Stony Brook University

HARVEY MUDD COLLEGE

UC SANTA CRUZ BaskinEngineering

Center for Research in Systems and Storage

NSF

Boots on the Ground with MP-SPDZ: A Developer's Perspective

Christopher Smith, Akshat Sharma, Ethan Lotan, Sadhvik Vutkur, Gaurav Kulhare, Ethan Miller†, Geoff Kuenning*, Erez Zadok

*Stony Brook University, UC Santa Cruz†, Harvey Mudd College**

Context

PhD candidate interested in decentralized key management

Distributed trust + MPC = match made in heaven

Too many functionalities, not enough time… MP-SPDZ to the rescue!

SysteMPC Lightning Talk

Prototyping is a Breeze!

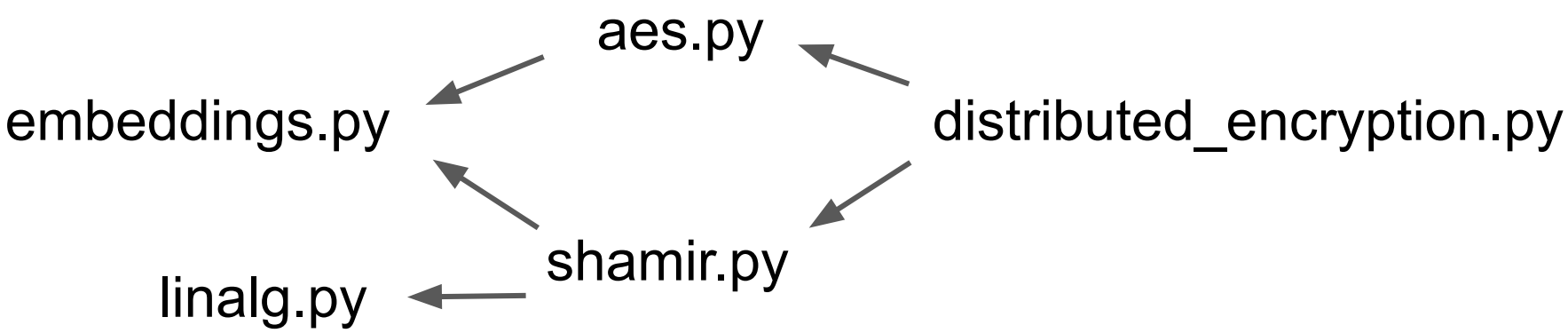
- Client interface + configurable executable makes it easy to spin up and “hook into” MPC parties as subprocesses

```
self.mpspdz_process = subprocess.Popen(
    [os.path.join(MP_SPDZ_DIR, "mascot-party.x"),
        '--player', f'{self.party_id}',
        '--portnumbase', f'{self.MPC_PORT_BASE}',
        '--ip-file-name', 'network.config',
        '--output-file', '.',
        '--nparties', f'{n}',
        # '--verbose',
        'root_key_gen'], # name of the .py MPC script
    stdout=sys.stdout,
    stderr=sys.stderr,
    cwd=MP_SPDZ_DIR
)

# connect as client to MP-SPDZ subprocess
client = Client(['localhost'], self.MPC_PORT_BASE + self.party_id, self.party_id)
root_key_share = client.receive_plain_values()
```

Prototyping is a Breeze!

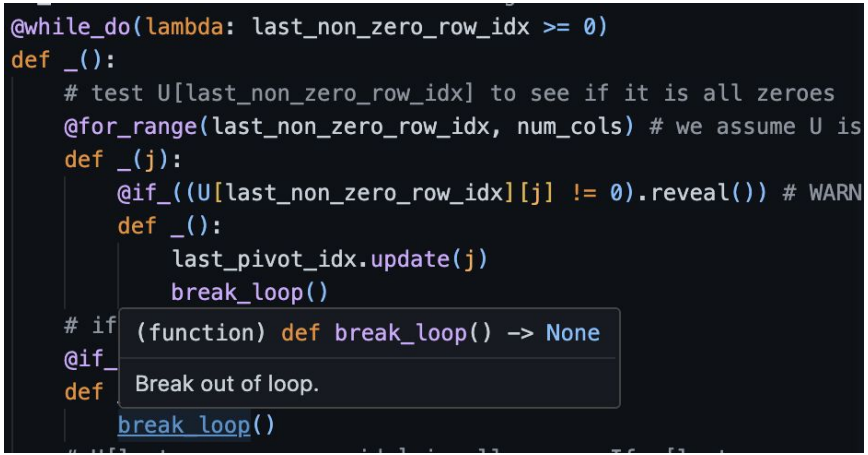
- Leverage Python modules and OOP abstractions to build out increasingly complex functionalities



