Практика безопасного программирования ПЛК: 20 золотых правил

10.01.2022
Версия перевода: 1.0

https://plc-security.com/
Перевод: https://oscat.ru/

Оглавление

Комментарий переводчиков

Данный документ включает в себя рекомендации по безопасному программированию ПЛК. Безопасное программирование – это подход к разработке программного обеспечения, целью которого является обеспечение устойчивости к воздействию вредоносных программ, эксплуатации уязвимостей, несанкционированного доступа и т.д. Подобный подход давно применяется в IT-сфере, но в области АСУ ТП серьезный интерес к информационной безопасности начал появляться только в течение последнего десятилетия.

Авторы документа не пытаются переложить практики информационной защиты из мира IT на АСУ ТП – вместо этого они рассматривают типичные для ПЛК/SCADA функционал и инструменты, показывая, как с их помощью можно повысить безопасность, надежность и простоту обслуживания системы управления. Поэтому, на наш взгляд, он может быть интересен широкому кругу инженеров АСУ ТП.

Официальный сайт проекта: https://plc-security.com/

20 золотых правил

1. Используйте модульный подход при разработке ПО

Разделяйте код приложения на модули с помощью функциональных блоков и подпрограмм. Тестируйте эти модули независимо друг от друга.

2. Отслеживайте режим работы ПЛК

В условиях нормальной работы системы приложение ПЛК должно находиться в режиме RUN («запущено»). Если оно находится в другом режиме – то оператор должен получить соответствующий аварийный сигнал.

3. Перенесите всю логику приложения в ПЛК (в тех случаях, где это возможно)

Перенесите как можно больше операций (суммирование, масштабирование, интегрирование и т.д.) из HMI в ПЛК. Время цикла HMI обычно недостаточно для таких операций.

4. Используйте флаги ошибок ПЛК для определения проблем с целостностью приложения

Используйте флаги ошибок для определения проблем в расчетах, связанных с делением на ноль, переполнением и т.д.

5. Используйте криптографические хеш-функции и/или контрольные суммы для подтверждения целостности и достоверности приложения ПЛК

Используйте криптографические хеш-функции или контрольные суммы (если хеш-функции не поддерживаются вашим ПЛК) для подтверждения целостности и достоверности приложения ПЛК. Формируйте сигнал тревоги, если результат проверки не совпадает с ожидаемым.

6. Проверяйте на корректность значения счетчиков и таймеров

Если уставки счетчиков/таймеров передаются в ПЛК (например, из HMI), то они должны проверяться на корректность (например, на неравенство нулю и отсутствию возможности выхода за ноль при обратном отсчете).

7. Проверяйте на соответствие «парные» входы-выходы и формируйте тревоги при их несовпадении

Если в системе присутствуют «парные» входы-выходы (т.е. сигналы, которые не могут изменяться одновременно – например, дискретные входы, к которым подключены кнопки запуска и остановки процесса), то проверяйте, что их состояния никогда не совпадают. В противном случае – формируйте тревогу для оператора. Продумайте возможность независимой обработки таких сигналов и добавления таймеров задержек, чтобы избежать повреждения исполнительных механизмов при неожиданном переключении выходов.

8. Проверяйте на корректность данные, полученные от HMI

Данные, вводимые на HMI, должны проверяться на корректность не только на стороне самого HMI, но и в приложении ПЛК. Это позволит избежать непредвиденных ошибок и предотвратить применение некорректных значений, а также сформировать предупреждения для оператора.

9. Проверяйте индексы массивов

Проверяйте нижние и верхние индексы массивов, чтобы избежать выхода за их границы и «ошибки на единицу».

10. Распределите используемые регистры по типу доступа (чтение/запись)

Распределите используемые регистры по типу доступа, чтобы организовать проверку корректности записанных данных, избежать переполнения буфера и не допустить записи значений извне в регистры, предназначенные только для чтения.

11. Проверяйте достоверность измерений

Разрабатывайте систему автоматизации таким образом, чтобы иметь возможность проводить несколько независимых проверок полученных измерений для подтверждения их достоверности.

12. Проверяйте реалистичность данных, вводимых оператором

Убедитесь, что операторы могут ввести только те значения параметров, которые имеют физический смысл. Запускайте таймеры при начале выполнения операций и формируйте сигналы тревог, если они не успевают выполниться за заданное время. Информируйте оператора о неожиданном бездействии системы.

13. Отключите неиспользуемые сетевые порты и протоколы

Контроллеры и коммуникационные модули, как правило, поддерживают несколько сетевых протоколов, которые обычно по умолчанию включены. Отключите сетевые порты и протоколы, которые не используются в данном конкретном приложении.

14. Ограничьте обмен данными с другими устройствами

Ограничьте тип подключений и наборы параметров, которые доступны для сторонних устройств. Типы подключений для каждого интерфейса должны быть четко определены, чтобы только необходимые в рамках конкретной задачи параметры были доступны для чтения и записи.

15. Определите безопасное состояние выходов при перезапуске ПЛК

Определите безопасное состояние выходов при перезапуске ПЛК (например, может потребоваться их включить, выключить или сохранить в текущем состоянии).

16. Измеряйте время цикла ПЛК и отображайте его на HMI в виде трендов

Измеряйте время цикла ПЛК каждые 2-3 секунды и отображайте его на HMI в виде тренда.

17. Регистрируйте время непрерывной работы ПЛК и отображайте его на HMI

Регистрируйте время непрерывной работы ПЛК, чтобы определить моменты, когда он перезапускался. Отображайте это время на HMI для диагностики.

18. Организуйте логирование остановок работы ПЛК по исключению и отображайте информацию об этом на HMI

Организуйте логирование остановок работы ПЛК по исключению и отображайте на HMI соответствующие аварийные сообщения, чтобы помочь оператору понять, какие действия необходимо совершить перед перезапуском ПЛК. Настройте в ПЛК синхронизацию времени для отображения точных меток времени этих событий.

19. Отслеживайте потребление памяти ПЛК и отображайте его на HMI

Отслеживайте потребление памяти ПЛК и отображайте его на HMI в виде тренда. Также отображайте усредненный ожидаемый уровень потребления памяти (который должен быть измерен заранее) в качестве точки отсчета для анализа.

20. Отслеживайте несрабатывания и ложные срабатывания для сигналов критических тревог

Определите условия возникновения критических тревог и отслеживайте их возникновение. Также отслеживайте условия срабатывания и состояния битов тревоги для любого отклонения в производственном процессе.

1. Используйте модульный подход при разработке ПО

Разделяйте код приложения на модули с помощью функциональных блоков и подпрограмм. Тестируйте эти модули независимо друг от друга.

Компонент, для которого обеспечивается безопасность / надежность	Ответственный
Логика приложения ПЛК	Разработчик приложения

Рекомендация

Не сосредотачивайте всю логику приложения ПЛК в одном компоненте – например, организационном блоке или программе. Вместо этого разделите ее на отдельные функциональные блоки и подпрограммы. Для каждого функционального блока / подпрограммы отслеживайте время выполнения и размер в килобайтах.

Создавайте ПО в виде набора модулей, которые функционируют независимо друг от друга. Это упростит проверку входных данных, обеспечение контроля доступа и т.д.

Модульный подход к разработке также облегчает тестирование и контроль целостности. Если код модуля был тщательно протестирован, то при загрузке модуля в память может производиться расчет хеша его кода и сравнение с хешем исходного кода модуля (в случае, если ПЛК поддерживает такую возможность). Такие проверки могут проводиться во время приемочных и квалификационных испытаний, а также в том случае, если целостность кода приложения вызывает сомнения после какого-либо инцидента.

Пример

Логика управления газовой турбиной включает в себя «запуск», «управление входным направляющим аппаратом», «управление клапаном сброса» и другие операции. Анализ этих операций и выделение в них общих фрагментов позволяет многократно использовать стандартные программные модули при разработке приложения. Это, в свою очередь, позволяет оперативно устранять неполадки в случае аварий и других инцидентов.

Пользовательские функциональные блоки, которые прошли тщательное тестирование, могут быть повторно использованы без изменения исходного кода (а также возможно формирование предупреждений при попытке изменить этот код) и защищены от непреднамеренных или предумышленных изменений с помощью пароля и/или цифровой подписи.

