Secure memory management is a vital low-level component of a security infrastructure. Capability Hardware Enhanced RISC Instructions (CHERI) is a specification of extensions to standard ISAs that provides precise memory protection and compartmentalisation to mitigate memory vulnerabilities such as a buffer overflow attack. CHERI also defines capabilities: secure pointers that contain an address and metadata including bounds, permissions and a validity tag to enforce fine-grained control of memory access rights.

Research done at the University of Glasgow that executed benchmarks against popular CHERI allocators for the Cheri-BSD port has identified vulnerabilities and unexpected behaviour, leading to security flaws such as allowing access to previous data from old capabilities. I propose to develop a CHERI allocator in C that defends or mitigates these vulnerabilities. Artifacts produced would include source code, documentation and shared and static libraries. I would also be interested in researching the feasibility of using the Cheri-Rust extension of Rust to develop a CHERI allocator, due to the language's increased memory and type safety. This would evaluate whether Cheri-Rust is suitable for creating allocators given the inherently unsafe nature of memory management, and if so whether it provides any significant benefits over languages such as C and C++.

This research would attempt to fix the vulnerabilities found during the benchmarks, which has not yet been carried out. The motivation for this project is further justified since Cheri-Rust could create a safer allocator if it is possible to use, which would help build more robust CHERI implementations. Although there is currently a Rust compiler for CHERI being developed that forks from the existing Rust LLVM compiler, there are no allocators that are either written in Cheri-Rust or fully satisfy all the security benchmarks, making this research both unique and existing within a cutting-edge and important field of cyber-security.