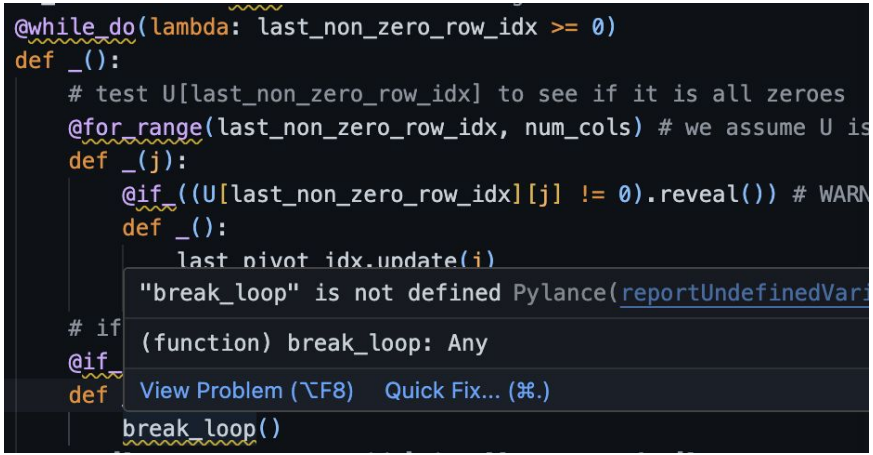
Prototyping is a Breeze!

- Recommendation: explicitly import MP-SPDZ compiler and write .py files instead of .mpc files for full IDE support

✅



❌



Now the Quirks…

Quirk: No Secret Branching

Surprise! Predicate of an if statement cannot depend on any secret values

```
@if_((U[last_non_zero_row_idx][j] != 0).reveal()) # WARNING: leaks info about U
```

Root cause has to do with circuit model of computation…

Solution proof-of-concept implemented only for semi-honest case

Secure Multiparty Computation with Free Branching

Eurocrypt '22

Aarushi Goel[1], Mathias Hall-Andersen[2], Aditya Hegde[1], and Abhishek Jain[1]

[1]Johns Hopkins University, {aarushig,ahegde,abhishek}@cs.jhu.edu
[2]Aarhus University, ma@cs.au.dk

Quirk: No Secret Indexing

Indices to Container data structures (e.g., Array, Matrix) need to be cleartext values

```
a = Array(1, sint).assign_all(1)
b = a[sint(0)]
print_ln("Compiler.exceptions.CompilerError: need cleartext index")
```

Root cause: memory accesses by secret addresses seem to require ORAM, which is expensive. See https://github.com/data61/MP-SPDZ/issues/29

Quirk: Field Embeddings

- Active security for SPDZ family depends on large field, e.g. GF(2^{40})
- But some functionalities (e.g., AES, Shamir's SS) might require operations over a smaller field like GF(2^8).
- MP-SPDZ basic types (e.g., `sgf2n`) only use the larger field.

=> Need a **field embedding** GF(2^8) $\hookrightarrow$ GF(2^{40})

- Unwelcome shock to developers without an algebra background
- Embeddings don't always exist!
- Somewhat unavoidable when using arithmetic circuits, but we could provide more support by precomputing + implementing common embeddings in a MP-SPDZ "standard library

Final Thoughts

- Quirks not specific to MP-SPDZ!
  - Secret branching + field embeddings arise from arithmetic circuits
  - Secret indexing requires ORAM
- Some may consider these quirks “solved problems”... but clearly these quirks are still hanging around in practice
- Appetite for front-end alternatives to Python?
- Appetite for an MP-SPDZ standard library?