Обоснование

Причина	Комментарий
Безопасность	Облегчает обнаружение вредоносного кода при внесении изменений в приложение ПЛК. Помогает в стандартизации логики, обеспечении ее согласованности и защите от несанкционированных изменений
Надежность	Помогает контролировать последовательность выполнения кода и избегать ситуаций, в которых работа приложения может стать некорректной
Обслуживание	Модульность приложения облегчает его отладку (так как модули можно тестировать независимо друг от друга), поддержку и доработку. Кроме того, модули можно переиспользовать при разработке других приложений, что позволяет создавать общую кодовую базу. Это упрощает поиск и устранение неисправностей для обслуживающего персонала

Ссылки

Стандарт	Раздел
MITRE ATT&CK for ICS	Tactic: TA002 - Execution Technique: T0844 - Program Organization Units
ISA 62443-3-3	SR 3.4: Software and information integrity
ISA 62443-4-2	CR 3.4: Software and information integrity
ISA 62443-4-1	SI-2: Secure coding standards
MITRE CWE	CWE-1120: Excessive Code Complexity CWE-653: Insufficient Compartmentalization

2. Отслеживайте режим работы ПЛК

В условиях нормальной работы системы приложение ПЛК должно находиться в режиме RUN («запущено»). Если оно находится в другом режиме – то оператор должен получить соответствующий аварийный сигнал.

Компонент, для которого обеспечивается безопасность / надежность	Ответственный
Логика приложения ПЛК	Системный интегратор / сервисный инженер

Рекомендация

Если ПЛК не находится в режиме RUN (например, его перевели в режим программирования), то в коде приложения можно реализовать отслеживание и индикацию этого состояния. Некоторые ПЛК вычисляют контрольную сумму приложения и позволяют отследить изменение кода приложения, но даже в случае отсутствия этой возможности должна быть реализована индикация режима работы приложения для оператора:

- если ПЛК не находится в состоянии RUN, то оператор должен получить соответствующее оповещение. Если он знает, что в данный момент с ПЛК кто-то работает – то просто сквитирует его.
- HMI должен быть сконфигурирован таким образом, чтобы повторно выдать это оповещение в конце смены, если приложение к тому моменту снова не будет запущено. Это позволяет избежать ситуации, в которой наладчик забыл перевести ПЛК в режим RUN после окончания своей работы – что может привести к остановке технического процесса.

Исключение: если приложение находится на стадии тестирования или отладки, то можно отключить эти оповещения – в том случае, если объект автоматизации изолирован от других сетей и систем.

Пример

Если ПЛК не имеет аппаратного переключателя для изменения режима работы, то рекомендуется ограничить возможность его перепрограммирования на уровне ПО (например, защитив функцию выгрузки и загрузки проекта с помощью пароля).

Обоснование

Причина	Комментарий
Безопасность	Режим работы определяет, можно ли изменить приложение ПЛК. Например, ПЛК Allen Bradley поддерживает следующие режимы: RUN (приложение запущено) / PROGram (ПЛК в режиме программирования, приложение не выполняется) / REMote (приложение запущено; в этом режиме возможно обновление программы и изменение режима работы через сетевой интерфейс ПЛК – в режимах RUN и PROgram изменение режима работы возможно только с помощью аппаратного переключателя)

Ссылки

Стандарт	Раздел
MITRE ATT&CK for ICS	Tactic: TA009 - Inhibit Response Function Technique: T0858 - Utilize/Change Operating Mode
ISA/IEC 62443-4-1	SI-1: Security implementation review

3. Перенесите всю логику приложения в ПЛК (в тех случаях, где это возможно)

Перенесите как можно больше операций (суммирование, масштабирование, интегрирование и т.д.) из HMI в ПЛК. Время цикла HMI обычно недостаточно для таких операций.

Компонент, для которого обеспечивается безопасность / надежность	Ответственный
Логика приложения ПЛК	Разработчик приложения Системный интегратор / сервисный инженер

Рекомендация

HMI обычно предоставляет определенный функционал для программирования (макросы, скрипты), который может быть полезным при создании визуализации и настройке тревог. Но некоторые разработчики используют эти средства для выполнения операций, которые должны оставаться на стороне ПЛК, чтобы сохранить его приложение целостным.

Обработка значений должна проводиться как можно ближе к месту их получения (то есть на уровне ПЛК). HMI обычно не может обеспечить приемлемого быстродействия для обработки сигналов. При передаче данных между ПЛК и HMI всегда возникают задержки. Кроме того, если вся логика приложения выполняется на стороне ПЛК, то не нужно заботиться о хранении значений в HMI – после перезагрузки они будут вычитаны из ПЛК.

Если же на стороне HMI используется какой-либо программный код – то он не должен быть связан с функциями надежности и безопасности (например, блокировками, таймерами задержки, контролем доступа и т.д.).

Для хранения исторических данных вместо HMI лучше использовать отдельный сервер архивации. HMI будет отправлять запросы к его базе данных и сравнивать полученные значения с теми, которые сохраняются в ПЛК. Если разница в значениях превышает допустимую погрешность – то следует сформировать соответствующее предупреждение.

Примеры

- Условия блокировки/разблокировки кнопок управления должны проверяться на стороне ПЛК, иначе управление со стороны HMI (или через сетевые интерфейсы) будет возможно даже в том случае, если эти условия не выполняются;
- Обработка таймеров (например, таймеров задержки запуска двигателя, таймеров открытия/закрытия клапана и т.д.) должна выполняться не на уровне HMI, а на уровне ПЛК, который непосредственно управляет этими исполнительными механизмами;
- Пороговые значения для формирования сигналов тревог должны обрабатываться в ПЛК, хотя отображение тревог происходит на HMI;

- Анимация уровня жидкости в резервуаре: ПЛК, который получает сигналы расхода на входе и выходе из резервуара, может легко рассчитать текущий объем жидкости в резервуаре. Это можно реализовать и на стороне HMI, но значения для расчета все равно будут получены от ПЛК. Кроме того, при перезагрузке HMI или проблемах с каналом связи часть значений будет утеряна – это потребует передачи меток времени вместе со значениями и усложнения логики расчета.

Обоснование

Причина	Комментарий
Безопасность	1. Обеспечение целостности приложения. Обычно изменения в проекте HMI плохо документируются (особенно изменения, которые вносятся в последний момент на этапе ввода в эксплуатацию) и через определенное время «рассинхронизация» приложения в ПЛК и HMI может стать очень сильной. Это затуманивает общую картину управления техническим процессом. Кроме того, скрипты HMI обычно тяжело отлаживать, так как для них нет характерных для ПЛК инструментов: «принудительной записи» сигналов (forcing), списка измененных значений и т.д. Поэтому перенос части кода на сторону HMI сильно усложняет управление изменениями проекта 2. Злоумышленнику сложнее манипулировать данными, расположенными на нескольких ПЛК, чем данными, размещенными в одном HMI, подключенном к этим ПЛК 3. Если логика обработки блокировок кнопок управления перенесена на сторону HMI, то злоумышленнику проще влиять на технический процесс, так как экран HMI уже предоставляет ему нужные для этого средства
Надежность	1. Расчеты будут более быстрыми и точными, если будут производиться максимально близко к полевому уровню. Кроме того, результаты расчетов будут доступны после перезагрузки ПЛК, поскольку сохраняются в его энергонезависимой памяти (и в целом ПЛК перезагружается значительно реже HMI) 2. Разделение логики блокировок на несколько отдельных фрагментов (ПЛК, HMI, SCADA), которые программируются разными способами, усложняет синхронизацию изменений в ПО между этими устройствами – например, при изменении блокировок в HMI можно забыть внести изменения в проект ПЛК, что приведет к сбоям в работе оборудования
Обслуживание	Код ПЛК проще анализировать и переносить на другие устройства (по сравнению с кодом скриптов HMI)

Ссылки

Стандарт	Раздел
MITRE ATT&CK for ICS	Tactic: TA010 - Impair Process Control Technique: T0836 - Modify Parameter
ISA 62443-3-3	SR 3.6: Deterministic Output
ISA 62443-4-2	CR 3.6: Deterministic Output

4. Используйте флаги ошибок ПЛК для определения проблем с целостностью приложения

Используйте флаги ошибок для определения проблем в расчетах, связанных с делением на ноль, переполнением и т.д.

