

Тема 1.5 Безопасность микропроцессорных систем

Некоторые встроенные системы находят массовое применение, например, устройства RFID. Встроенные системы являются привлекательной целью для создателей вредоносного кода из-за своей распространённости и относительной незащищённости. Постепенно злоумышленники пытаются создать вредоносный код для встроенных систем (например, RFID-вирус, Cabir). Этот процесс пока затрудняется разнородностью встроенных устройств, отсутствием доминирующего ПО и ограниченной функциональностью некоторых видов устройств. С другой стороны, задача антивирусных компаний и исследователей компьютерной безопасности осложнена теми же обстоятельствами, а также маломощностью встроенных систем, зачастую не позволяющей пользоваться распространённым антивирусным ПО.

При передаче сигнала через любой канал связи возможно возникновение ошибок, которые могут приводить к искажению переносимой информации. Существует много методов для исправления подобных ошибок, но прежде чем исправлять, необходимо эти ошибки обнаружить. Для этого также существуют определенные методы, основанные на избыточности передаваемой информации, что позволяет не только выявлять наличие факта искажения информации, но и в ряде случаев устранять эти искажения. Наиболее известные из методов обнаружения ошибок передачи данных являются:

1. Посимвольный контроль четности, называемый также поперечным, подразумевает передачу с каждым байтом дополнительного бита, принимающего единичное значение по четному или нечетному количеству единичных битов в контролируемом байте. Посимвольный контроль четности прост как в программной, так и в аппаратной реализации, но его вряд ли можно назвать эффективным методом обнаружения ошибок, так как искажение более одного бита исходной последовательности резко снижает вероятность обнаружения ошибки передачи. Этот вид контроля обычно реализуется аппаратно в устройствах связи.

2. Поблочный контроль четности, называемый продольным. Схема данного контроля подразумевает, что для источника и приемника информации заранее известно, какое число передаваемых символов будет рассматриваться ими как единый блок данных. В этой схеме контроля для каждой позиции разрядов в символах блока (поперек блока) рассчитываются свои биты четности, которые добавляются в виде обычного символа в конец блока. По сравнению с посимвольным контролем четности поблочный контроль четности обладает большими возможностями по обнаружению и даже корректировке ошибок передачи, но все равно ему не удается обнаруживать определенные типы ошибок.

3. Вычисление контрольных сумм. В отличие от предыдущих методов для метода контрольных сумм нет четкого определения алгоритма. Каждый разработчик трактует понятие контрольной суммы по-своему. В простейшем виде контрольная сумма — это арифметическая сумма двоичных значений контролируемого блока символов. Но этот метод обладает практически теми же недостатками, что и предыдущие, самый главный из которых — нечувствительность контрольной суммы к четному числу ошибок в одной колонке и самому порядку следования символов в блоке.

4. Контроль циклически избыточным кодом — CRC (Cyclical Redundancy Check). Это гораздо более мощный и широко используемый метод обнаружения ошибок передачи информации. Он обеспечивает обнаружение ошибок с высокой вероятностью. Кроме того, этот метод обладает рядом других полезных моментов, которые могут найти свое воплощение в практических задачах.

Пять шагов к обеспечению безопасности встраиваемых систем

В современных условиях растущей связности вычислительных устройств проблема информационной безопасности выходит на один уровень с проблемой сложности ПО. Статья описывает пять шагов по обеспечению информационной безопасности встраиваемых систем с учётом всех стадий их жизненного цикла – от проектирования до разработки, тестирования, развертывания и обслуживания.

Первый «звоночек»

2010 год ознаменовался первым «червём», способным внедриться в промышленную инфраструктуру и позволять хакеру переключать управление критическими системами, — Stuxnet. Он был разработан для атаки на центрифуги, используемые в ядерной программе Ирана, и отбросил работу по программе предположительно на несколько лет назад. Stuxnet обладал

неспособностью классических ПЛК противостоять сетевым вторжениям, так как при их проектировании предполагалось, что изолированная сеть системы управления безопасна по умолчанию. Stuxnet развеял этот миф, продемонстрировав способность вторгаться в изолированные сети систем управления через подключенные к ним компьютеры. Многие современные сетевые угрозы являются ещё более изощ-

