Rule 49. Miscellaneous (MSC)

- MSC50-CPP. Do not use std::rand() for generating pseudorandom numbers
- MSC51-CPP. Ensure your random number generator is properly seeded
- MSC52-CPP. Value-returning functions must return a value from all exit paths
- MSC53-CPP. Do not return from a function declared [[noreturn]]
- MSC54-CPP. A signal handler must be a plain old function

The following rules from the SEI CERT C Coding Standard also apply in C++:

- MSC30-C. Do not use the rand() function for generating pseudorandom numbers
- MSC32-C. Properly seed pseudorandom number generators
- MSC33-C. Do not pass invalid data to the asctime() function
- MSC37-C. Ensure that control never reaches the end of a non-void function
- MSC38-C. Do not treat a predefined identifier as an object if it might only be implemented as a macro
- MSC39-C. Do not call va_arg() on a va_list that has an indeterminate value
- MSC40-C. Do not violate constraints
- MSC41-C. Never hard code sensitive information
- FLP30-C. Do not use floating-point variables as loop counters
- FLP32-C. Prevent or detect domain and range errors in math functions
- FLP34-C. Ensure that floating-point conversions are within range of the new type
- FLP36-C. Preserve precision when converting integral values to floating-point type
- FLP37-C. Do not use object representations to compare floating-point values
- ENV30-C. Do not modify the object referenced by the return value of certain functions
- ENV31-C. Do not rely on an environment pointer following an operation that may invalidate it
- ENV32-C. All exit handlers must return normally
- ENV33-C. Do not call system()
- ENV34-C. Do not store pointers returned by certain functions
- SIG31-C. Do not access shared objects in signal handlers
- SIG34-C. Do not call signal() from within interruptible signal handlers
- SIG35-C. Do not return from a computational exception signal handler
- PRE30-C. Do not create a universal character name through concatenation
- PRE31-C. Avoid side effects in arguments to unsafe macros
- PRE32-C. Do not use preprocessor directives in invocations of function-like macros

Risk Assessment Summary

<table>
<thead>
<tr>
<th>Rule</th>
<th>Severity</th>
<th>Likelihood</th>
<th>Remediation Cost</th>
<th>Priority</th>
<th>Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSC50-CPP</td>
<td>Medium</td>
<td>Unlikely</td>
<td>Low</td>
<td>P6</td>
<td>L2</td>
</tr>
<tr>
<td>MSC51-CPP</td>
<td>Medium</td>
<td>Likely</td>
<td>Low</td>
<td>P18</td>
<td>L1</td>
</tr>
<tr>
<td>MSC52-CPP</td>
<td>Medium</td>
<td>Probable</td>
<td>Medium</td>
<td>P8</td>
<td>L2</td>
</tr>
<tr>
<td>MSC53-CPP</td>
<td>Medium</td>
<td>Unlikely</td>
<td>Low</td>
<td>P2</td>
<td>L3</td>
</tr>
<tr>
<td>MSC54-CPP</td>
<td>High</td>
<td>Probable</td>
<td>High</td>
<td>P6</td>
<td>L2</td>
</tr>
</tbody>
</table>