Содержание

От издательства

Отзывы и пожелания

Мы всегда рады отзывам наших читателей. Расскажите нам, что вы думаете об этой книге – что понравилось или, может быть, не понравилось. Отзывы важны для нас, чтобы выпускать книги, которые будут для вас максимально полезны.

Вы можете написать отзыв на нашем сайте www.dmkpress.com, зайдя на страницу книги и оставив комментарий в разделе «Отзывы и рецензии». Также можно послать письмо главному редактору по адресу dmkpress@gmail.com; при этом укажите название книги в теме письма.

Если вы являетесь экспертом в какой-либо области и заинтересованы в написании новой книги, заполните форму на нашем сайте по адресу http://dmkpress.com/authors/publish_book/ или напишите в издательство по адресу dmkpress@gmail.com.

Список опечаток

Хотя мы приняли все возможные меры для того, чтобы обеспечить высокое качество наших текстов, ошибки все равно случаются. Если вы найдете ошибку в одной из наших книг, мы будем очень благодарны, если вы сообщите о ней главному редактору по адресу dmkpress@gmail.com. Сделав это, вы избавите других читателей от недопонимания и поможете нам улучшить последующие издания этой книги.

Нарушение авторских прав

Пиратство в интернете по-прежнему остается насущной проблемой. Издательство «ДМК Пресс» очень серьезно относится к вопросам защиты авторских прав и лицензирования. Если вы столкнетесь в интернете с незаконной публикацией какой-либо из наших книг, пожалуйста, пришлите нам ссылку на интернет-ресурс, чтобы мы могли применить санкции.

Ссылку на подозрительные материалы можно прислать по адресу электронной почты dmkpress@gmail.com.

Мы высоко ценим любую помощь по защите наших авторов, благодаря которой мы можем предоставлять вам качественные материалы.

Глава 1

Введение

В этом стандарте приведены правила безопасного кодирования на языке программирования C++. Их цель – помочь в разработке безопасных и надежных систем, в частности путем устранения неопределенного поведения, которое может привести к появлению уязвимостей, допускающих эксплуатацию. Следование определенным в этом стандарте правилам кодирования – необходимое (но недостаточное) условие безопасности и надежности систем, написанных на языке C++. Дополнительно необходим, например, безопасный дизайн. К особо ответственным системам обычно предъявляются более строгие требования, чем описаны в этом стандарте, например чтобы вся память выделялась только статически. Однако применение этого стандарта позволит создавать высококачественные системы – надежные, безотказные и устойчивые к атакам.

Каждое правило содержит *заголовок*, *описание*, *примеры не соответствующего наставлению кода* и *соответствующее наставлению решение*, а также другую информацию, описанную в разделе «Как устроен этот стандарт кодирования». Заголовок представляет собой краткое, но иногда неточное описание правила. В описании приводятся нормативные требования, составляющие правило. В не соответствующих наставлению примерах приведен код, нарушающий правило. В соответствующем наставлению решении показан эквивалентный код, не нарушающий ни это, ни другие правила.

Хорошо документированный и подлежащий обязательному исполнению стандарт – важный элемент кодирования на языке C++. Стандарты кодирования побуждают программистов следовать единому набору правил, определенных требованиями проекта и организации, а не личными пристрастиями программиста. Будучи сформулированы, эти стандарты могут использоваться в качестве метрик для оценки качества исходного кода (с помощью ручных или автоматических процессов).

На вики-сайте стандарта кодирования SEI CERT C++ выкладываются обновления стандарта после выхода очередной версии. Если вы хотите принять участие в создании этих правил, заведите учетную запись на вики-сайте и запросите права соавтора, отправив запрос по адресу info@sei.cmu.edu.

В информационный бюллетень по безопасному кодированию (Secure Coding eNewsletter) включены новости инициативного проекта CERT по безопасному кодированию, а также краткая информация о последних обновлениях стандартных правил. Если вы хотите получать обновления напрямую, подпишитесь на этот бюллетень на нашем сайте или отправьте запрос по адресу info@sei.cmu.edu.