ными, чем Stuxnet, и требуют большего объёма требований увеличивает трудность разработки, а значит, и её стоимость. К тому же огромное число потенциально уязвимых встраиваемых систем, основанных на устаревших стратегиях безопасности, уже давно развёрнуты и эксплуатируются в современной тесно связанной сетевой среде. Потребности в поддержке облачных вычислений увеличивают степень связности устройств, тем са-

мым образом увеличивая сложность обеспечения безопасности. В то же время, с другой стороны, современные системы управления промышленными объектами (SCADA) и системы управления производством (MES) становятся всё более сложными и требуют более тщательного контроля безопасности.

В то же время, с другой стороны, современные системы управления промышленными объектами (SCADA) и системы управления производством (MES) становятся всё более сложными и требуют более тщательного контроля безопасности.

В то же время, с другой стороны, современные системы управления промышленными объектами (SCADA) и системы управления производством (MES) становятся всё более сложными и требуют более тщательного контроля безопасности.

В то же время, с другой стороны, современные системы управления промышленными объектами (SCADA) и системы управления производством (MES) становятся всё более сложными и требуют более тщательного контроля безопасности.

В то же время, с другой стороны, современные системы управления промышленными объектами (SCADA) и системы управления производством (MES) становятся всё более сложными и требуют более тщательного контроля безопасности.

В то же время, с другой стороны, современные системы управления промышленными объектами (SCADA) и системы управления производством (MES) становятся всё более сложными и требуют более тщательного контроля безопасности.

В то же время, с другой стороны, современные системы управления промышленными объектами (SCADA) и системы управления производством (MES) становятся всё более сложными и требуют более тщательного контроля безопасности.

В то же время, с другой стороны, современные системы управления промышленными объектами (SCADA) и системы управления производством (MES) становятся всё более сложными и требуют более тщательного контроля безопасности.

В то же время, с другой стороны, современные системы управления промышленными объектами (SCADA) и системы управления производством (MES) становятся всё более сложными и требуют более тщательного контроля безопасности.

В то же время, с другой стороны, современные системы управления промышленными объектами (SCADA) и системы управления производством (MES) становятся всё более сложными и требуют более тщательного контроля безопасности.

В то же время, с другой стороны, современные системы управления промышленными объектами (SCADA) и системы управления производством (MES) становятся всё более сложными и требуют более тщательного контроля безопасности.

В поисках золотой

СЕРЕДИНЫ

В то же время, с другой стороны, современные системы управления промышленными объектами (SCADA) и системы управления производством (MES) становятся всё более сложными и требуют более тщательного контроля безопасности.

В то же время, с другой стороны, современные системы управления промышленными объектами (SCADA) и системы управления производством (MES) становятся всё более сложными и требуют более тщательного контроля безопасности.

В то же время, с другой стороны, современные системы управления промышленными объектами (SCADA) и системы управления производством (MES) становятся всё более сложными и требуют более тщательного контроля безопасности.

[illegible]



Рис. 1. Пять шагов к обеспечению безопасности встраиваемых систем

изготовителя устройства, операторов (если они предусмотрены) и конечных пользователей (исключая соответствующую среду эксплуатации) и описывать в виде пар так называемых векторов атаки (то есть способов осуществления атаки) и эксплуатируемых ими уязвимостей (то есть незащищенностей или слабостей в программном или аппаратном обеспечении, позволяющих атаке достигнуть цели). В качестве примера вектора атаки можно привести кабельное Ethernet-соединение и предоставляемые через него типовые сервисы наподобие HTTP, FTP, SSH или отладочных агентов. Примером уязвимости может служить слишком простой или установленный по умолчанию пароль, или ошибка кодирования (например, отсутствие проверки переполнения стека), или даже концептуальная ошибка проектирования (например, некорректная последовательность начальной загрузки). Самая трудная часть здесь — это предсказать и предотвратить все возможные векторы атаки и уязвимости заранее.

