Eric Kelly (ekelly@ccs.neu.edu) & Francis Nimick (fnimick@ccs.neu.edu)
Extensible Systems Memo 2
Project: The Silver Searcher

The Silver Searcher (Ag) is a fast, feature-complete replacement for Ack. It is a tool for quickly parsing a large codebase, looking for strings which match the input regular expression. It is specialized to searching code repositories: it ignores version control directories such as '.git' and also does not search files in .gitignore, .hgignore or similar files. Its output is also tailored to developers: it prints line numbers with every result by default. The Silver Searcher is implemented in C and optimized for speed. It also takes full advantage of modern computing hardware by using threads for parallel searching.

The Silver Searcher is available both as a pre-built binary and as source. To build from source requires the following dependencies: automake, pkg-config, pcre, and xz. It also has an optional dependency on Cram for testing. Once these dependencies are installed, building the Silver Searcher from source involves running the build.sh script in the root of the repository, which automatically configures and "makes" the project. Running the Silver Searcher requires the following command format: `ag [file-type] [options] PATTERN [PATH]`. It does make certain assumptions about the files it iterates through. For instance, it assumes that files being searched have appropriate extensions, e.g. that a .html file contains HTML. For speed, it also assumes that the repository is not being actively modified at the time of the search: if a file changes length while being searched, undefined behavior may result.

The Silver Searcher's repository is clearly organized. The root of the Silver Searcher codebase has files to assist in building the Silver Searcher, files for informational purposes (such as the README), as well as several subfolders - one of which is the src subfolder. This is the folder most relevant to our project as it contains the C source code of the Silver Searcher. The tests subfolder is also significant, as any tests we add will likely be placed in this directory. Inside the src folder, program code is modularized. There are nine submodules: decompress, ignore, lang, log, options, print, scandir, search, and util. There is also one "main" module which is the entry point for the program.

Our goal for this project is to decompress and search .zip files. The Silver Searcher can currently search within .tar and .tgz (.tar.gz) files. Searching within zip files would be similarly useful to end users. While the architecture to enable zip decompression exists, it is currently not yet implemented. We will implement the stored (no compression), deflate, and deflate64 zip file formats in Rust. This will involve modifying the existing C code only in the 'decompress' module to call our Rust decompression library, rather than ignoring zip files as it currently does.