Компонент, для которого обеспечивается безопасность / надежность	Ответственный
Логика приложения ПЛК	Разработчик приложения Системный интегратор / сервисный инженер

Рекомендация

Если приложение ПЛК работало нормально, но внезапно в нем произошло деление на 0 – необходимо тщательно исследовать эту ситуацию. Если ПЛК производит обмен данными с другим ПЛК и в процессе этого произошло деление на 0 – необходимо разобраться в том, что именно произошло.

Большинство разработчиков посчитают это математической ошибкой и не усомнятся в правильности своего кода, позволив перейти ПЛК в состояние отказа. В процессе разработки программист должен тестировать свои программные модули (функциональные блоки, подпрограммы и т.д.), используя в качестве входных данных в том числе и некорректные данные, выходящие за диапазон ожидаемых значений. Это называется модульным тестированием.

Рекомендуется выделять отдельные изолированные сегменты памяти для прошивки ПЛК, его приложения и стеков коммуникационных протоколов. Стеки протоколов должны быть протестированы на предмет уязвимостей – например, передачи пакетов со специфическими заголовками и т.д.

Примеры

Сбои в работе ПЛК, вызванные выходом данных за допустимый диапазон, крайне распространены. К ним, например, можно отнести доступ к массиву, когда индекс элемента выходит за границы массива, вызов таймера с отрицательным значением времени, деление на 0 и т.д.

Типичные флаги ошибок, которые представляют интерес:

- деление на 0;
- переполнение счетчиков;
- отрицательные значения в качестве уставок счетчиков или таймеров;
- недопустимые задержки в цикле обработки входов-выходов.

Обоснование

Причина	Комментарий
Безопасность	Атаки на ПЛК могут включать изменение его логики (в том числе, добавление новой логики), загрузку новых рецептов, запуск каких-либо функций или подпрограмм и т.д. Поскольку большинство ПЛК не предоставляет средств для проверки целостности приложения, то флаги ошибок в вычислениях могут быть косвенным индикатором изменения программы
Надежность	Внимательное отношение к флагам ошибок может позволить найти проблему в коде программы (или проблемы, связанные с устройствами, подключенными к ПЛК – например, если они передают некорректные значения). Кроме того, это позволяет быстрее локализовать ошибку при ее возникновении

Ссылки

Стандарт	Раздел
MITRE ATT&CK for ICS	Tactic: TA010 - Impair Process Control Technique: T0836 - Modify Parameter
ISA 62443-3-3	SR 3.5: Input Validation SR 3.6: Deterministic Output
ISA 62443-4-2	CR 3.5: Input Validation CR 3.6: Deterministic Output
ISA 62443-4-1	SI-2: Secure coding standards SVV-1: Security requirements testing
MITRE CWE	CWE-128: Wrap-around CWE-190: Integer Overflow CWE-369: Divide by Zero CWE-754: Improper Check for Unusual or Exceptional Conditions

5. Используйте криптографические хеш-функции и/или контрольные суммы для подтверждения целостности и достоверности приложения ПЛК

Используйте криптографические хеш-функции или контрольные суммы (если хеш-функции не поддерживаются вашим ПЛК) для подтверждения целостности и достоверности приложения ПЛК. Формируйте сигнал тревоги, если результат проверки не совпадает с ожидаемым.

Компонент, для которого обеспечивается безопасность / надежность	**Ответственный**
Логика приложения ПЛК	Разработчик приложения Системный интегратор / сервисный инженер

Рекомендация

А) Контрольные суммы

Если хеш-функции не поддерживаются вашим ПЛК – используйте контрольные суммы. Некоторые ПЛК генерируют контрольную сумму при загрузке приложения. Контрольная сумма приложения должна быть задокументирована поставщиком оборудования / системным интегратором после проведения приемочных испытаний и указана в договоре о сервисном обслуживании.

Если ПЛК не поддерживает расчет контрольной суммы для своего приложения – то она может быть сгенерирована на HMI или АРМ инженера. После этого следует периодически повторять ее расчет (например, раз в сутки) и сравнивать полученный результат с «эталонным», чтобы убедиться в их совпадении. Этого будет достаточно, чтобы узнать, что кто-то пытается изменить приложение ПЛК – хотя, конечно, без оповещений в реальном времени.

Значение контрольной суммы можно считать в приложении ПЛК и настроить генерацию сигнала тревоги при его изменении, периодическую отправку этого значения на сервер архивации и т.д.

Б) Хеш-функции

Сам ПЛК обычно не обладает достаточным ресурсами для расчета или проверки хешей в процессе работы. Попытка выполнения такой операции может привести к сбою в работе ПЛК. Но среда разработки может поддерживать расчет хеша и его сохранения в ПЛК или на ПК.

Примеры

Производители ПЛК, которые поддерживают расчет контрольной суммы приложения:

- Siemens
- Rockwell

Также для генерации контрольной суммы может использоваться отдельное ПО:

- Version Dog
- Asset Guardian
- PAS

Пример для ПЛК Siemens

Для расчета контрольной суммы приложения ПЛК Siemens S7-1500 используйте скрипт SAT-Checksum (выполняемый в утилите SIMATIC Automation Tool). Для проверки текущей контрольной суммы в приложении ПЛК используйте функциональный блок GetChecksum. Сравнение текущей и сохраненной контрольной суммы можно произвести с помощью функции Datalog.

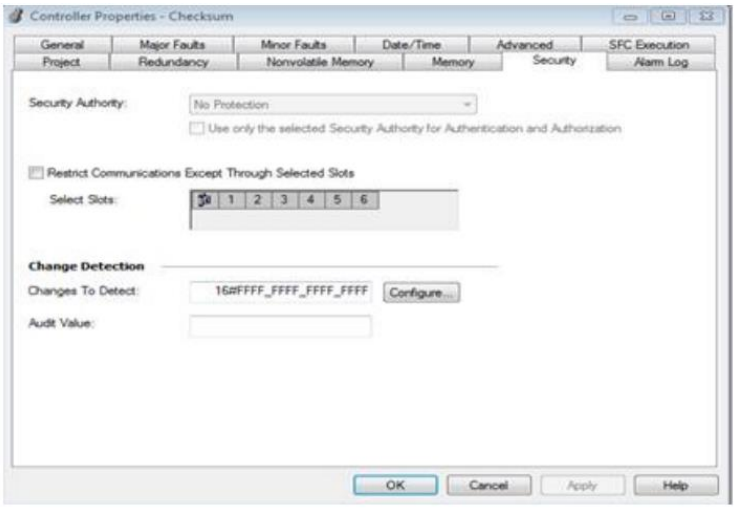
	Date	UTC Time	Referenz	Aktuell
1	11/21/2019	9:55:11	84 2A 76 DF 5B 31 F4 16	FF 2C EA 71 44 D7 81 04
2	11/21/2019	9:57:33	FF 2C EA 71 44 D7 81 04	FF 2C EA 71 44 D7 81 04
3	11/21/2019	9:58:17	FF 2C EA 71 44 D7 81 04	5B 7C 57 7E E2 3E EF C3
4	11/21/2019	9:58:36	FF 2C EA 71 44 D7 81 04	5B 7C 57 7E E2 3E EF C3
5	11/21/2019	9:58:44	5B 7C 57 7E E2 3E EF C3	5B 7C 57 7E E2 3E EF C3

Пример для ПЛК Rockwell

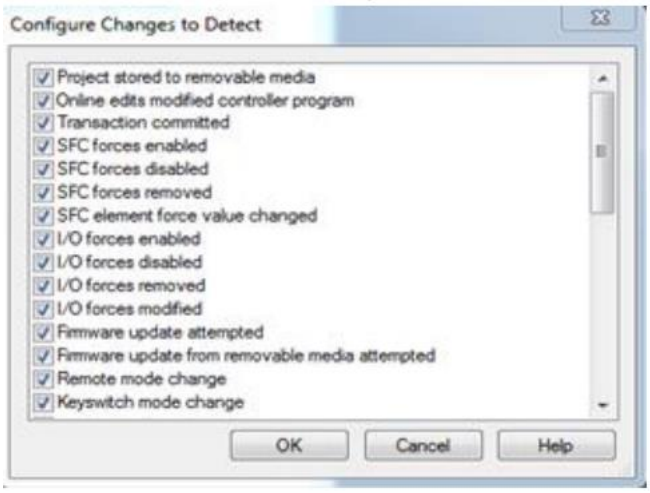
Этот упрощенный пример демонстрирует, как можно разработать механизм обнаружения изменения ПО для своей АСУ ТП. Пример разработан для ПЛК ControlLogix (Rockwell Automation) и не является полностью законченным; тем не менее, он демонстрирует, как получить в приложении сигнал об изменении контрольной суммы этого приложения. После этого появляется возможность сравнения этой контрольной суммы с «эталонной», генерации тревоги при их отличии, передачи рассчитанной контрольной суммы на сервер архивации и т.д.