Чтобы оценка принесла плоды, необходимо проанализировать устройство в очень широком спектре потенциальных угроз. Множество реальных современных угроз возникло из предположения, что устройство просто не может использоваться тем или иным способом. Становит, в частности, атакуют ПЛК, расположенные в общей сети с инфицированными настольными и портативными компьютерами. Однако даже учитывая, что такая сеть изначально не подключена к Интернету, вполне вероятно, что к ней иногда будут подключаться диагностические или инструментальные компьютеры. При оценке угроз важно рассматривать не только сами устройства, но и внешнюю среду, в частности, операторов и конечных пользователей.

В процессе оценки угроз для встраиваемого устройства необходимо сделать как минимум следующее.

- Провести полный анализ жизненного цикла продукта, при этом необходимо учесть как разработчиков, так и изготовителей, дистрибьюторов, поставщиков и конечных пользователей, чтобы получить полную картину их влияния на безопасность. Необходимо также оценить приоритет кибербезопасности и защиты информации для данного устройства.
- Определить и описать все возможные входные точки атаки, при этом не следует закрываться на сетевом доступе, так как существуют и другие варианты, например, физический доступ через USB или последовательные порты. Когда входные точки определены, нужно оценить уязвимости для каждой из них. Защищено ли устройство от атаки через TCP-порт 80? Используется ли межсетевая экран? Если нет, какие порты TCP/UDP открыты? Аналогично для физического доступа: поддерживает ли устройство загрузку с внешнего USB-носителя? Необходимо также анализ возможных комбинаций входных точек.
- Построить матрицу рисков. Поскольку возможных вариантов атаки может быть множество, требуется оценка вероятности атаки по каждому каналу и возможного уровня от такой атаки.
- Разработать стратегию сокращения рисков, исходя из заданных приоритетов. Например, выигрышной стратегией может быть разбиение системы на разделы с разными уровнями безопасности. В ряде случаев это может вести к усложнению архитектуры, однако эта стратегия окупается за счёт уменьшения объёма тестирования и сокращения стоимости обслуживания на более поздних стадиях жизненного цикла.

- Создать техническую спецификацию, включающую в себя требования к безопасности, полученные на основе предыдущих действий. Это равноправная часть процесса разработки, но её следует рассматривать как высокоприоритетную. План работ по проектированию, разработке, тестированию и сопровождению средств безопасности должен стать частью общего рабочего плана.

Шаг 2: проработка безопасной архитектуры

Ответом на усиление значения безопасности связанных устройств стало развитие ряда технологий и методологий разработки. Одной из важных парадигм разработки безопасных устройств является использование готовых коммерческих (commercial off-the-shelf — COTS) системных компонентов, которое позволяет обеспечивать безопасность, одновременно контролируя затраты. В качестве примеров можно привести как сертифицируемые ОС и связующее ПО, так и технологии виртуализации, разбиения на разделы и виртуальные среды исполнения, позволяющие увеличить уровень абстракции и разделения компонентов.

В последнее время набирает особую популярность виртуализация, так как она позволяет выполнять несколько ОС на общей разделённой аппаратной платформе. Это даёт проектировщикам дополнительную гибкость, а также позволяет более эффективно использовать возможности оборудования по сравнению с дублированием на базе одной ОС. Виртуализация также предоставляет хорошую почву для распределения функциональности устройства по нескольким разделённым виртуальным средам, что позволяет разворачивать на одной физической платформе компоненты с повышенными требованиями к функциональной или информационной безопасности и некритические приложе-



Рис. 2. Пример разбиения на разделы с использованием гипервизора

ника. Например, в подобной виртуализированной среде можно разместить ОС реального времени (OS RV) и ОС обще-

го назначения. Не стоит забывать, что встраиваемые системы имеют жесткие требования к функциональности (рис. 2). В то же время, если речь идет о универсальных ОС общего назначения, то можно использовать ОС RV, — здесь убудет: конкретная функциональность и ее качество проверяется заданную ил.

Классификация встраиваемых компонентов по назначению, тогда и сразу, которую актуально характеризовать по ее функциональным признакам:

конкретных аппаратных средств. Переход на коммерческие средства поддержки оборудования — естественный шаг на пути внедрения повторно используемых компонентов; однако здесь очень важно правильно выбрать поставщиков. Коммерческие средства поддержки оборудования (например BSP) оптимизированы для конкретных целевых устройств, обеспечены технической поддержкой, а в ряде случаев ещё и снабжены сертификационной документацией, позволяющей использовать их в составе систем, предъявляющих повышенные требования к функциональной и информационной безопасности.

