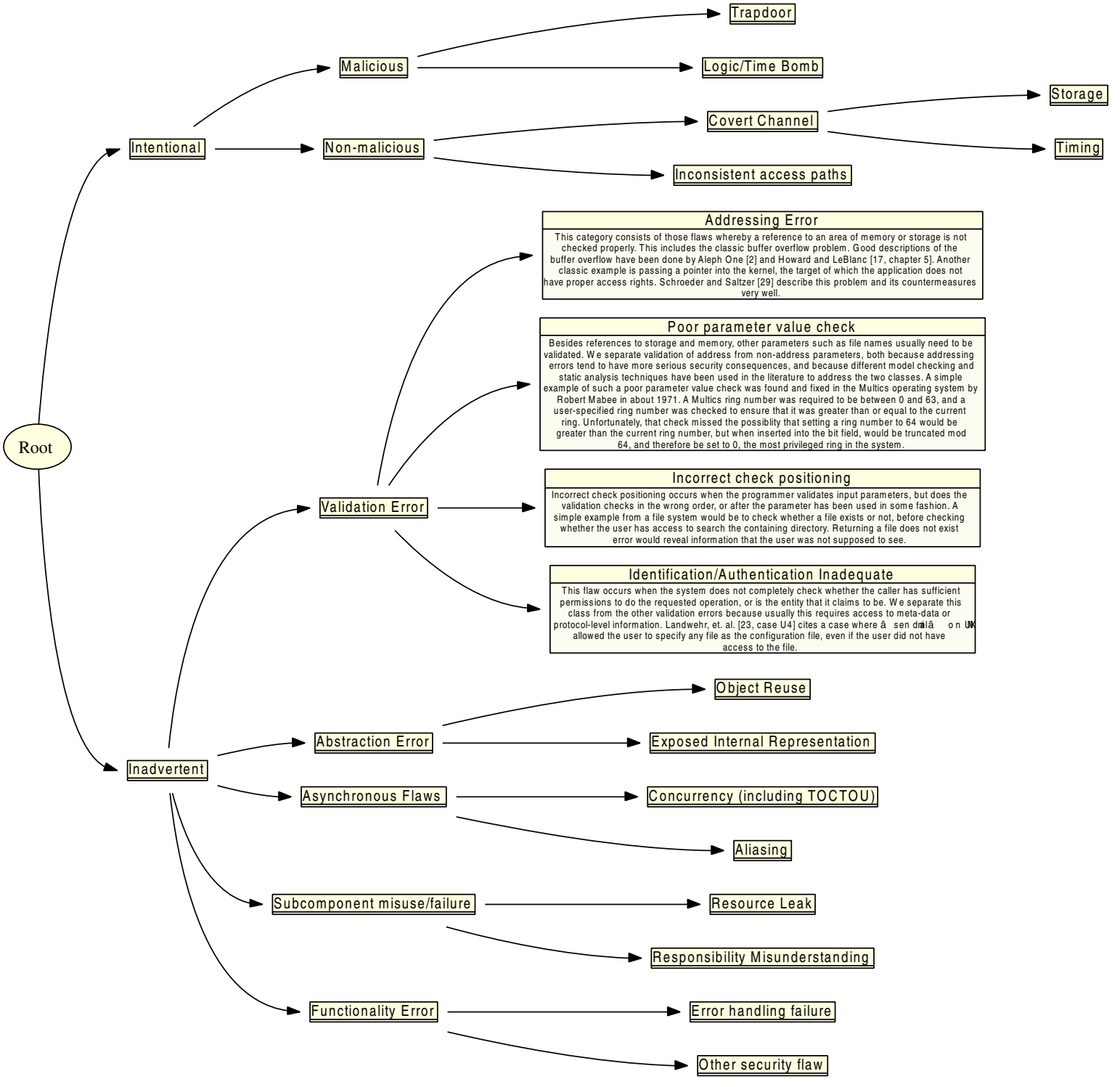
Root
Intentional
Malicious
Trapdoor
Logic/Time Bomb
Non-malicious
Covert Channel
Storage
Timing
Inconsistent access paths
Inadvertent
Validation Error
Addressing Error
This category consists of those flaws whereby a reference to an area of memory or storage is not checked properly. This includes the classic buffer overflow problem. Good descriptions of the buffer overflow have been done by Aleph One [2] and Howard and LeBlanc [17, chapter 5]. Another classic example is passing a pointer into the kernel, the target of which the application does not have proper access rights. Schroeder and Saltzer [29] describe this problem and its countermeasures very well.
Poor parameter value check
Besides references to storage and memory, other parameters such as file names usually need to be validated. We separate validation of address from non-address parameters, both because addressing errors tend to have more serious security consequences, and because different model checking and static analysis techniques have been used in the literature to address the two classes. A simple example of such a poor parameter value check was found and fixed in the Multics operating system by Robert Mabee in about 1971. A Multics ring number was required to be between 0 and 63, and a user-specified ring number was checked to ensure that it was greater than or equal to the current ring. Unfortunately, that check missed the possiblity that setting a ring number to 64 would be greater than the current ring number, but when inserted into the bit field, would be truncated mod 64, and therefore be set to 0, the most privileged ring in the system.
Incorrect check positioning
Incorrect check positioning occurs when the programmer validates input parameters, but does the validation checks in the wrong order, or after the parameter has been used in some fashion. A simple example from a file system would be to check whether a file exists or not, before checking whether the user has access to search the containing directory. Returning a file does not exist error would reveal information that the user was not supposed to see.
Identification/Authentication Inadequate
This flaw occurs when the system does not completely check whether the caller has sufficient permissions to do the requested operation, or is the entity that it claims to be. We separate this class from the other validation errors because usually this requires access to meta-data or protocol-level information. Landwehr, et. al. [23, case U4] cites a case where â sen dmalâ on UN allowed the user to specify any file as the configuration file, even if the user did not have access to the file.
Abstraction Error
Object Reuse
Exposed Internal Representation
Asynchronous Flaws
Concurrency (including TOCTOU)
Aliasing
Subcomponent misuse/failure
Resource Leak
Responsibility Misunderstanding
Functionality Error
Error handling failure
Other security flaw