GEUR-QKD Quantum Cryptography Simulator

Full Project Documentation

Version: 1.0 | License: MIT

1 Introduction

This project implements a **Generalized Entropic Uncertainty Relation (GEUR)**-based **Quantum Key Distribution (QKD)** simulator in Python. It models, tests, and visualizes the core processes of a composably secure quantum cryptographic workflow:

- Quantum bit preparation and transmission
- Basis selection and sifting
- Error estimation (QBER)
- Advantage distillation (AD)
- Error correction (EC) cost tracking
- Privacy amplification (PA) with a 2-universal Toeplitz extractor (design)
- Authentication with Wegman–Carter MAC (simulated)
- Finite-key ε-budget and composable accounting

The simulator is a *toy model* for education and experimentation—useful for understanding realistic trade-offs between security (Eve's information), error rates, authentication, and key length.

2 Objectives

1. Implement a working GEUR-based QKD simulator with adaptive tuning.
2. Incorporate a composable ε-budget and surface it in the CLI summary.
3. Add authentication using a Wegman–Carter MAC model with configurable $\varepsilon_{\text{auth}}$.
4. Provide a realistic PA mechanism (Toeplitz hashing design, seed handling, min-entropy inputs).
5. Offer clear plots/outputs and a simple CLI for exploration, sweeps, and tuning.

3 Development Timeline and Setbacks

Phase 1 – Core Simulation

Built the foundation (`geur_qkd_sim.py`) with randomized qubit transmission, basis matching (Z/X/Y), QBER calculation, advantage distillation (AD), error correction (EC) leakage tracking, and key-rate computation.

Setback: Unstable QBER and zero-length keys due to missing caps.
Fix: Introduced error-thresholds and bounded leakage; stabilized outputs.

Phase 2 – Parameter Tuning

Implemented a tuner to search for configurations matching target QBER/Eve bounds. Records results to CSV in `runs/`.

Setback: First pass returned `NoneType` (empty candidate selection).
Fix: Added range checks, robust filtering, and deterministic selection rules.

Phase 3 – Composable Security Model

Added a full ε-budget line itemizing: Parameter Estimation ($\varepsilon_{\mathrm{PE}}$), Error Correction ($\varepsilon_{\mathrm{EC}}$), Privacy Amplification ($\varepsilon_{\mathrm{PA}}$), Authentication ($\varepsilon_{\mathrm{AUTH}}$), and Abort ($\varepsilon_{\mathrm{ABORT}}$).

Setback: Repeated `final_len` keyword and incomplete f-strings.
Fix: Standardized summary printing and argument passing.

Phase 4 – Wegman–Carter Authentication

Integrated simulated authentication of the classical channel with Wegman–Carter MAC. CLI options:

```
--auth mac
--auth-eps 1e-20
```

MAC tag length is derived via $\varepsilon_{\mathrm{auth}} \approx 2^{-\mathrm{tag\_bits}}$.

Setback: `NameError: ap is not defined`.
Fix: Corrected CLI parser variable from `ap` to `p`.

Setback: `auth_tag_bits` undefined.
Fix: Added on-the-fly computation from $\varepsilon_{\mathrm{auth}}$.

Phase 5 – Privacy Amplification (Toeplitz)

Added a design path for 2-universal Toeplitz hashing (seeded extractor) to compress the reconciled key using min-entropy estimates from GEUR/QBER. Interface cleanly separates seed generation, matrix application, and entropy accounting.

Phase 6 – Sweep and Export

Added `-sweep mini` to collect performance statistics across small grids, and `-export-best` to save the top configuration as `runs/best_config.json`.

Setback: `NameError: run_quick_sweep not defined`.
Fix: Implemented a minimal helper and standardized CSV output.

4 Composable Security and ε-Budget

The simulator surfaces the global ε-budget:

$$\varepsilon_{\mathrm{total}} = \varepsilon_{\mathrm{PE}} + \varepsilon_{\mathrm{EC}} + \varepsilon_{\mathrm{PA}} + \varepsilon_{\mathrm{AUTH}} + \varepsilon_{\mathrm{ABORT}}.$$

Typical defaults (editable via CLI) are:

$$\varepsilon_{\mathrm{PE}} = \varepsilon_{\mathrm{EC}} = \varepsilon_{\mathrm{ABORT}} = 2 \times 10^{-10}, \quad \varepsilon_{\mathrm{PA}} = 10^{-15}, \quad \varepsilon_{\mathrm{AUTH}} = 10^{-20}.$$

The summary prints an explicit ε-line to support composable interpretations.

