantic constraints (intra & inter-message) - Resource-constrained environments	- Input generator models including communication constraints - Enhanced executors and bug collectors for protocol testing - Novel feedback and oracle design for coverage & accuracy	Establishes an in-depth roadmap for future protocol fuzzing, stressing the need for domain-specific fuzzing infrastructure

Literature Survey

S.No	Title	Year	Main Focus	Key Challenges Addressed	Proposed Solutions	Inference
11	Flying Ad-Hoc Networks (FANETs): A Survey	2013	First comprehensive FANET survey	High node mobility, low density, 3D topology, real-time constraints	Introduces FANET vs MANET/VANET differences; proposes MAC, PHY, routing designs	Establishes FANET as a unique research domain with protocol and hardware challenges
12	Optimized Link State Routing Protocol for Ad Hoc Networks (OLSR)	~2001–2003	Proposing and optimizing the OLSR protocol	Excess flooding, control overhead, bidirectional link maintenance	Multipoint Relay (MPR) selection, HELLO & TC messages, compact control formats	Highly scalable and proactive routing suitable for large ad hoc wireless networks
13	BorderSense: Border Patrol through Advanced Wireless Sensor Networks	2011	Designing a hybrid wireless sensor network architecture for real-time border patrol surveillance	- Large-scale border coverage - Reliable detection - Energy efficiency - Sensor mobility integration	- Combines fixed sensors and mobile surveillance units - Sensor deployment and optimization strategies for wide border regions	Demonstrates feasibility of sensor networks for autonomous, scalable border security monitoring
14	Fuzzing Drones for Anomaly Detection: A Systematic Literature Review	2025	Systematic review of fuzzing techniques applied to UAVs to detect software bugs and vulnerabilities	- UAV fuzzing is immature - Overreliance on closed-source components - Challenges with flight control fuzzing and communication protocol testing	- Categorization of blackbox/whitebox/greybox fuzzing strategies - Mapping drone subsystems to fuzzing domains - Highlights importance of coverage-guided, communication-protocol-aware fuzzing	Emphasizes gap in scenario-aware UAV fuzzing and recommends tailored approaches for drone control and communication layers
15	A Survey of Network Protocol Fuzzing: Model, Techniques and Directions	2024	Comprehensive survey of network protocol fuzzing models, techniques, challenges, and research directions	- Stateful protocols with structured inputs - Dependency on real-time communication - Non-uniform protocol semantics	- Unified 4-phase protocol fuzzing model (syntax modelling, test generation, execution, feedback) - Categorization of protocol-specific fuzzers - Strategy enhancements per model phase	Provides a generic model and design framework for next-gen protocol fuzzers tailored to the unique challenges of protocol behaviour

Literature Survey

S.No	Title	Year	Main Focus	Key Challenges Addressed	Proposed Solutions	Inference
16	Flying Ad-Hoc Networks (FANETs): Review of Communications, Challenges, Applications, Future direction and Open Research Topics	2024	Reviews how UAVs form self-organizing aerial networks and examines FANET communication needs, protocols, and application areas.	Deals with dynamic topology, limited energy, bandwidth constraints, and lack of secure routing mechanisms.	Suggests AI-based adaptive routing, edge/fog computing, and topology-aware relay UAV deployment.	FANETs are promising for real-time missions but need secure, scalable, and latency-aware communication protocols.
17	FANET Routing Protocol Analysis for Multi-UAV-Based Reconnaissance Mobility Models	2024	Examines the design and performance of routing protocols tailored for UAV-based ad hoc networks.	Unstable links due to 3D mobility, limited energy, and communication range issues.	Recommends mobility-aware and mission-specific routing strategies using different protocol types.	Flexible routing solutions are essential to handle the dynamic and heterogeneous nature of FANETs.