1.1. Область применимости

Стандарт кодирования CERT C++ разработан специально для версий языка C++, описанных в следующих документах:

- ISO/IEC 14882-2014, Programming Languages–C++, Fourth Edition, 2014 [ISO/IEC 14882-2014]

Хотя рекомендации в рамках этого стандарта были подготовлены для версии C++14, они применимы и к более ранним версиям, включая C++11. Там, где это необходимо, отмечаются различия между версиями стандарта C++, влияющие на применение рекомендаций.

В большинство наставлений включен пример не соответствующего наставлению кода, отвечающего стандарту C++14, так чтобы проблема, рассматриваемая в наставлении, попадала в область применимости данного стандарта. В большинство реализаций также включено соответствующее наставлению решение, работающее на нескольких платформах. Расширениям языка и библиотекам, опубликованным в виде технических отчетов или технических спецификаций ISO/IEC, часто отдается предпочтение. Изредка для иллюстрации описываются интересные поведения, зависящие от реализации.

1.1.1. Обоснование

Чтобы стандарт кодирования на языке C++ сохранял ценность как можно дольше, он должен быть ориентирован на стандарт C++14 и последующие технические отчеты.

Стандарт C++ документирует сложившуюся практику там, где это возможно. То есть большинство средств должны быть протестированы в реализации до включения в стандарт. У стандарта кодирования CERT C++ есть и дополнительная цель: установить набор наилучших практик, что иногда требует внедрения новых практик, возможно, малоизвестных или используемых в тех случаях, когда существующие практики не годятся. Иначе говоря, стандарт кодирования CERT C++ пытается играть роль застрельщика изменений, а не просто документировать их.

Некоторые поставщики предлагают расширения C++, тогда как другие реализовали только часть стандарта C++, прежде чем прекратить разработку. Поэтому невозможно обсуждать только C++98, C++03 или C++11. Уравне-

ние, описывающее поддержку со стороны поставщиков, слишком сложно, чтобы провести черту и сказать, что некоторый компилятор поддерживает в точности определенный стандарт. Какую бы точку разграничения ни выбрать, найдутся поставщики, которые находятся по разные стороны от нее для разных частей языка. Чтобы узнать о поддержке всех возможностей, пришлось бы протестировать комбинации каждого компилятора с каждым языковым средством. Поэтому мы выбрали в качестве точки разграничения самый поздний момент, желая, чтобы правила и рекомендации, определенные в стандарте, оставались актуальными как можно дольше. Как следствие поддерживаемые вариации и переносимость исходного кода оказываются лучше, если программист использует только средства, определенные в стандарте C++14. Это один из многих компромиссов между безопасностью и переносимостью, свойственных программированию на языке C++.

Ценность опережающей информации сначала возрастает со временем, а потом начинает снижаться. Ценность информации, полученной в результате анализа прошлого, начинает снижаться немедленно.

По всем этим причинам приоритетной целью настоящего стандарта является поддержка разработки нового кода с использованием C++14. Чуть менее важная цель – поддержка исправлений старого кода, написанного на C++11.

В этом стандарте уделено некоторое внимание поддержке старых компиляторов, если это существенно и не противоречит другим, более приоритетным целям. Наша задача – собрать не все отклонения от стандарта, а лишь немного, но важные.

1.1.2. Какие вопросы не рассматриваются

Ряд вопросов в этом стандарте безопасного кодирования не рассматриваются.

1.1.2.1. Стиль кодирования

Стиль кодирования – вещь субъективная, и практика показала, что прийти к консенсусу по этому поводу невозможно. Поэтому стандарт безопасного кодирования CERT C++ не требует обязательного соблюдения какого-то определенного стиля, а лишь рекомендует организациям принять наставления по стилю кодирования и неукоснительно им следовать. Проще всего добиться применения единого стиля – воспользоваться инструментом форматирования кода. Во многих интерактивных средах разработки (IDE) такая возможность имеется.