• **Встраиваемые средства виртуализации.** Встраиваемый гипервизор способен обеспечить разбиение системы на разделы, параллельное выполнение нескольких ОС и поддержку многозадачных приложений — всё это позволяет повысить степень интеграции. В сочетании с коммерческими средствами поддержки оборудования встраиваемые средства виртуализации, обеспеченные коммерческими пакетами сертификационной доку-

му и различных проектах. На более высоком уровне это означает использование готовых модулей, гарантированно удовлетворяющих требованиям качества и безопасности, а это проверенный путь к сокращению себестоимости, сроков разработки и главное — к управлению рисками. Многие коммерческие программные продукты проходят исчерпывающее тестирование, валидацию и сертификацию и могут быть идеальными кандидатами на роль повторно используемых компонентов для устройств нового поколения. Разработчики могут сразу начать пользоваться необходимой функциональностью и готовой сертификационной документацией.

Шаг 3: выбор безопасной программной платформы

Выбор программной платформы для встраиваемой системы — решение сложное. Решения системы по безопасности имеют решающее значение для сертификационной документации, связанной с уровнем



Рис. 3. Стек программных компонентов среды разработки и исполнения

безопасности и сократить затраты; есть также и другие преимущества использования коммерческих компонентов

Важно отметить, что встраиваемые системы имеют жесткие требования к функциональности (рис. 2). В то же время, если речь идет о универсальных ОС общего назначения, то можно использовать ОС RV, — здесь убудет: конкретная функциональность и ее качество проверяется заданную ил.

Классификация встраиваемых компонентов по назначению, тогда и сразу, которую актуально характеризовать по ее функциональным признакам:

конкретных аппаратных средств. Переход на коммерческие средства поддержки оборудования — естественный шаг на пути внедрения повторно используемых компонентов; однако здесь очень важно правильно выбрать поставщиков. Коммерческие средства поддержки оборудования (например BSP) оптимизированы для конкретных целевых устройств, обеспечены технической поддержкой, а в ряде случаев ещё и снабжены сертификационной документацией, позволяющей использовать их в составе систем, предъявляющих повышенные требования к функциональной и информационной безопасности.

• **Встраиваемые средства виртуализации.** Встраиваемый гипервизор способен обеспечить разбиение системы на разделы, параллельное выполнение нескольких ОС и поддержку многозадачных приложений — всё это позволяет повысить степень интеграции. В сочетании с коммерческими средствами поддержки оборудования встраиваемые средства виртуализации, обеспеченные коммерческими пакетами сертификационной доку-

ментации, помогают быстрее разрабатывать устройства, требующие реализации нескольких уровней безопасности одновременно.

• **ОС реального времени.** Многие встраиваемые системы предъявляют жесткие требования к ресурсоемкости и временным характеристикам ПО, а также требуют сертификации по различным стандартам функциональной/информационной безопасности. Хорошим фундаментом для таких приложений будет ОС реального времени (OS RV). (Объем кода ОС RV обычно значительно меньше, чем объем кода ОС общего назначения, поэтому сертификация ОС RV обходится существенно дешевле. — Прим. пер.). При выборе ОС RV следует обратить особое внимание на следующее:

- Безопасность конфигурации по умолчанию. Является ли безопасной конфигурация ОС RV по умолчанию (например, отключены ли второстепенные сервисы, закрыты ли сетевые порты и т.п.)?
- Безопасность коммуникационных средств. Поддерживает ли ОС RV сервисы, реализующие стандарты защищенных коммуникаций? Содержит ли ОС RV криптографические средства? Позволяет ли сетевой стек ОС RV использовать безопасные сокет, виртуальные частные сети (VPN), IPsec и т.п.?
- Сертификация. Есть ли для данной ОС RV сертификационные пакеты по стандартам безопасности, применимым к данному приложению? Проходил ли сетевой стек ОС RV тестирование и валидацию на предмет безопасности?
- Приоритеты производителя в вопросах безопасности. Имеют ли высокий приоритет вопросы безопасности у производителя ОС RV? Есть ли у него выделенная рабочая группа по вопросам безопасности?