Research gap

- Existing fuzzers are not designed for aerial routing protocols such as OLSR or AODV. They fail to account for high mobility, dynamic topology, and real-time communication constraints in FANETs.
- Most fuzzing methods do not use coverage feedback, resulting in limited protocol exploration and low efficiency in identifying deeper logic errors.
- Current research focuses on telemetry or application-level fuzzing. The routing layer, which is critical in FANETs, remains largely unexplored and vulnerable.
- There is no comparative evaluation of black-box, grey-box, and grammar-based fuzzing techniques for FANET routing protocols, leaving a gap in performance validation.
- Fuzzing in embedded systems is often done without access to source code, which can miss protocol-level issues. By focusing on specific modules, fuzzers have a better chance of uncovering more meaningful and hidden vulnerabilities.

Proposed Solution – Innovations in Fuzzing for FANET

Objective:

To detect vulnerabilities in FANET routing protocols (e.g., OLSR) using **coverage-guided grey-box fuzzing** in a realistic simulation environment.

Framework: CGF-FRP (Coverage-Guided Fuzzing of FANET Routing Protocols)

- **FAF Engine:**
  Greybox fuzzer (LibFuzzer/AFLNet) generating state-aware, malformed OLSR packets.
- **Network Simulator or run it in RaspberryPi:**
  Emulates 3D UAV movement, realistic radio conditions, and node dynamics
- **Instrumentation Layer:**
  Code compiled with LLVM/AFL for feedback collection.
- **Control & Monitoring Harness:**
  Injects packets, tracks coverage, monitors performance degradation, manages test loop.

Feedback Loop:

1.Generate → 2. Inject → 3. Simulate → 4. Monitor → 5. Learn → 6. Repeat

Proposed Solution

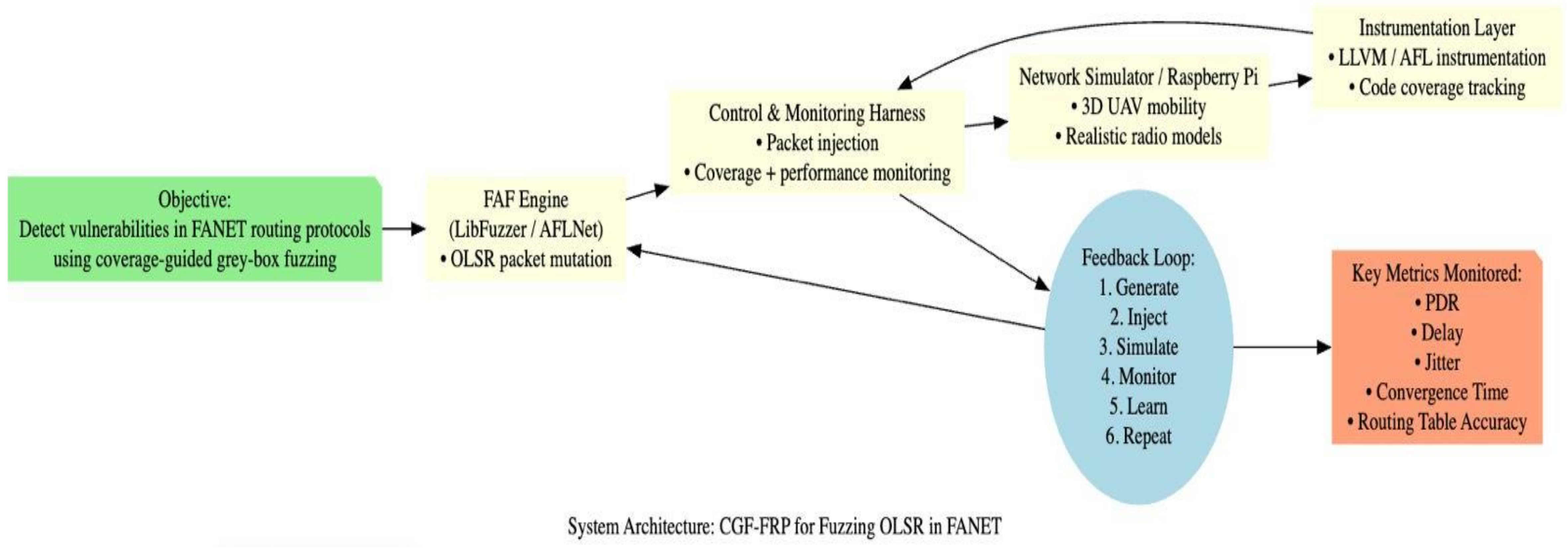


Fig. 3

Proposed Solution

System Workflow:

1. Seed with valid OLSR packets (HELLO, TC)-simulator / RaspberryPi
2. Generate mutated input using coverage-based mutation
3. Inject into live FANET simulation
4. Collect feedback:
   - Code coverage
   - Crash logs
   - Performance degradation
5. Add interesting inputs back to corpus- done by fuzzing tool

Approach:

- **LibFuzzer**: Module-level fuzzing
- **AFLNet**: Full-system fuzzing via network socket simulation

Challenges in FANET Routing protocol fuzzing

FANET Challenge	Fuzzer Adaptation
3D mobility, rapid topology changes	Topology-aware mutations
Stateful protocol behavior	OLSR-specific state model
Distributed node impact	Aggregated coverage from all UAVs
Non-crashing logic flaws	Impact-aware monitoring PDR, Delay, etc.

[5] S. A. Hasan, M. A. Mohammed, and S. K. Sulaiman, “Flying Ad-Hoc Networks (FANETs): Review of Communications, Challenges, Applications, Future Direction and Open Research Topics,” ITM Web of Conferences, vol. 64, Art. no. 01002, Jul. 2024, doi: 10.1051/itmconf/20246401002.

Performance metrics

- **Packet Delivery Ratio (PDR):** Measures how reliably data packets reach the destination. Drops if routing is disrupted by fuzzing.
- **End-to-End Delay:** Average time taken for a packet to reach from source to destination. Increases if routes are incorrect or unstable.
- **Routing Overhead:** Amount of control packets compared to data packets. Increases if fuzzing triggers frequent updates or unnecessary floods.
- **Throughput:** Total data received per unit time. Decreases if the network becomes unstable due to invalid routes.
- **Routing Table Accuracy:** Measures correctness of routing tables. Fuzzed messages can inject false or outdated information.
- **Convergence Time:** Time taken to establish correct routes after topology changes. Increases if fuzzing causes instability or delays in recovery.
- **Jitter**: Variation in packet arrival time. Higher jitter indicates route inconsistency or frequent changes.
- **Code Coverage:** Measures the percentage of code paths (functions, branches, or lines) executed during fuzzing.

J. Carvajal-Rodríguez, W. Moposita, C. Tipantuñna, L. F. Urquiza and D. Vega-Sanchez, "FANET Networks: Analysis of Routing Protocols," 2024 IEEE Eighth Ecuador Technical Chapters Meeting (ETCM), Cuenca, Ecuador, 2024, pp. 1-6, doi: 10.1109/ETCM63562.2024.10746121. keywords: {Performance evaluation;Analytical models;Solid modeling;Protocols;Autonomous aerial vehicles;Throughput;Ad hoc networks;Topology;Communication networks;Vehicle dynamics;FANET;NS-3;UAV;OLSR;DSDV;AODV;DSR;AOMDV},

F. Horváth, **B. Vancsics**, **L. Vidács**, **Á. Beszédes**, **D. Tengeri**, **T. Gergely**, and **T. Gyimóthy**, "Test suite evaluation using code coverage based metrics," in *Proc. 14th Symp. Programming Languages and Software Tools (SPLST'15)*, Tampere, Finland, Oct. 2015, pp. 46–60.

