Introduction

- **Compartmentalization**
  Breaking software into compartments is a well-known security technique for security, it will restrict the impact of errors in a part of the program.
- **Debugging**
  Compartmentalization makes debugging harder, as the developer might need to access all compartments with different privileges. Remote debugging is also required if the system is distributed on separate machines.

Motivation

- **Research at Penn**
  This is part of a research project at Penn focusing on making compartmentalization of software easier through a new tool-chain we are developing.
- **Current Status**
  There are no existing solutions for debugging compartmentalized software in our research. We want a debugger that is compartment-aware and does not make compromises.

Debugger - Front End

- **An Example**
  The example output of switching between two clients and running some basic debugging command could be captured from the above figure. As we could see, the interface is working and debugging two compartments.

Debugger - Back End

- **Back-End Design**
  We used the GDB’s standard remote protocol at the backend so the client could talk remotely to the stub we developed. And the stub will run the debugging instructions on distributed compartments.

Further References

For more informations about the debugger and libcompart, please see [https://cobre.cis.upenn.edu/](https://cobre.cis.upenn.edu/)

Or contact me at junyong@seas.upenn.edu