Такой подход позволяет использовать существующие инструменты для получения информации о незапланированном изменении приложения ПЛК. Разумеется, данный пример должен быть адаптирован для применения в рамках конкретного используемого ПО.

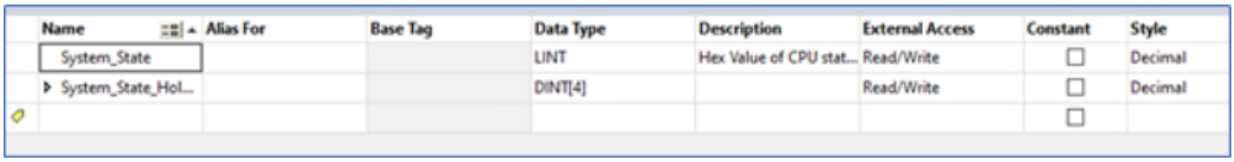
1. В свойства ПЛК на вкладке Security нажмите кнопку Configure, расположенную рядом с полем Changes To Detect.



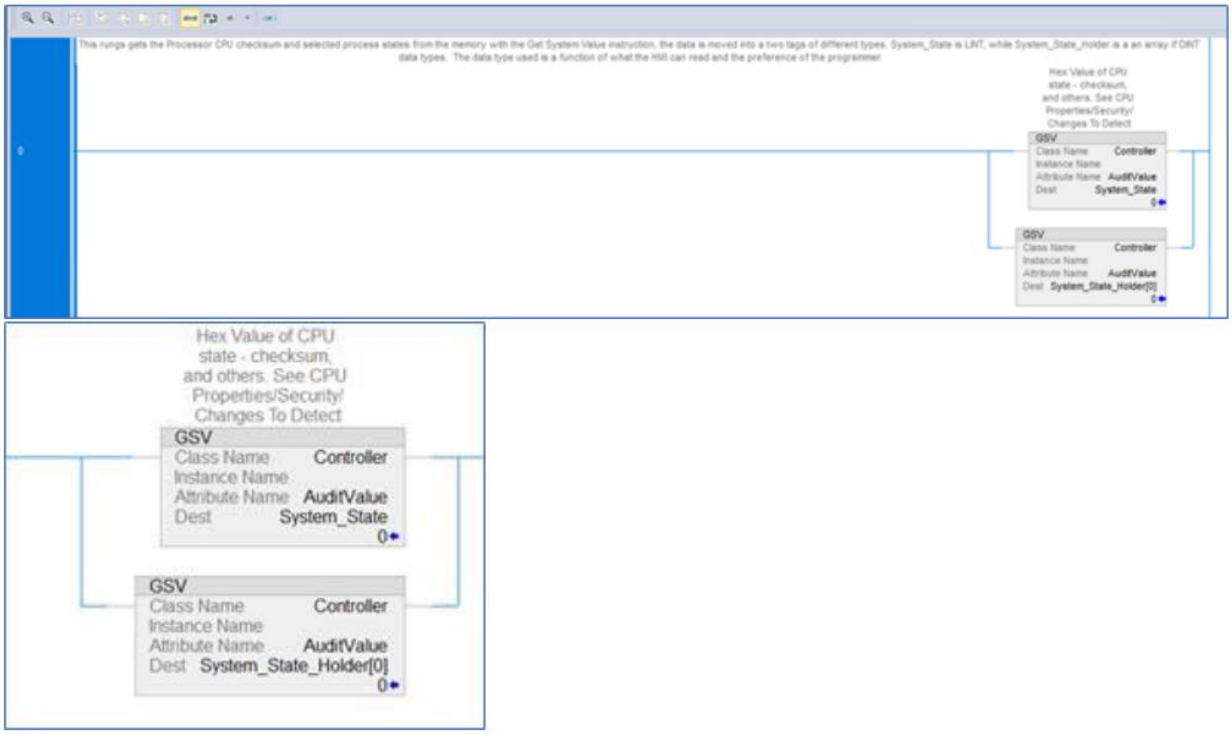
2. В открывшемся окне проставьте галочки рядом со всеми компонентами, изменения которых должны учитываться при расчете контрольной суммы.



3. Создайте тег типа LINT (или массив из двух элементов типа DINT), который будет использоваться для хранения состояния ПЛК.



4. Используйте функцию Get System Values (GSV), чтобы считать текущее состояние ПЛК (которое включает в себя контрольную сумму) и скопируйте его в созданный тег, который может использоваться в логике приложения или быть передан на HMI.



Обоснование

Причина	Комментарий
Безопасность	Информация об изменении приложения ПЛК имеет важное значение для обнаружения факта атаки на систему автоматизации и проверки ее безопасности в том случае, если доступ к ней был скомпрометирован
Надежность	Хеш и контрольная сумма позволяют подтвердить, что в ПЛК исполняется приложение, загруженное поставщиком оборудования или системным интегратором

Ссылки

Стандарт	Раздел
MITRE ATT&CK for ICS	Tactic: TA002 - Execution, TA010 - Impair Process Control Technique: T0873 – Project File Infection, T0833 - Modify Control Logic
ISA 62443-3-3	SR 3.4: Software and information integrity
ISA 62443-4-2	CR 3.4: Software and information integrity EDR 3.12: Provisioning product supplier roots of trust
ISA 62443-4-1	SI-1: Security implementation review SVV-1 Security requirements testing
MITRE CWE	CWE-345: Insufficient Verification of Data Authenticity • (child) CWE-353: Missing Support for Integrity Check • (child) CWE-354: Improper Validation of Integrity Check Value

6. Проверяйте на корректность значения счетчиков и таймеров

Если уставки счетчиков/таймеров передаются в ПЛК (например, из HMI), то они должны проверяться на корректность (например, на неравенство нулю и отсутствию возможности выхода за ноль при обратном отсчете).

Компонент, для которого обеспечивается безопасность / надежность	Ответственный
Переменные ПЛК	Разработчик приложения Системный интегратор / сервисный инженер Разработчик инструментального ПО

Рекомендация

Технически счетчики и таймеры могут работать с любыми уставками. Но допустимый диапазон значений для этих уставок должен быть выбран в соответствии с требованиями конкретного технического процесса.

Если другие устройства (например, HMI) могут записывать значения уставок для счетчиков и таймеров в ПЛК, то требуется:

- не допускать ситуации, при которой запись этих значений происходит «напрямую», без проверок на корректность;
- добавить проверки на корректность значений уставок в приложение ПЛК.

Провести проверку значений уставок счетчиков и таймеров в программе ПЛК довольно просто, потому что ПЛК знает текущее состояние технического процесса и его особенности. Соответственно, можно проверить значения уставок на корректность и убедиться, что команда на изменение получена в подходящий момент времени.

Пример

Во время загрузки проекта уставки счетчиков и таймеров обычно инициализируются значениями по умолчанию.

Если в проекте есть таймер задержки срабатывания тревоги на 1.3 секунды, но злоумышленник перенастроит его на 5 минут – то оператор своевременно не получит оповещение об аварийной ситуации, что может привести к нарушению технического процесса или более катастрофическим последствиям.

Если в проекте есть счетчик, который останавливает технический процесс после выполнения 10000 операций, но злоумышленник перенастроит его на 11000 – то это может привести к нарушению технического процесса.