Time Line

References

[1] Malviya, Vikas Kumar, et al. “Fuzzing Drones for Anomaly Detection: A Systematic Literature Review.” *Computers & Security*, vol. 148, 2025, Article 104157, https://doi.org/10.1016/j.cose.2024.104157.
[2] Rudo, David, and Kai Zeng. “Consumer UAV Cybersecurity Vulnerability Assessment Using Fuzzing Tests.” *CoRR*, 9 Aug. 2020, https://doi.org/10.48550/arXiv.2008.03621.
[3] Bekmezci, Ilker, Özgür Koray Sahingöz, and Şamil Temel. “Flying Ad-Hoc Networks (FANETs): A Survey.” *Ad Hoc Networks*, vol. 11, no. 3, May 2013, pp. 1254–1270. https://doi.org/10.1016/j.adhoc.2012.12.004
[4]Li, W., Shi, J., Li, F., Lin, J., Wang, W., & Guan, L. (2022, May). μAFL: Non-intrusive feedback-driven fuzzing for microcontroller firmware. *In Proceedings of the 44th International Conference on Software* Engineering (pp. 1–12). ACM. https://doi.org/10.1145/3510003.3510208
[5] S. A. Hasan, M. A. Mohammed, and S. K. Sulaiman, “Flying Ad-Hoc Networks (FANETs): Review of Communications, Challenges, Applications, Future Direction and Open Research Topics,” ITM Web of Conferences, vol. 64, Art. no. 01002, Jul. 2024, doi: 10.1051/itmconf/20246401002.
[6]M. Erdelj, E. Natalizio, K. R. Chowdhury, and I. F. Akyildiz, “Help from the sky: Leveraging UAVs for disaster management,” IEEE Pervasive Comput., vol. 16, no. 1, pp. 24–32, Jan. 2017, doi: 10.1109/MPRV.2017.11
[7] Z. Sun, P. Wang, M. C. Vuran, M. Al-Rodhaan, A. Al-Dhelaan, and I. F. Akyildiz, “BorderSense: border patrol through advanced wireless sensor networks,” Ad Hoc Netw., vol. 9, no. 3, pp. 468–477, May 2011, doi: 10.1016/j.adhoc.2010.09.008
[8] C. Beaman, M. Redbourne, J. D. Mummery, and S. Hakak, “Fuzzing vulnerability discovery techniques: Survey, challenges and future directions,” Computers & Security, vol. 120, p. 102813, 2022, doi: 10.1016/j.cose.2022.102813
[9] Code Intelligence, “Product – CI Fuzz | Code Intelligence.” Accessed: July 25, 2025. [Online]. Available: https://www.code-intelligence.com/product-ci-fuzz

References

[10]M. A. Khan, A. Safi, I. M. Qureshi and I. U. Khan, "Flying ad-hoc networks (FANETs): A review of communication architectures, and routing protocols," 2017 First International Conference on Latest trends in Electrical Engineering and Computing Technologies (INTELLECT), Karachi, Pakistan, 2017, pp. 1-9, doi: 10.1109/INTELLECT.2017.8277614. keywords: {Routing protocols;Ad hoc networks;Routing;Unmanned aerial vehicles;Computer architecture;Robustness;Ad-hoc Network;UAVs;FANETs;Routing},
[11]Alkhatieb,& Felemban, Emad & Naseer, Atif. (2020). Performance Evaluation of Ad-Hoc Routing Protocols in (FANETs). 1-6. 10.1109/WCNCW48565.2020.9124761.
[12] Mohammed J. Almansor, Norashidah Md Din, Mohd Zafri Baharuddin, Ode Ma, Huda M. Alsayednoor, Mahmood A. Al-Shareeda, and Ahmed J. Al-asadi, "Routing protocols strategies for flying Ad-Hoc network (FANET): Review, taxonomy, and open research issues," Alexandria Engineering Journal, vol. 109, pp. 553–577, Dec. 2024, doi: 10.1016/j.aej.2024.09.032