MSC00-PL. Detect and remove dead code

Code that is never executed is known as *dead code*. Typically, the presence of dead code indicates that a logic error has occurred as a result of changes to a program or the program's environment. To improve readability and ensure that logic errors are resolved, dead code should be identified, understood, and eliminated.

Noncompliant Code Example

This noncompliant code example contains code that cannot possibly execute.

```perl
sub fix_name {
    my $name = shift;
    if ($name eq "") {
        return $name;
    }
    $name =~ s/^([a-z])//g;
    $name =~ s/([a-z])//g;
    if (length($name) == 0) {
        die "Invalid name";  # cannot happen
    }
    return $name;
}
```

Compliant Solution

This compliant solution makes the dead code reachable.

```perl
sub fix_name {
    my $name = shift;
    $name =~ s/^([a-z])//g;
    $name =~ s/([a-z])//g;
    if (length($name) == 0) {
        die "Invalid name";  # cannot happen
    }
    return $name;
}
```

Risk Assessment

The presence of dead code may indicate logic errors that can lead to unintended program behavior. As a result, resolving dead code can be an in-depth process requiring significant analysis.

<table>
<thead>
<tr>
<th>Recommendation</th>
<th>Severity</th>
<th>Likelihood</th>
<th>Remediation Cost</th>
<th>Priority</th>
<th>Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSC00-PL</td>
<td>low</td>
<td>unlikely</td>
<td>high</td>
<td>P1</td>
<td>L3</td>
</tr>
</tbody>
</table>

Automated Detection

<table>
<thead>
<tr>
<th>Tool</th>
<th>Diagnostic</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perl::Critic</td>
<td>Subroutines::ProhibitUnusedPrivateSubroutines</td>
</tr>
<tr>
<td>Perl::Critic</td>
<td>ControlStructures::ProhibitUnreachableCode</td>
</tr>
</tbody>
</table>

Related Guidelines

<table>
<thead>
<tr>
<th>SEI CERT C Coding Standard</th>
<th>MSC00-PL. Detect and remove dead code</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEI CERT C++ Coding Standard</td>
<td>VOID MSC07-CPP. Detect and remove dead code</td>
</tr>
</tbody>
</table>
### Bibliography

<table>
<thead>
<tr>
<th>[CPAN]</th>
<th>Elliot Shank, Perl-Critic-1.116 Subroutines::ProhibitUnusedPrivateSubroutines, Variables::ProhibitUnreachableCode</th>
</tr>
</thead>
</table>

---