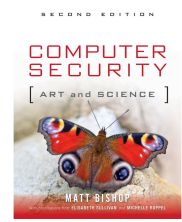
Programming Rules

Appendix H



Outline

- Implementation rules
- Management rules

Implementation Rules 1-4

1. Structure the process so that all sections requiring extra privileges are modules. The modules should be as small as possible and should perform only those tasks that require those privileges.
2. Ensure that any assumptions in the program are validated. If this is not possible, document them for the installers and maintainers, so they know the assumptions that attackers will try to invalidate.
3. Ensure that the program does not share objects in memory with any other program, and that other programs cannot access the memory of a privileged process.
4. The error status of every function must be checked. Do not try to recover unless the cause of the error, and its effects, do not affect any security considerations. The program should restore the state of the system to the state before the process began, and then terminate.

Implementation Rules 5-8

5. If a process interacts with other processes, the interactions should be synchronized. In particular, all possible sequences of interactions must be known and, for all such interactions, the process must enforce the required security policy.
6. Asynchronous exception handlers should not alter any variables except those that are local to the exception handling module. An exception handler should block all other exceptions when begun, and should not release the block until the handler completes execution, unless the handler has been designed to handle exceptions within itself (or calls an uninvoked exception handler).
7. Whenever possible, data that the process trusts and data areas of memory. If data from a trusted source is overwritten with data from an untrusted source, a memory error will occur.
8. Do not use components that may change between the time the program is created and the time it is run.

Implementation Rules 9-12

9. The process must ensure that the context in which an object is named identifies the correct object.
10. When the process finishes using a sensitive object (one that contains confidential information or one that should not be altered), the object should be erased, then deallocated or deleted. Any resources not needed should also be released.
11. Ensure that all array references access existing elements of the array. If a function that manipulates arrays cannot ensure that only valid elements are referenced, do not use that function. Find one that does, write a new version, or create a wrapper.
12. Check the types of functions and parameters.

Implementation Rules 13-16

13. When compiling programs, ensure that the compiler reports inconsistencies in types. Investigate all such warnings and either fix the problem or document the warning and why it is spurious.
14. Check all function and procedure executions for errors.
15. Check that a variable's values are valid.
16. If a trade-off between security and other factors results in a mechanism or procedure that can weaken security, document the reasons for the decision, the possible effects, and the situations in which the compromise method should be used. This informs others of the trade-off and the attendant risks.

Implementation Rules 17-20

17. Check all user input for both form and content. In particular, check integers for values that are too big or too small, and check character data for length and valid characters.
18. Create data structures and functions in such a way that they can be validated.
19. If two operations must be performed sequentially without an intervening operation, use a mechanism to ensure that the two cannot be divided.
20. Describe the legal sequences of operations on a resource or object. Check that all possible sequences of the program(s) involved match one (or more) legal sequences.

Management Rules 1-3

1. Check that the process privileges are set properly.
2. The program that is executed to create the process, and all associated control files, must be protected from unauthorized use and modification. Any such modification must be detected.
3. Configure memory to enforce the principle of least privilege. If a section of memory is not to contain executable instructions, turn execute permission off for that section of memory. If the contents of a section of memory are not to be altered, make that section read-only.

Management Rules 4-6

4. Identify all system components on which the program depends. Check for errors whenever possible, and identify those components for which error checking will not work.
5. Unique objects require unique names. Interchangeable objects may share a name.
6. Use software engineering and assurance techniques (such as documentation, design reviews, and code reviews) to ensure that operations and operands are appropriate.