• **Встраиваемые ОС общего назначения.** Стандартные дистрибутивы ОС общего назначения (например Linux или Microsoft Windows) не входят в класс ПО повышенной безопасности. Лучше всего использовать их в сочетании со встраиваемым гипервизором, который обеспечит их выполнение в выделенном защищенном разделе. Такой подход обеспечивает их изоляцию от критических частей си-

• Средства симуляции оборудования. Симуляция оборудования на уровне процессора, отдельных плат и системы в целом помогает ускорить процесс интеграции аппаратного и программного обеспечения, так как гарантирует доступность оборудования в нужном месте, в нужный срок и в достаточном количестве. (В процессе разработки встраиваемой системы это часто становится серьёзной проблемой, так как пока прототип не будет разработан, стабилизирован и произведён в достаточном количестве, ПО отладить будет физически не на чём. — *Прим. пер.*) К тому же средства симуляции позволяют использовать более эффективные методики отладки и тестирования, которые на реальном оборудовании реализовать физически невозможно (например, выполнение кода от момента ошибки в обратном порядке, чтобы

Поскольку чёрные списки гораздо объёмнее белых и гораздо чаще меняются, встраиваемым системам обычно недостаёт ресурсов для постоянной поддержки актуальности чёрных списков, так как для этого необходимо наличие постоянного транзита, сдвиги сетевой синхронизации

В контексте безопасности разработчикам внутренних систем следует учитывать как минимум следующие аспекты управления жизненным циклом:

- Безопасность должна быть встроена во все стадии жизненного цикла. Безопасный дизайн должен присутствовать в системе изначально и быть основой

[illegible]

• Безопасность должна иметь приоритет в процессе разработки и тестирования. Проблемы с безопасностью ПО фактически являются следствием ошибок в требованиях (будь то их формирование или реализация), и чем раньше в цикле разработки они будут обнаружены, тем дешевле обойдётся их исправление, а значит, и ликвидация уязвимостей. Тестирование безопасности должно включать в себя определение границ системы и методов эксплуатации слабых мест в их защите. Очень хороши здесь могут быть техники наподобие fuzz-тестирования (тестирование на основе заведомо некорректных или случайных входных данных, которое позволяет, в частности, выявлять ошибки в редко используемых участках кода. — *Прим. пер.*) или испытания на проникновение (симуляция вестиров атаки потенциальных злоумышленников). Автоматизация тестирования безопасности вкупе с использованием средств симуляции оборудования способна существенно повысить эффективность процесса разработки и обеспечить гораздо более

детальное исследование поведения продуктов, чем интуитивное (ad hoc) тестирование.

• Устранение дефектов, связанных с безопасностью, должно иметь высший приоритет. Безопасности следует уделить особое внимание не только в процессе разработки, но и в процессе сопровождения. Когда уязвимость обнаруживается профессиональным сообществом, она быстро становится достоянием общественности. Производителям необходимо уметь оперативно реагировать на подобные ситуации по мере их возникновения.

• Вопросы безопасности должны занимать специальную рабочую группу. В её задачи будут входить анализ уязвимостей, проработка возможных путей их устранения, внутреннее и внешнее коммуникации, планирование выпуска обновлений и контроль выполнения запланированных мероприятий. Подобные группы обычно кросс-функциональны и включают в себя представителей подразделений разработки аппаратуры и ПО, тестирования, технической поддержки,

управления требованиями и документирования.

ЗАКЛЮЧЕНИЕ

Описанные пять шагов позволяют организациям достичь фундаментального прогресса в области защиты связанных встраиваемых систем от угроз. Учёт вопросов безопасности в процессе разработки встраиваемых систем сегодня является ключевым требованием и предусматривает вовлечение всех уровней и подразделений организации в создание всеобъемлющей защитной структуры, особенно необходимой в современных связанных вычислительных средах. ●

ЛИТЕРАТУРА

1. National Information Assurance (IA) Glossary : CNSS Instruction No. 4009. — 26 April 2010. — Committee on National Security Systems.