Обоснование

Причина	Комментарий
Безопасность	Если значения уставок записываются «напрямую» в логику ПЛК из других устройств (HMI и т.д.) без какой-либо проверки – то это создает необоснованно высокий уровень доверия к этим устройствам, что может стать угрозой нарушения технического процесса (см. примеры выше)
Надежность	Проверка корректности уставок позволяет детектировать ситуации, в которых оператор случайно ввел некорректное значение уставки
Обслуживание	Документирование и автоматическая проверка допустимых диапазонов уставок счетчиков и таймеров облегчает отладку и доработку приложения

Ссылки

Стандарт	Раздел
MITRE ATT&CK for ICS	Tactic: TA010 - Impair Process Control Technique: T0836 - Modify Parameter
ISA 62443-3-3	SR 3.5: Input Validation
ISA 62443-4-2	CR 3.5: Input Validation
ISA 62443-4-1	SI-2: Secure coding standards SVV-1: Security requirements testing

7. Проверяйте на соответствие «парные» входы-выходы и формируйте тревоги при их несовпадении

Если в системе присутствуют «парные» входы-выходы (т.е. сигналы, которые не могут изменяться одновременно – например, дискретные входы, к которым подключены кнопки запуска и остановки процесса), то проверяйте, что их состояния никогда не совпадают. В противном случае – формируйте тревогу для оператора. Продумайте возможность независимой обработки таких сигналов и добавления таймеров задержек, чтобы избежать повреждения исполнительных механизмов при неожиданном переключении выходов.

Компонент, для которого обеспечивается безопасность / надежность	**Ответственный**
Переменные ПЛК	Разработчик приложения Системный интегратор / сервисный инженер

Рекомендация

Хотя «парные» входы-выходы не должны находиться в одинаковых состояниях – это возможно из-за сбоя приложения или действий злоумышленника. Несмотря на это, разработчики часто игнорируют такую ситуацию и никак не проверяют состояния этих входов-выходов.

Эти проверки проще всего выполнить на стороне ПЛК, потому что ПЛК имеют полную информацию о текущем состоянии технологического процесса. Для облегчения работы с «парными» сигналами имеет смысл использовать для них последовательную нумерацию (например, входы 1 и 2).

Еще одна возможная проблема, связанная с «парными» выходами – если они переключаются не одновременно, но настолько быстро, что это может привести к повреждению исполнительных механизмов. В этом случае следует добавить таймер задержки.

Примеры

Примеры «парных» сигналов:

- Сигналы запуска и остановки
  - Независимая обработка сигналов: используйте отдельные выходы для подачи команд на запуск и остановку. В логике должна быть предусмотрена блокировка их одновременного срабатывания. Для злоумышленника гораздо сложнее будет установить два выхода в различные состояния, чем переключить один выход;
  - Таймер задержки переключения: добавьте таймер задержки, который позволит избежать быстрых последовательных включений-отключений оборудования (что может привести к его неисправности).
- Сигналы «Вперед» и «Назад»;
- Сигналы «Открыть» и «Закрыть».

Примеры «парных» сигналов, которые могут привести к повреждению оборудования: если управление оборудованием производится через дискретные входы ПЛК, которые влияют на состояния дискретных выходов, то это дает злоумышленнику возможность нанести ущерб оборудованию. Доказательством этому является эксперимент «Тест генератора Аврора», проведенный в 2007 году национальной лабораторией штата Айдахо.

Обоснование

Причина	Комментарий
Безопасность	1. Если приложение ПЛК не обрабатывает случай одновременного срабатывания «парных» сигналов – то это может быть одним из векторов атаки на систему автоматизации 2. Проверка «парных» сигналов также позволяет обнаружить ошибку в ПО или действиях оператора 3. Подобные проверки позволяют предотвратить атаки, направленные на вывод из строя исполнительных механизмов
Надежность	1. Одновременное срабатывание «парных» сигналов может свидетельствовать о неисправности датчиков, неправильном подключении оборудования или механических проблемах (например, «залипании» переключателя) 2. Быстрое переключение выходов, управляющих исполнительными механизмами, может быть вызвано ошибкой в программе. Проверка «парных» сигналов может предотвратить подобные ситуации

Ссылки

Стандарт	Раздел
MITRE ATT&CK for ICS	Tactic: TA010 - Impair Process Control Technique: T0836 - Modify Parameter, T0806 - Brute Force I/O
ISA 62443-3-3	**SR 3.5**: Input Validation **SR 3.6:** Deterministic Output
ISA 62443-4-2	**CR 3.5:** Input Validation **CR 3.6:** Deterministic Output
ISA 62443-4-1	**SI-2:** Secure coding standards **SVV-1:** Security requirements testing
MITRE CWE	**CWE-754:** Improper Check for Unusual or Exceptional Conditions

8. Проверяйте на корректность данные, полученные от HMI

Данные, вводимые на HMI, должны проверяться на корректность не только на стороне самого HMI, но и в приложении ПЛК. Это позволит избежать непредвиденных ошибок и предотвратить применение некорректных значений, а также сформировать предупреждения для оператора.

Компонент, для которого обеспечивается безопасность / надежность	Ответственный
Переменные ПЛК	Разработчик приложения Системный интегратор / сервисный инженер

Рекомендация

Полученные данные (в частности, уставки и значения конфигурационных параметров) должны проверяться на принадлежность диапазону допустимых значений. Если же значение не укладывается в этот диапазон, то возможны следующие варианты обработки этой ситуации:

- зафиксировать значение на границе диапазона и сгенерировать сигнал тревоги;
- сохранить предыдущее корректное значение параметра и записать информацию об этом событии в лог для последующего анализа.

Примеры

Пример 1

Оператор вводит на HMI значение уставки для давления клапана. Допустимый диапазон значений составляет 0…100, и введенное значение передается в ПЛК и записывается в переменную V1.

1. В HMI должны быть установлены ограничения для вводимого значения: 0…100

2. В ПЛК должна быть реализована дополнительная проверка полученного значения:

```
IF V1 < 0 OR IF V1 > 100, SET V1 = 0
```

Это позволит избежать аварийной ситуации даже в том случае, если по каким-то причинам ПЛК получит от HMI некорректное значение.

Пример 2

Оператор вводит на HMI значение некого параметра, которое передается в ПЛК и записывается в переменную V2 типа INT (16-битное целое со знаком).

1. В HMI должны быть установлены ограничения для вводимого значения: -32768…32767 (в соответствии с диапазоном значений для типа INT)

2. В ПЛК должна быть реализована дополнительная проверка полученного значения, которая отслеживает, не произошло ли переполнение (для этого используется дополнительная переменная V3):

```
IF V2 = -32768 OR IF V2 = 32767 AND V3 != 0,
SET V2 = 0 AND SET V3 = 0 AND SET DataTypeOverflowAlarm = TRUE
```

Пример 3

Приводите текущее значение (PV), уставку (SP) и выходное значение (CV) ПИД-регулятора к «сырым» и согласованным величинам, чтобы избежать ошибок, связанных с масштабированием в конкретные физические единицы. Подобные ошибки могут привести к аварийным ситуациям.

Обоснование

Причина	Комментарий
Безопасность	1. Хотя на уровне HMI обычно можно добавить ограничение для вводимого значения – злоумышленник может отправить в ПЛК специально сформированный пакет с некорректным значением 2. Протоколы обмена, используемые ПЛК, обычно являются открытыми и их спецификации общедоступны. Соответственно, для злоумышленника не будет проблемой создать ПО для отправки специальных пакетов. Если у него есть доступ к сети АСУ – то с помощью анализа трафика он может сопоставить передаваемые пакеты и зависящие от них переменные ПЛК, что позволит ему манипулировать технологическим процессом. Проверка в ПЛК диапазона допустимых значений параметров позволяет частично предотвратить эту ситуацию и, по крайней мере, сохранить значения параметров в допустимых переделах, обеспечив безопасность работы системы

Ссылки

Стандарт	Раздел
MITRE ATT&CK for ICS	Tactic: TA010 - Impair Process Control Technique: T0836 - Modify Parameter
ISA 62443-3-3	**SR 3.5**: Input Validation **SR 3.6:** Deterministic Output
ISA 62443-4-2	**CR 3.5:** Input Validation **CR 3.6:** Deterministic Output
ISA 62443-4-1	**SI-2:** Secure coding standards **SVV-1:** Security requirements testing
MITRE CWE	**CWE-1320**: Improper Protection for Out of Bounds Signal Level Alerts

9. Проверяйте индексы массивов

Проверяйте нижние и верхние индексы массивов, чтобы избежать выхода за их границы и «ошибки на единицу».

