**Practical projects**

1. Pick your favorite cloud storage system (e.g., Google docs, Amazon S3, Dropbox) and turn it into verifiable. Report performance results.
2. Implement a cloud system that performs a keyword search obliviously. Use an ORAM algorithm and a searchable encryption algorithm for this task and compare the results.
3. Pick, implement and compare three ORAM algorithms (for the comparison you can benchmark the time it takes to READ and WRITE obliviously).
4. Extend POR with an implementation for efficient recovery (talk to me for the paper).
5. Compare the performance of three verifiable cloud systems (for the comparison you can pick a benchmark task, e.g., polynomial evaluation, matrix multiplication):
    a. Pinocchio (see SSP 2013)
    b. Pantry (see SOSP 2013)
    c. SNARKs for C (see CRYPTO 2013)
6. Use a TPM chip to implement more efficient verifiable storage or verifiable computation (verification at the server side).
7. Implement hashing or verifiable computation on FPGAs and report performance results.
8. Build a parallel implementation of existing verifiable computation schemes (e.g., Pinocchio).
9. Replace the hash function used in the Bitcoin source to implement Merkle hash trees with a lattice-based hash function (e.g., SWIFFT).
10. Build a private email service where, apart from just storing encrypted emails, it will allow for searching encrypted emails.
11. Find, exploit (be ethical), and fix security vulnerabilities of existing cloud services.

**Theoretical projects**

1. Study the streaming authenticated data structures paper (EUROCRYPT). Try to replace the hash function used with the SWIFFT hash function (FSE).
2. Devise a verifiable scheme for ranking operations (advanced).
3. Devise a verifiable scheme for shortest path computations in general graphs (advanced).