Aquib Aquil

Jaipur, India
7070418638 | aquibaquil41@gmail.com | github.com/AquibAquil

Profile

Third-year Computer Science (Cybersecurity) student at Manipal University Jaipur with hands-on experience in applied cryptography, secure backend systems, privacy-preserving computation, and autonomous agent design. Built an autonomous multi-strategy options allocation agent for the Alpaca AI Trading Agents Hackathon, a self-expiring data clean room on the Atsign platform, and a JNI-based Android cryptography backend. Active in vulnerability research and coordinated disclosure, with a credited GitHub Security Advisory and a private disclosure to the Frappe Security Team.

Education

Manipal University Jaipur Aug 2024 – Jul 2028
B.Tech in Computer Science (Cybersecurity)

Projects

- **Multi-Strategy Options Allocation Agent** | *Python, Alpaca MCP, Groq (Llama 3.3 70B), pytest* | github
  - Built an autonomous agent that runs three QQQ options strategies concurrently and decides at runtime which deserve capital; the live experiment is whether adaptive AI allocation beats equal weighting across the same three strategies, reported regardless of sign.
  - Separated allocator, challenger, hard risk gates, and sizing as independent concerns: the LLM proposes shares of a risk budget in structured JSON and never a contract count, and parsers reject any payload smuggling a `confidence` or `probability` field, so no made-up number can enter sizing.
  - Interfaced with Alpaca through their official MCP server (v2.3.0) as a Python MCP *client* — not a chat client — keeping one persistent stdio session on a background asyncio loop for an unattended agent.
  - Enforced "hold last valid allocation" at every failure boundary (model call, broker call, unconfirmed order status), with each cycle logged to `jsonl` for post-hoc audit; most of the orchestrator test suite is failure paths.
  - Wrote 18 test files exercising the full flow offline against synthetic fat-tailed price series and a captured QQQ chain, so the maths and failure branches are checkable without credentials. Built for the Alpaca AI Trading Agents Hackathon, Aug 28 – Sep 4 2026.
- **Self-Expiring Data Clean Room** | *Dart, Flutter, Atsign Platform SDK* | github
  - Built a privacy-preserving two-party data clean room whose cryptographic keys are destroyed on mutual stakeholder confirmation, leaving an immutable audit receipt — the access becomes unrecoverable, not just unused.
  - Enforced inter-company isolation *structurally*: company agents contain zero `atClient.get()` / `lookup()` calls, so the privacy boundary is a property of the code's shape rather than a permission flag a bug could disable.
  - Implemented salted SHA-256 set intersection as a Private Set Intersection floor over hashed identifier submissions, with documented production upgrade path to ECDH/OPRF-based PSI over NoPorts tunnels.
  - Designed the multi-agent topology (2 stakeholders, 2 company agents, clean-room compute, retirement policy) in Atsign AI Architect before implementation, with the deployed system matching the blueprint node-for-node.
  - Shipped Dart CLI agents compiled to native single-binary executables plus a Flutter stakeholder app; verified end-to-end on Windows desktop and a real Android device. Built for the Atsign Hackathon 2026.
- **Android Cryptography Backend using JNI** | *Dart, Java, C/C++ (JNI)*
  - Designed a JNI-based Android backend bridging Dart (Flutter) and Java Cryptography Architecture (JCA).
  - Implemented cryptographic primitives including AES (CBC, GCM, CTR), RSA (PKCS1, OAEP), and ECDH.
  - Built a cross-language data pipeline handling memory allocation and lifecycle management across layers.
  - Developed benchmarking framework analyzing cold-start vs steady-state latency and JNI overhead.
  - Reduced latency from 200ms cold-start to 1–4ms steady-state execution.
  - Ensured safe JNI reference handling and efficient memory usage.

Technical Skills

Languages: C++, Python, Java, Dart
Systems & Backend: REST APIs, System Design, Model Context Protocol (MCP), Autonomous Agents
Cryptography: AES (CBC, GCM, CTR), RSA (PKCS1, OAEP), ECDH, Secure Key Exchange, Private Set Intersection
Technologies: JNI, Java Cryptography Architecture (JCA), Flutter, Atsign Platform SDK, Alpaca MCP, pytest
Concepts: Memory Management, Cross-Language Integration, Performance Benchmarking, Privacy-Preserving Computation, LLM System Design

Achievements

- Received a GitHub Security Advisory (GHSA) credit for identifying and responsibly reporting a denial-of-service vulnerability in `python-multipart` (CVE-2026-40347, GHSA-mj87-hwqh-73pj).
- Privately disclosed an authenticated boolean-blind SQL injection in the Frappe Framework query layer (`frappe.client.get_list`, `validate_order_by_and_group_by` denylist bypass via `IF()` + `REGEXP`) to the Frappe Security Team under coordinated disclosure (internal ref FRP-2026-0001).