Компонент, для которого обеспечивается безопасность / надежность	Ответственный
Переменные ПЛК	Разработчик приложения Системный интегратор / сервисный инженер

Рекомендация

В приложениях ПЛК часто используется косвенная адресация – в основном, при работе с массивами. В этих случаях одна из переменных представляет собой номер элемента массива, к которому производится обращение.

Примеры использования косвенной адресации:

- работа с ПЧВ, которые выполняют определенные действия в зависимости от заданной частоты;
- автоматическое переключение насосов по времени наработки.

При разработке приложений ПЛК обычно нет возможности получить флаг достижения конца массива, поэтому подобные проверки требуется организовать в коде самостоятельно – чтобы избежать выхода за границу массива, которая может привести к непредсказуемым последствиям.

Примеры

Описанный ниже подход можно оформить в виде функциональных блоков и повторно использовать в разных приложениях.

1. Используйте маску для доступа к элементу массива

Проверьте, является ли размерность вашего массива степенью двойки. Если нет – то дополните его до ближайшей степени двойки с помощью «пустых» элементов. Например, вам нужно создать массив из 5 элементов:

```
[21 31 41 51 61]
```

Тогда объявите массив из 8 элементов ($8 = 2^3$):

```
[xx xx 21 31 41 51 61 xx]
```

В данном случае смещение для индекса массива будет равно 2 (так как в начале массива расположено два «пустых» элемента). Обратите внимание, что нумерация элементов массива ведется с нуля. Таким образом, для обращения к третьему элементу значение индекса должно быть 3 + 2 = 5.

```
[xx xx 21 31 41 51 61 xx]
_______________^
```

При использовании индекса применяйте оператор AND вместе с маской, которая будет равна номеру последнего элемента массива. В данном случае массив состоит из 8 элементов, и при нумерации с нуля индекс последнего элемента будет равен 7.

```
6 AND 0x07 вернет 6
7 AND 0x07 вернет 7
8 AND 0x07 вернет 0 (!)
9 AND 0x07 вернет 1 (!)
```

Таким образом, даже если по каким-то причинам индекс выйдет за границы массива – не произойдет обращения к памяти другого объекта (что может привести к непредсказуемым сбоям в программе). Вместо этого будут перезаписаны «пустые» элементы данного массива.

2. Используйте особые значения для «пустых» элементов

Использование особых значений является опциональным. Вы можете использовать косвенную адресацию и без них, но особые значения позволяют легко отследить ошибку при формировании индекса, потому что вы получаете значение, которое не имеет смысла. Например, такими особыми значениями могут быть -1, 65535 и т.д. (то есть нужно выбрать значение, которое в рамках конкретной задачи никогда не должно появиться в массиве при нормальных обстоятельствах).

Массив, рассмотренный в примере 1, после добавления особых значений может выглядеть следующим образом:

```
[-1 -1 21 31 41 51 61 -1]
```

3. Обратитесь к элементу массива без маски и сохраните считанное значение

Теперь сформируйте индекс для доступа к массиву без маски и сохраните считанное значение. Пусть, например, вы обращаетесь к третьему элементу массива – он имеет значение 51:

```
[21 31 41 51 61]
__________^
__индекс = 3
```

4. Примените маску, после чего сравните значения

Примените смещение и маску (см. пп. 1), после чего сравните полученное значение с сохраненным (см. пп. 3).

4A. Корректная адресация

Пусть мы работаем с массивом из 5 элементов из пп. 1 и обращаемся к его 3-му элементу.

1. Вычисляем смещение: 3 + 2 = 5
2. Применяем маску: 5 AND 0x07 = 5
3. Обращаемся к элементу массива без маски:

```
[21 31 41 51 61]
__________^
__индекс = 3
```

4. Обращаемся к элементу массива с использованием маски:

```
[-1 -1 21 31 41 51 61 -1]
_________________^
__индекс = 5
```

Сравниваем значения – они совпадают. Это значит, что индекс массива находится в допустимом диапазоне.

4B. Ошибка адресации

Предположим, что из-за ошибки мы пытаемся обратиться к 7-му элементу массива (напомним, в качестве примера мы рассматриваем массив из 5 элементов).

1. Вычисляем смещение: 7 + 2 = 9
2. Применяем маску: 9 AND 0x07 = 1
3. Обращаемся к элементу массива без маски:

```
[21 31 41 51 61]
_____________^
__индекс = 7
```

4. Обращаемся к элементу массива с использованием маски:

```
[-1 -1 21 31 41 51 61 -1]
_____________^
__индекс = 1
```

Сравниваем значения и видим, что при доступе к массиву с помощью маски мы получили особое значение «-1» (см. п. 2). Это значит, что значение индекса выходит за границы массива.

5. Сформируйте предупреждения

Если значение элемента массива, полученное при доступе с маской, не соответствует сохраненному значению, полученному при доступе без маски – сформируйте предупреждение. Если при доступе с маской вы получили особое значение – то сформируйте другое предупреждение. Это позволит избежать «ошибки на единицу» и выхода за границы массива.

Обоснование

Причина	Комментарий
Безопасность	Большинство ПЛК не имеют встроенных средств для обработки ситуаций, в которых индекс массива выходит за его границы. Возможны два основных варианта потенциально опасных ситуаций, к которым это может привести: 1. При чтении из «несуществующего» элемента массива – программа продолжит выполняться с некорректными значениями 2. При записи в «несуществующий» элемент массива – будет перезаписан пользовательский код или другие данные Оба случая могут быть вызваны как ошибками в программе, так и действиями злоумышленника
Надежность	Позволяет выявить ошибки в программе

Ссылки

Стандарт	Раздел
MITRE ATT&CK for ICS	Tactic: TA010 - Impair Process Control Technique: T0836 - Modify Parameter
ISA 62443-3-3	**SR 3.5**: Input Validation **SR 3.6:** Deterministic Output
ISA 62443-4-2	**CR 3.5:** Input Validation **CR 3.6:** Deterministic Output
ISA 62443-4-1	**SI-2:** Secure coding standards **SVV-1:** Security requirements testing
MITRE CWE	**CWE-129:** Improper Validation of Array Index

10. Распределите используемые регистры по типу доступа (чтение/запись)

Распределите используемые регистры по типу доступа, чтобы организовать проверку корректности записанных данных, избежать переполнения буфера и не допустить записи значений извне в регистры, предназначенные только для чтения.

Компонент, для которого обеспечивается безопасность / надежность	Ответственный
Переменные ПЛК	Разработчик приложения Системный интегратор / сервисный инженер

Рекомендация

Временная память (также известная как «сверхоперативная память») является легко доступным вектором кибератак, если при работе с ней не используются советы, приведенные в данном пункте. Например, если значения регистров Modbus не проверяются на соответствие диапазону допустимых значений, то запись в них заведомо некорректных данных может привести к сбою в работе системы управления.

Как правило, к регистрам ПЛК могут обращаться другие устройства – например, некоторые регистры могут считываться HMI, другие – записываться SCADA-системой и т.д. Выделение для каждого устройства отдельной группы регистров упрощает их настройку в ПЛК или межсетевом экране – например, для одной группы можно настроить тип доступа «только чтение» и т.д.

Примеры функций, которые можно применять к таким группам регистров:

- настройка типа доступа «только чтение»;
- настройка типа доступа «чтение и запись»;
- проверка корректности записанных значений;
- обработка записанных значений (масштабирование и т.д.).

Регистры можно использовать как буфер для хранения данных. Например, если в случае действия злоумышленника значения переменных в памяти ПЛК будут изменены – то можно будет восстановить их из регистров. При работе с буфером необходимо проверять его целостность (при обновлении данных в буфере следует убедиться, что они обновились целиком – чтобы избежать ситуации, когда в буфере смешались «старые» и «новые» данные).