1.1.2.2. Спорные правила

Вообще говоря, в стандарты кодирования CERT обычно не включаются спорные правила, для которых отсутствует единодушная поддержка.

1.2. Аудитория

Стандарт безопасного кодирования CERT C++ рассчитан в первую очередь на разработчиков программ на C++, но может также использоваться заказчиками программного обеспечения для определения требований к программному обеспечению. Стандарт представляет особый интерес для разработчиков, которые стремятся создавать высококачественные системы, надежные и устойчивые к атакам.

Хотя стандарт и не предназначен программистам на C, он может представлять для них определенную ценность, потому что подавляющее большинство проблем, свойственных программам на языке C, являются также проблемами и в C++, хотя решения во многих случаях различны.

1.3. Практическое применение

Правила, изложенные в этом стандарте, нацелены на повышение безопасности программ путем повышения осведомленности разработчиков, а также усовершенствования используемых ими практик и инструментов.

Этот стандарт можно использовать для разработки адаптированных стандартов кодирования для проектов и организаций, обеспечивающих единообразный подход к безопасности разработки программного обеспечения. Он может быть дополнен правилами, относящимися к конкретной организации. Однако заявлять о согласованности с настоящим стандартом допустимо только при соблюдении содержащихся в нем правил.

Этот стандарт можно также использовать для тестирования на согласованность для выбора инструмента и выполнения проверки. Коль скоро стандарт кодирования определен, можно разработать или модифицировать инструменты и процессы, согласующиеся с ним.

Также этот стандарт можно использовать для разработки учебных курсов и обучения профессионалов правильному применению стандартов кодирования. Институт программной инженерии (Software Engineering Institute – SEI) предлагает несколько курсов по безопасному кодированию с выдачей сертификатов – как очных, так и онлайновых. Материалы настоящего стандарта, а также и дополнительные учебные и оценочные материалы могут быть использованы для:

1) выявления претендентов на работу с конкретными навыками программирования;
2) демонстрации наличия хорошо обученного персонала;
3) предоставления руководящих материалов учебным и образовательным учреждениям.

1.4. Как устроен этот стандарт кодирования

Настоящий стандарт кодирования состоит из 11 глав, которые содержат правила в основных областях, и четырех приложений. Приложение A содержит список литературы, приложение B – определения терминов, приложение C – связи между правилами стандарта и двух других наставлений на ту же тему: *MISRA C++ 2008: Guidelines for the Use of the C++ Language in Critical Systems* [MISRA 2008] и *MITRE's Common Weakness Enumeration (CWE)* [MITRE 2010]. Наконец, в приложении D приводятся оценки риска для каждого правила, приведенного в стандарте кодирования.

Большинство правил придерживаются единой структуры. У каждого правила имеется уникальный идентификатор, включенный в заголовок. В заголовке и начальных абзацах правило определяется, обычно далее следуют одна или несколько пар примеров не соответствующего наставлению кода и соответствующего наставлению решения. Для каждого правила включены также оценка риска, связанные наставления и литература (там, где это имеет смысл). Правило может также включать таблицу связанных с ним уязвимостей.

1.4.1. Идентификаторы

Каждому правилу и рекомендации (см. раздел «Правила и рекомендации) присваивается уникальный идентификатор. Идентификаторы состоят из трех частей:

- трехбуквенное мнемоническое обозначение раздела стандарта;
- двузначное числовое значение в диапазоне от 00 до 99;
- суффикс, описывающий язык или платформу:
  - «-C» – стандарт кодирования SEI CERT C;
  - «-CPP» – стандарт кодирования SEI CERT C++;
  - «-J» – стандарт кодирования SEI CERT Oracle для Java;
  - «-PL» – стандарт кодирования SEI CERT Perl.