5 Command Reference

Option	Description
`-profile target10`	Load preset system parameters
`-plots -band`	Enable plots and uncertainty bands (if implemented)
`-auth none|mac`	Disable/enable Wegman–Carter MAC
`-auth-eps <val>`	Set authentication $\varepsilon_{\mathrm{auth}}$ (e.g., `1e-20`)
`-eps-pa <val>`	Set privacy amplification $\varepsilon_{\mathrm{PA}}$
`-sweep mini`	Run a small configuration sweep; write CSV to `runs/`
`-tune`	Activate parameter tuner
`-tune-rounds <N>`	Tuning rounds per candidate (e.g., 8000 or 12000)
`-tune-max-qber <q>`	QBER cap (e.g., `0.18`)
`-tune-target <e>`	Target Eve knowledge (e.g., `0.10`)
`-tune-max-eve <e>`	Hard Eve cap filter (e.g., `0.12`)
`-export-best`	Save best config to `runs/best_config.json`

6 Example Sessions and Observations

Tuned weighted-AD regime (low Eve, moderate key)

```
[tune] candidates kept: 1072
[tune] QBER kept range: 0.091  0.180 (median 0.132)
Best config: {
  "P_BASIS":[0.88,0.06,0.06],
  "CHANNEL_ERROR":0.004,
  "USE_AD":true, "USE_AD_WEIGHTED":true,
  "AD_BLOCK_SIZE":4, "AD_SCORE_THRESH":2.8,
  "EC_EFFICIENCY":1.05, "AD_PARITY_LEAK":false,
  "EVE_STRENGTH":0.2
}
Metrics: { "eve":0.0836, "final_len":417, "sift_len":6323,
          "qber":0.095, "leak":539.30 }
```

Composable ε-budget print and MAC

```
-budget: PE=2.00e-10 + EC=2.00e-10 + PA=1.00e-15 + Auth=1.00e-20 + Abort=2.00e-10 ->
    _total=6.00e-10
Authentication: WegmanCarter MAC (_auth=1.00e-20, tag=67 bits)
===== GEUR-QKD TOY SIMULATION =====
Total rounds sent: 20000
Sifted key length (pre-AD): 6632
Raw error rate (pre-AD): 10.87%
Eve knowledge used for PA: 25.44%
Final key length after PA: 37 bits
```

7 Security Principles and Scope

Kerckhoffs's Principle. The simulator assumes the *protocol, code, and parameters are public*; only the random seeds and final keys are secret. This mirrors good practice in real cryptosystems and aligns with composable security treatments.

Device model. This is *not* a full device security proof. Side channels, detector control, and calibration biases are out-of-scope. The model focuses on logical post-processing under GEUR-style min-entropy bounds.

Authentication. Wegman–Carter MAC is modeled as information-theoretic authentication; tag bits are burned from pre-shared key material proportional to $\varepsilon_{\text{auth}}$.

8 Lessons Learned

- Minimal viable core first; add features incrementally.
- CLI robustness matters; small parser mistakes cause big headaches.
- Finite-key accounting tangibly changes key lengths; surfacing the ε-line helps reason about trade-offs.
- Weighted AD often improved Eve leakage at a cost in kept bits; classic AD kept more bits but increased Eve's knowledge.
- Grid searches (`-tune`, `-sweep`) are essential to map stable operating regions.

9 How to Run

Prerequisites

Python 3.10+ recommended. Standard library only (no external deps for core runs).

Quick start

```
# Baseline run
python geur_qkd_sim.py --profile target10

# Enable composable auth + PA epsilon choices
python geur_qkd_sim.py --profile target10 --auth mac --auth-eps 1e-20 --eps-pa 1e-15

# Mini sweep to CSV
python geur_qkd_sim.py --profile target10 --sweep mini

# Tuning toward a target Eve and capped QBER, then export best
python geur_qkd_sim.py --profile target10 --tune \
  --tune-target 0.10 --tune-max-qber 0.18 --tune-rounds 8000 --export-best

# Re-run with exported best config
python geur_qkd_sim.py --profile target10 --override @runs/best_config.json
```

10 Future Work

- Implement the full Toeplitz extractor (bit-matrix apply, seed management).
- Add physical channel models (loss, decoy states, detector mismatch).
- Provide Qiskit back-end for physical qubit emulation.
- Multi-party/entanglement-based scenarios and MDI-QKD variants.
- Automated ε-budget validation across batches with plots.

11 References

Foundational QKD

- Bennett, C. H., Brassard, G. (1984). Quantum cryptography: Public key distribution and coin tossing.
- Bennett, C. H. (1992). Quantum cryptography using any two nonorthogonal states. *Phys. Rev. Lett.* 68, 3121.

Composable Security and Entropy

- Renner, R. (2005). Security of Quantum Key Distribution. PhD Thesis, ETH Zurich.
- Tomamichel, M., Renner, R. (2011). Uncertainty Relation for Smooth Entropies. *Phys. Rev. Lett.* 106, 110506.
- Tomamichel, M. et al. (2009). A fully composable framework for quantum cryptography. *New J. Phys.* 11, 045009.
- König, R., Renner, R., Schaffner, C. (2009). The operational meaning of min- and max-entropy. *IEEE TIT* 55, 4337–4347.
- Coles, P. J., Berta, M., Tomamichel, M., Wehner, S. (2017). Entropic uncertainty relations and their applications. *Rev. Mod. Phys.* 89, 015002.

Authentication and PA

- Wegman, M. N., Carter, J. L. (1981). New hash functions and their use in authentication and set equality. *JCSS* 22, 265–279.
- Krawczyk, H. (1994). LFSR-based hashing and authentication. In *CRYPTO '94*, LNCS 839.
- Gottesman, D., Lo, H.-K., Lütkenhaus, N., Preskill, J. (2004). Security of QKD with imperfect devices. *QIC* 4, 325–360.

Surveys and Practical

- Scarani, V. et al. (2009). The security of practical QKD. *Rev. Mod. Phys.* 81, 1301.

12 License (MIT)

MIT License