Автор — вице-президент компании Wind River по решениям в области жизненного цикла

Перевод Николая Горбунова, сотрудника фирмы ПРОСОФТ

Телефон: (495) 234-0636

E-mail: info@prosoft.ru

[illegible]

Сложение двух чисел в CRC арифметике полностью аналогично обычному арифметическому действию за исключением отсутствия переносов из разряда в разряд. Это означает, что каждая пара битов определяет результат своего разряда вне зависимости от результатов других пар. Например:

$$\begin{array}{r} 10011011 \\ + 11001010 \\ \hline 01010001 \end{array} \quad \begin{array}{r} 10011011 \\ - 11001010 \\ \hline 01010001 \end{array}.$$

Т.е. и при сложении и при вычитании над каждым битом в отдельности выполняется операция, аналогичная операции *XOR*, поэтому в CRC арифметике две арифметические операции (сложение и вычитание) заменяются операцией *XOR*.

Умножение, как и в обычной арифметике, считается суммой значений первого множителя, сдвинутых в соответствии со значением второго множителя. Причем при суммировании так же используется CRC сложение. Например:

$$\begin{array}{r} 1101 \\ 1011 \\ \hline 1101 \\ 1101 \\ 0000 \dots \\ 1101 \dots \\ \hline 1111111 \end{array}$$

Деление в CRC арифметике определяется аналогично с учетом того, что вычитание выполняется по правилам CRC арифметики. Например:

$$\begin{array}{r} 11010110110110000 \mid 10011 \\ 10011 \\ \hline 10011 \\ 10011 \\ \hline 00001 \\ 00000 \\ \hline 00010 \\ 00000 \\ \hline 00101 \\ 00000 \\ \hline 01011 \\ 00000 \\ \hline 10110 \\ 10011 \\ \hline 01010 \\ 00000 \\ \hline 10100 \\ 10011 \\ \hline 01110 \\ 00000 \\ \hline 1110 = \text{Остаток} \end{array}$$

В CRC арифметике считается что число A делится на число B , если его можно получить из нуля путем некоторого числа сложений сдвинутого числа B . Например, пусть $A=0111010110_2$ и $B=11_2$, тогда

$$\begin{array}{r} 0111010110 \\ + \quad \quad \quad 11 \\ + \quad \quad 11 \\ + \quad 11 \\ + 11 \end{array}$$

Перед началом вычисления CRC исходное сообщение следует дополнить W нулями справа и выполнить деление по правилам CRC-арифметики. Рассмотрим пример:

Исходное сообщение: 1101011011
 Генераторный полином: 10011
 Сообщение, дополненное H битами: 11010110110000

$$\begin{array}{r}
 1101011011 \leftarrow \text{Исходное сообщение} \\
 1101011011 + 0000 \leftarrow \text{Выровненное сообщение} \\
 \hline
 11010110110000 \mid 10011 \leftarrow \text{Полином} \\
 10011 \\
 \hline
 10011 \\
 10011 \\
 \hline
 00001 \\
 00000 \\
 \hline
 00010 \\
 00000 \\
 \hline
 00101 \\
 00000 \\
 \hline
 01011 \\
 00000 \\
 \hline
 10110 \\
 10011 \\
 \hline
 01010 \\
 00000 \\
 \hline
 10100 \\
 10011 \\
 \hline
 01110 \\
 00000 \\
 \hline
 1110 = \text{Остаток} = \text{Контрольная сумма!}
 \end{array}$$

Частное (оно отбрасывается, т.к. не представляет интереса)

Как видно, контрольная сумма (CRC) в этом примере равна 1110. Как правило, контрольная сумма добавляется к исходному сообщению (в нашем примере передаваемое сообщение будет равно 11010110111110) и полученное расширенное сообщение передается через канал связи.

На другом конце канала приемник может сделать одно из возможных действий (оба варианта совершенно равноправны):

1. Выделить текст собственно сообщения, вычислить для него контрольную сумму (не забыв при этом дополнить сообщение H битами), и сравнить ее с переданной.
2. Вычислить контрольную сумму для всего переданного сообщения (без добавления нулей), и посмотреть, получится ли в результате нулевой остаток.