# Bulldogs Team

Team Member: Dr. Rui Zhu
(Kettering University)


Mentor: Dr. John Holmen
(Oak Ridge National Laboratory)

Mentor: Yvonne Phillips
(Morehouse College)

- Target Course(s): CS425 Parallel Programming and Algorithms, CS457 Wireless and Mobile Security
- Goal:
  - Integrating HPC with Cybersecurity, Cryptography, and Machine Learning to develop curriculums
  - Identify applicable HPC resources from ORNL/wider HPC community and develop course descriptions
  - Create and refine course schedules, hands-on labs, etc.
- GitHub Repo: https://github.com/ruikobe/KetteringTeamFacHack23
- Theme Song: George Thorogood & The Destroyers - Bad To The Bone

# Syllabus

GitHub link: https://github.com/ruikobe/KetteringTeamFacHack23/blob/main/CS425%20Syllabus.pdf

**CS-425 Parallel Programming and Algorithms**

The CS-425 course introduces you to the foundations of parallel computing including the principles of parallel algorithm design, analytical modeling of parallel programs, programming models for shared- and distributed-memory systems, parallel computer architectures, along with numerical and non-numerical algorithms for parallel systems. The course will include material on emerging multicore hardware, shared-memory programming models, message passing programming models used for cluster computing, data-parallel programming models for GPUs, and problem-solving on large-scale clusters using MapReduce. A key aim of the course is for you to gain a hands-on knowledge of the fundamentals of parallel programming by writing efficient parallel programs using some of the programming models that you learn in class.

**Prerequisites**: CS-401 Programming Methods for Data Science
CS-231 Programming Paradigms

**Class Schedule**: 240 minutes of lecture

**Textbooks**
• Multicore and GPU Programming: An Integrated Approach 2nd Edition, Gerassimos Barlas (2022)

# Sample Exercises

1. Create accounts and get access to Kettering University High Performance Computer Cluster (KU-HPC). We will be using these machine for all our assignments and projects.

2. Compile hello_mpi.c and submit a job on both machines.

3. Write a simple MPI program that does the following: generate a random integer array, with 100 elements per process, on all processes. hint: initialize the RNG using some function of the rank of the process.

# Potential Science Gateways

https://catalog.sciencegateways.org/#/home/8895729113541766680-242ac119-0001-012?pageName=Jupyter%20Notebook

https://catalog.sciencegateways.org/#/home/2727092443435831786-242ac116-0001-012?pageName=SMALTR%20Gateway

https://catalog.sciencegateways.org/#/home/8870203472310365720-242ac11c-0001-012?pageName=CILogon

https://catalog.sciencegateways.org/#/home/7684577338378415640-242ac11d-0001-012?pageName=Center%20for%20Applied%20Internet%20Data%20Analysis%20(CAIDA)