Комментарий

Основная и регистровая память используются по-разному. В основной памяти хранятся значения, которые использует выполняемая программа. Регистровая память является временной памятью, в которой размещаются значения, полученные от других устройств, перед их копированием в основную память. Тем не менее, значения, хранящиеся в регистрах, также могут повлиять на логику работы программы.

Примеры уязвимостей

По материалам статьи G. P. H. Sandaruwan, P. S. Ranaweera, Vladimir A. Oleshchuk. PLC Security and Critical

- в ПЛК Siemens в качестве сверхоперативной памяти обычно используются флаги 200.0…255.7 (см. статью). Если злоумышленник произведет перезапись флагов в этой области – то это может привести к ошибкам в работе ПЛК (в зависимости от того, какие флаги будут изменены);
- предположим, злоумышленник получил доступ к одному из устройств, подключенных к ПЛК. В этом случае он может записывать в регистровую память произвольные значения, что может привести к сбоям в работе системы управления;
- если в программе отсутствует проверка корректности записываемых значений, то значения из регистровой памяти могут быть сразу использованы в логике приложения, что может привести к сбоям в работе системы управления.

Примеры реализации рекомендации

- если регистры ПЛК должны быть доступны только для чтения – то приложение контроллера или межсетевой экран должны предотвращать запросы записи;
- если часть регистров ПЛК должны быть доступны только для чтения, а часть – для чтения и записи, то следует разделить их на две группы (не смешивая адреса). Это упростит настройку регистров в ПЛК или межсетевом экране.

Обоснование

Причина	Комментарий
Безопасность	Описанный подход упрощает настройку доступа к регистрам в ПЛК и межсетевом экране – сразу становится ясно, какие из них доступны только для чтения. Проверка значений в регистровой памяти на корректность позволяет избежать атак, связанных с их изменением – поскольку сразу станет ясно, что в приложение были переданы некорректные данные, которые не должны использоваться в логике
Надежность	Распределение регистров по группам в соответствии с типом доступа позволяет использовать групповые запросы и ускорить обмен из-за уменьшения числа транзакций. Запись некорректных значений в регистровую память может быть связана не только с атакой, но, например, с повреждением пакетов в процессе передачи – а проверка корректности записанных данных позволяет обнаруживать такие ошибки и не использовать эти значения в логике приложения
Обслуживание	Ошибки, связанные с непреднамеренной записью в регистровую память, сложно обнаружить, поэтому ограничение доступа к регистрам (с помощью выбора для них типа доступа «только чтение») упрощает отладку приложения

Ссылки

Стандарт	Раздел
MITRE ATT&CK for ICS	Tactic: TA009 – Inhibit Response Function, TA010 – Impair Process Control Technique: T0835 – Manipulate I/O image, T0836 – Modify Parameter
ISA 62443-3-3	**SR 3.4**: Software and information integrity **SR 3.5**: Input Validation **SR 3.6**: Deterministic Output
ISA 62443-4-1	**SD-4**: Secure design best practices **SI-1**: Security implementation review **SI-2**: Secure coding standards **SVV-1:** Security requirements testing
ISA 62443-4-2	**CR 3.4**: Software and information integrity **CR 3.5**: Input Validation **CR 3.6**: Deterministic Output
MITRE CWE	**CWE-787:** Out-of-bounds Write **CWE-653:** Insufficient Compartmentalization

11. Проверяйте достоверность измерений

Разрабатывайте систему автоматизации таким образом, чтобы иметь возможность проводить несколько независимых проверок полученных измерений для подтверждения их достоверности.

Компонент, для которого обеспечивается безопасность / надежность	Ответственный
Входы-выходы ПЛК	Разработчик приложения Системный интегратор / сервисный инженер

Рекомендация

Существует несколько способов проверки достоверности измерений:

1. Интегрирование или дифференцирование зависящих от времени параметров и сравнение их с параметрами, не зависящими от времени.

2. Сравнение значений одного параметра, полученных из разных источников (например, от разных датчиков или по разным каналам связи).

Примеры

1.

- насос и индикатор уровня в баке: изменение объема жидкости должно соответствовать расходу;
- горелка котла: дополнительное количество выделенной теплоты должно соответствовать увеличению температуры.

2.

- для контроля набора высоты / снижения самолета используется индикатор воздушной скорости, авиагоризонт, вариометр и высотомер;
- сравнение значений в SCADA-системе, полученных от ПЛК/HMI и из независимых регистраторов (использующих те же самые или независимые дискретные и аналоговые сигналы) по разным каналам связи. В случае расхождения этих значений должны формироваться сигналы тревоги.

Обоснование

Причина	Комментарий
Безопасность	Позволяет отследить попытку «подмены» данных (кроме ситуаций, где происходит подмена сразу всех данных в системе)
Надежность	Позволяет отследить и не использовать в приложении некорректные данные (например, из-за повреждения датчика)
Обслуживание	Повышает скорость устранения неисправностей измерительных приборов

Ссылки

Стандарт	Раздел
MITRE ATT&CK for ICS	Tactic: TA010 - Impair Process Control Technique: T0806 - Brute Force I/O
ISA 62443-3-3	**SR 3.5**: Input Validation **SR 3.6**: Deterministic Output
ISA 62443-4-2	**CR 3.5**: Input Validation **CR 3.6**: Deterministic Output
MITRE CWE	**CWE-754**: Improper Check for Unusual or Exceptional Conditions

12. Проверяйте реалистичность данных, вводимых оператором

Убедитесь, что операторы могут ввести только те значения параметров, которые имеют физический смысл. Запускайте таймеры при начале выполнения операций и формируйте сигналы тревог, если они не успевают выполниться за заданное время. Информируйте оператора о неожиданном бездействии системы.

Компонент, для которого обеспечивается безопасность / надежность	**Ответственный**
Входы-выходы ПЛК	Системный интегратор / сервисный инженер

Рекомендация

a) Контролируйте продолжительность выполняемых операций

Если выполнение операции занимает больше времени, чем ожидалось – формируйте сигнал тревоги. Если операция выполнилась быстрее, чем ожидалось – то это тоже является поводом для формирования сигнала тревоги.

Простым решением является добавление таймаута для выполняемых операций. Такой подход является особенно полезным в системах с пошаговым управлением. Например, ожидаемое время выполнения шага «перемещение объекта из точки A в точку B» занимает 5 секунд (от момента перехода на шаг до момента срабатывания датчика положения). Если выполнение операции занимает гораздо больше или меньше времени – то должен сработать таймер таймаута.

b) Контролируйте бездействие системы

В ряде процессов можно выделить четкие паттерны повторяемых событий и операций. Если какие-то сигналы неожиданно остаются неизменными в течение длительного времени – то следует сформировать соответствующее оповещение для оператора.

Примеры

a) Контролируйте продолжительность выполняемых операций

- Задвижке водоспуска на плотине требуется определенное время для перехода из состояния «полностью закрыто» в состояние «полностью открыто»;
- В системе очистки сточных вод заполнение резервуара происходит за определенное время.

b) Контролируйте бездействие системы

- Производственный процесс состоит из регулярно повторяемого цикла операций (например, упаковка, маркировка и т.д.);
- В системах очистки сточных вод в суточных колебаниях притока сточных вод можно выделить характерные паттерны.

c) Убедитесь, что операторы могут ввести только те значения параметров, которые имеют физический смысл

- Атака на систему управления очистными сооружениями в городе Олдсмар (штат Флорида) оказалась успешной, потому что операторский интерфейс позволил установить концентрацию щёлочи в воде на уровне 11100 ppm (в то время как нормальная концентрация соответствует 100 ppm).

