

SUMMARY OF BROWN'S DISCLOSURE

- The Disclosure, as provided, cannot generate any target computer language (i.e. "C source code").
- The "Branch Resolution" function does not work.
- The Disclosure does not include a means to distinguish code from data. Brown claimed that he had no algorithmic way to identify code vs. data, and stated that this is not part of the Solution.
- Currently, the Disclosure cannot:
 - process multiple entry points
 - identify high level loops
 - generate "if-then-else" control constructs
 - handle unions
 - resolve machine dependant sub-library routines, segmented addressing or data structures with gaps
 - handle indirect function calls [perhaps does not handle any indirect references], "computed goto" commands [jump tables]
 - resolve or take into consideration the semantics and context of a given routine/program
 - handle high level program structures, return values, or functions identified via pointer
 - handle dynamic situations (as opposed to static structures) that arise in a given program/code

- Currently, the Disclosure can:
 - locate callable entry points to a degree
 - ascertain data types for in-line code (this process is incomplete, however). The Disclosure cannot ascertain data types for out-of-line code, or across subroutine call boundaries.
 - convert machine binary code to an intermediate, machine independent assembly-type of code (but he has not tested this conversion to determine whether it is accurate)
 - identify balanced save/restore of registers. The program identifies "fragments" [unclear precisely what this refers to.]
 - locate branches (i.e. transfers of control) and handles them to a degree, but not completely.
 - identify direct data references and immediate references, and assigns sizes and partial data types
 - unwind branches to expand code (but the Solution does not resolve infinite loops, which present a "catastrophic" problem at present)

- When asked "Just how much of an idea do you have?" Brown responded that he has the idea of producing an abstract assembly language source suitable for representing the kinds of operations which are performed on any processor, without being tied to the architecture of any one of them. Brown has not supplied any documentation regarding the intermediate "abstract assembly language".