Трехбуквенное мнемоническое обозначение используется, чтобы сгруппировать схожие практики кодирования и указать, какой категории принадлежит практика.

Числовое значение играет роль уникального идентификатора внутри категории. Числа от 00 до 49 зарезервированы для рекомендаций, а от 50 до 99 – для правил. (В стандарте кодирования SEI CERT C соглашения иные.) В наставлениях из этого стандарта ссылки на правила и рекомендации часто даются в виде идентификатора и заголовка.

Ниже приведено несколько примеров идентификаторов с пояснениями:

- INT50-CPP. Не приводите к значению перечисления, выходящему за пределы диапазона
  - Это идентификатор правила.
  - «INT» означает категорию Integer.
  - «50» – уникальный идентификатор.
  - «-CPP» означает язык C++.
- EXP00-J. Не игнорируйте значения, возвращаемые методами
  - Это идентификатор рекомендации.
  - «EXP» означает категорию Expressions.
  - «00» – уникальный идентификатор.
  - «-J» означает язык Java.
- FLP00-C. Уясните ограничения чисел с плавающей точкой
  - Это идентификатор рекомендации.
  - «FLP» означает категорию Floating Point.
  - «00» – уникальный идентификатор.
  - «-C» означает язык C.

1.4.2. Примеры не соответствующего наставлению кода и соответствующих наставлению решений

Примеры не соответствующего наставлению кода иллюстрируют код, нарушающий обсуждаемое наставление. Важно отметить, что это всего лишь примеры, а исключение из программы всех вхождений примера еще не означает, что анализируемый код теперь соответствует наставлению.

За примерами не соответствующего наставлению кода обычно следуют соответствующие наставлению решения, которые показывают, как пример не соответствующего кода можно переписать безопасно. Если явно не оговорено противное, то примеры не соответствующего наставлению кода должны содержать только нарушения обсуждаемого наставления. Соответствующие наставлению решения должны быть согласованы со всеми правилами безопасного кодирования, но иногда могут расходиться с рекомендациями.

1.4.3. Исключения

В любом правиле или рекомендации может быть определен небольшой набор исключений, подробно описывающих обстоятельства, при которых несоблюдение наставления не влечет отсутствие гарантий безопасности или надежности программы. Исключения приведены только для информации, следовать им необязательно.

1.4.4. Оценка риска

В каждом наставлении из стандарта кодирования CERT C++ имеется раздел с оценкой риска, цель которого – попытаться дать разработчикам ПО представление о потенциальных последствиях пренебрежения данным правилом или рекомендацией (а также некоторое представление об ожидаемых затратах на исправление). Команда разработчиков может использовать эту информацию, чтобы упорядочить исправления нарушенных правил по приоритетности. Метрика предназначена главным образом для проектов по исправлению старого кода. Вообще говоря, предполагается, что вновь разрабатываемый код будет соответствовать стандарту в целом и всем применимым рекомендациям.

Каждому правилу и рекомендации присваивается *приоритет*. Приоритеты назначаются на основе анализа видов, последствий и критичности отказов (АВПКО) (*англ.* FMECA) [IEC 60812 2006]. Каждому правилу сопоставляются три значения от 1 до 3: тяжесть последствий, вероятность и стоимость исправления.

Тяжесть последствий – насколько серьезны последствия игнорирования правила?

Значение	Интерпретация	Примеры уязвимостей
1	Низкая	Атака типа «отказ в обслуживании», аварийное завершение
2	Средняя	Нарушение целостности данных, непреднамеренное раскрытие информации
3	Высокая	Выполнение произвольного кода

Вероятность – насколько вероятно, что дефект, возникший из-за игнорирования этого правила, может привести к допускающей эксплуатацию уязвимости?

Значение	Интерпретация
1	Маловероятно
2	Вероятно
3	Весьма вероятно

Стоимость исправления – затраты на приведение в соответствие с правилом.

Значение	Интерпретация	Обнаружение	Исправление
1	Высокая	Ручное	Ручное
2	Средняя	Автоматическое	Ручное
3	Низкая	Автоматическое	Автоматическое

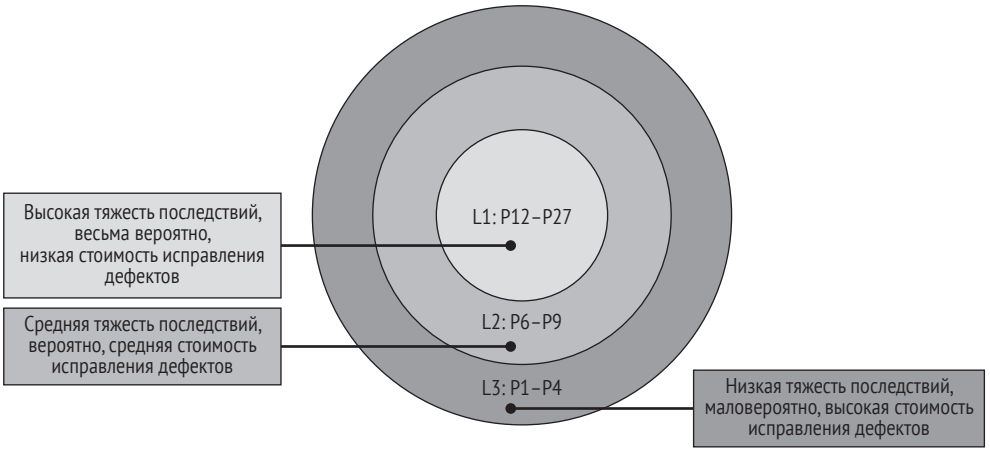
Для каждого правила все три значения перемножаются. Произведение является метрикой, которую можно использовать для упорядочения при-

менения правил по приоритету. Произведения изменяются в диапазоне от 1 до 27, и всего имеется 10 возможных значений: 1, 2, 3, 4, 6, 8, 9, 12, 18, 27. Наставления с приоритетом от 1 до 4 называются правилами **уровня 3**, от 6 до 9 – **уровня 2**, а от 12 до 27 – **уровня 1**. Ниже приведены возможные интерпретации приоритетов и уровней.

Приоритеты и уровни – насколько серьезны последствия игнорирования правила?

Уровень	Приоритеты	Возможная интерпретация
L1	**12, 18, 27**	Высокая тяжесть последствий, весьма вероятно, стоимость исправления низкая
L2	**6, 8, 9**	Средняя тяжесть последствий, вероятно, стоимость исправления средняя
L3	**1, 2, 3, 4**	Низкая тяжесть последствий, маловероятно, стоимость исправления высокая

В конкретных проектах исправление может начаться с реализации всех правил определенного уровня, после чего можно переходить к низкоприоритетным правилам, как показано на рисунке ниже.



Рекомендации не обязательны и приведены только для информации

1.4.5. Автоматическое обнаружение

На вики-сайте (https://clck.ru/3MaLxS), а также в правилах и в рекомендациях часто имеются разделы, в которых описывается автоматическое обнаружение. Там приводится дополнительная информация об анализаторах, которые умеют автоматически диагностировать нарушения наставлений по кодированию. Автоматические виды анализа для языка C++ по большей части не являются ни надежными, ни полными, поэтому включение инструмента в этот раздел

обычно означает, что инструмент может диагностировать некоторые нарушения этого конкретного правила. Пакет Secure Coding Validation Suite (https://clck.ru/3MaM2H) можно использовать для проверки способности анализаторов диагностировать нарушения правил стандарта ISO/IEC TS 17961:2012, который тесно связан с правилами из стандарта кодирования SEI CERT C.

Информация, приведенная в разделах, посвященных автоматическому обнаружению, может быть:

- предоставлена поставщиками;
- получена CERT путем неформальной оценки анализатора;
- получена CERT путем анализа документации поставщика.

Там, где возможно, мы стараемся ссылаться на точную версию инструмента, для которой получены результаты. Поскольку эти инструменты постоянно развиваются, информация довольно быстро устаревает.

1.4.6. Связанные уязвимости

В разделах по оценке рисков на вики-сайте приводятся также ссылки на поиск связанных уязвимостей на сайте CERT. Там, где возможно, примечания об уязвимостях CERT помечены ключевым словом, соответствующим уникальному идентификатору наставления по кодированию. Этот поиск дает актуальный список реальных уязвимостей, причиной которых хотя бы отчасти является нарушение данного наставления. Уязвимости помечаются таким образом, только если группа анализа уязвимостей (https://clck.ru/3MaM9y) в CERT/CC имеет возможность ознакомиться с исходным кодом и точно определить причину уязвимости. Поскольку многие уязвимости ссылаются на программные системы с закрытым исходным кодом, провести такой дополнительный анализ не всегда возможно. Поэтому поле связанных уязвимостей заполнено редко.

Разделы, посвященные связанным уязвимостям, включаются только для некоторых правил, если информация считается релевантной и интересной.

1.4.7. Связанные наставления

Раздел связанных наставлений содержит ссылки на родственные стандарты, технические спецификации и собрания наставления, например: *Information Technology–Programming Languages, Their Environments and System Software Interfaces–C Secure Coding Rules* [ISO/IEC TS 17961:2012]; *Information Technology–Programming Languages–Guidance to Avoiding Vulnerabilities in Programming Languages through Language Selection and Use* [ISO/IEC TR 24772:2013]; *MISRA C++ 2008: Guidelines for the Use of the C++ Language in Critical Systems* [MISRA 2008] и идентификаторы CWE в *MITRE's Common Weakness Enumeration (CWE)* [MITRE 2010].

Вы можете построить уникальный URL-адрес для получения дополнительной информации, добавив интересующий вас идентификатор в конец фиксированной строки. Например, чтобы найти информацию о «CWE 192: Integer Coercion Error», добавьте 192.html к http://cwe.mitre.org/data/definitions/ и введите результирующий URL в адресную строку браузера: http://cwe.mitre.org/data/definitions/192.html.

Другие технические спецификации, технические отчеты и наставления доступны на коммерческой основе.

1.4.8. Литература

В большинстве наставлений имеется небольшой раздел литературы, в котором перечислены документы и разделы в них, содержащие информацию, относящуюся к данному наставлению.

1.5. Связь со стандартом кодирования CERT C

В разделе [intro.scope] стандарта C++, параграф 2 [ISO/IEC 14882-2014], читаем:

> C++ – универсальный язык программирования, основанный на языке программирования C, описанном в документе ISO/IEC 9899:1999 Programming languages – C (который далее называется стандартом C). В дополнение к средствам, предоставляемым C, язык C++ предоставляет дополнительные типы данных, шаблоны, исключения, пространства имен, перегрузку операторов, перегрузку имен функций, ссылки, операторы управления свободной памятью и дополнительные библиотечные средства.

Поскольку C++ основан на языке программирования C, между наставлениями в данном стандарте и в стандарте кодирования SEI CERT C имеется значительное перекрытие. Чтобы уменьшить объем повторений, в этом стандарте внимание уделено тем частям языка программирования C++, которые не полностью покрываются стандартом кодирования CERT C. Из-за повышенного внимания к типам в C++ некоторые правила C дополнены в стандарте безопасного кодирования CERT C++. Если явно не оговорено противное, содержимое стандарта кодирования CERT C равным образом применимо к коду, написанному на C++. То содержимое стандарта кодирования CERT C, которое применимо к стандарту кодирования CERT C++, описано в соответствующих главах стандарта C++.