Обоснование

Причина	Комментарий
Безопасность	1. Отклонения в длительности выполняемых операций могут свидетельствовать о поломке исполнительного механизма или атаке, которая ведется путем передачи в систему управления «фальшивых» данных 2. Предупреждения о бездействии системы упрощают обнаружение «замороженных» значений (например, уставок), которые могут быть результатом атаки на систему
Надежность	1. Контроль продолжительности выполняемых операций позволяет своевременно выявить поломку оборудования 2. Контроль бездействия системы позволяет отследить поломку оборудования, ошибки в алгоритме программ или ввод оператором некорректных значений параметров

Ссылки

Стандарт	Раздел
MITRE ATT&CK for ICS	Tactic: TA010 - Impair Process Control Technique: T0806 - Brute Force I/O
ISA 62443-3-3	**SR 3.5**: Input Validation **SR 3.6**: Deterministic Output
ISA 62443-4-2	**CR 3.5**: Input Validation **CR 3.6**: Deterministic Output
MITRE CWE	**CWE-754**: Improper Check for Unusual or Exceptional Conditions

13. Отключите неиспользуемые сетевые порты и протоколы

Контроллеры и коммуникационные модули, как правило, поддерживают несколько сетевых протоколов, которые обычно по умолчанию включены. Отключите сетевые порты и протоколы, которые не используются в данном конкретном приложении.

Компонент, для которого обеспечивается безопасность / надежность	Ответственный
Информационная безопасность	Системный интегратор / сервисный инженер

Рекомендация

Обычно в ПЛК по умолчанию доступны промышленные и прикладные протоколы – Modbus, Profibus, EtherNet/IP, HTTP(S), SNMP, FTP, TELNET, ICMP и т.д.

Лучшее решение – разработать диаграмму потоков данных в системе управления. На этой диаграмме должны быть показаны физические интерфейсы контроллера и сети, к которым они подключены. Для каждого интерфейса должны быть определены необходимые протоколы; неиспользуемые протоколы следует отключить.

Пример

Многие ПЛК имеют встроенный web-сервер для диагностики и настройки. Если его использование не предполагается, то он должен быть отключен, так как в противном случае может стать одной из точек атаки.

Обоснование

Причина	Комментарий
Безопасность	Каждый доступный порт и протокол увеличивают число вариантов для потенциальной атаки. Самый простой способ лишить злоумышленника этой возможности – отключить их
Надежность	Если в ПЛК отключены неиспользуемые протоколы, то это уменьшает получаемый трафик и снижает вероятность получения некорректных (поврежденных или специально сформированных) пакетов, которые могут привести к сбою (например, из-за переполнения буфера)
Обслуживание	Отключение неиспользуемых портов и протоколов упрощает обслуживание ПЛК, так как в этом случае на их настройки можно не обращать внимания

Ссылки

Стандарт	Раздел
MITRE ATT&CK for ICS	**Tactic**: TA005 - Discovery **Technique**: T0808 - Control Device Identification, T0841 - Network Service Scanning, T0854 - Serial Connection Enumeration
ISA 62443-3-3	**SR 7.6**: Network and security configuration settings **SR 7.7**: Least functionality
ISA 62443-4-2	**EDR 2.13**: Use of physical diagnostic and test interfaces

ISA 62443-4-1	**SD-4**: Secure design best practices **SI-1**: Security implementation review **SVV-1**: Security requirements testing

14. Ограничьте обмен данными с другими устройствами

Ограничьте тип подключений и наборы параметров, которые доступны для сторонних устройств. Типы подключений для каждого интерфейса должны быть четко определены, чтобы только необходимые в рамках конкретной задачи параметры были доступны для чтения и записи.

Компонент, для которого обеспечивается безопасность / надежность	Ответственный
Информационная безопасность	Системный интегратор / сервисный инженер

Рекомендация

В некоторых случаях объединение устройств в локальную сеть является более предпочтительным, чем соединение их напрямую отдельными кабелями связи (например, в случае передачи больших объемов данных или ограничений, связанных с прокладкой таких кабелей).

При проектировании сети и настройке интерфейсов ПЛК следует соблюдать приведенные ниже рекомендации:

- используйте специальный коммуникационный модуль для вашего ПЛК или промежуточное сетевое оборудование для подключения контроллера к локальной сети;
- большинство устройств с функциями кибербезопасности позволяет считать MAC-адрес устройства с помощью системной переменной. Это позволяет верифицировать устройства по IP- и MAC-адресам. Безусловно, этот подход не гарантирует подлинности устройства, потому что эти адреса могут быть подделаны, но, тем не менее, повышает уровень безопасности системы управления;
- при выборе коммуникационного протокола следует предпочесть тот, который ограничивает возможность записи другими устройствами данных в ПЛК (за исключением записи необходимых параметров);
- выбранный для обмена интерфейс и порт ПЛК не должны позволять конфигурировать и программировать ПЛК;
- другие устройства не должны иметь возможность считывать или записать параметры ПЛК, для которых в явном виде не указана эта возможность;
- используйте сторожевой таймер для диагностики связи, чтобы не отправлять команды в ПЛК, который перешел в состояние отказа;
- для последовательных интерфейсов используйте специальные коммуникационные модули с ограниченным набором данных. Используйте ПЛК в режиме master;
- протокол EtherNet/IP позволяет коммуникационным модулям работать в режиме межсетевого экрана и выполнить функции DPI (deep packet inspection), а также принимать запросы только от разрешенных IP-адресов. Если ваш модуль поддерживает эти функции, то они должны быть включены и настроены;
- если по каким-то причинам выполнить приведенные выше рекомендации не удается, то следует рассмотреть возможность добавления в систему отдельного ПЛК в качестве шлюза между полевым уровнем и внешней сетью. Для такого ПЛК (и

подключенных к нему устройств) не должно быть возможности конфигурирования и программирования из внешней сети.

Примеры

- автоматизированные групповые замерные установки используются как добывающей компанией, так и компанией, которая занимается транспортировкой нефти, при этом в рамках установки должна быть организована передача данных между системами, которые принадлежат разным компаниям;
- поставщик питьевой воды предоставляет услуги водозаборным станциям местного муниципалитета. Системы управления станциями и системы поставщика должны быть интегрированы.

Обоснование

Причина	Комментарий
Безопасность	1. Ограничение доступа к сторонним сетям и оборудованию 2. Аутентификация других устройств для подтверждения их подлинности
Надежность	Ограничение непреднамеренного или преднамеренного доступа к системе управления

Ссылки

Стандарт	Раздел
MITRE ATT&CK for ICS	**Tactic**: TA010 - Impair Process Control **Technique**: T0836 - Modify Parameter
ISA 62443-3-3	**SR 7.6:** Network and security configuration settings **SR 7.7:** Least functionality
ISA 62443-4-2	**CR 7.6:** Network and security configuration settings **CR 7.7:** Least functionality
ISA 62443-4-1	**SD-4:** Secure design best practices **SI-1:** Security implementation review **SVV-1:** Security requirements testing

15. Определите безопасное состояние выходов при перезапуске ПЛК

Определите безопасное состояние выходов при перезапуске ПЛК (например, может потребоваться их включить, выключить или сохранить в текущем состоянии).

Компонент, для которого обеспечивается безопасность / надежность	Ответственный
Отказоустойчивость	Разработчик приложения Системный интегратор / сервисный инженер

Рекомендация

Если по какой-либо причине ПЛК перезагружается, то это не должно приводить к существенным сбоям в технологическом процессе. Предварительно следует убедиться, что контролируемый процесс подразумевает возможность перезапуска ПЛК.

Если после перезагрузки ПЛК нецелесообразно пытаться продолжать управление процессом, то ПЛК должен проинформировать оператора о факте перезагрузки и перейти в состояние, в котором не происходит управление исполнительными механизмами и другим оборудованием. В инструкциях для обслуживающего персонала должен быть четко описан порядок действий для возобновления технологического процесса в подобной ситуации.

Кроме того, в инструкциях должны быть описаны процедуры запуска, остановки и перезапуска системы управления.

Обоснование

Причина	Комментарий
Безопасность	Устранение непредвиденных ситуаций: самый простой способ атаки на ПЛК – заставить его аварийно остановить работу и/или перезагрузиться. Для многих ПЛК это не так сложно сделать, потому что они не могут обработать некорректные данные или, например, слишком интенсивный поток сетевых запросов. Хотя какие-то средства диагностики почти всегда доступны, обычно неясно, что произойдет, если ПЛК перезагрузится во время управления технологическим процессом. Это не является штатной ситуацией, но это один из основных вариантов атак на ПЛК
Надежность	Устранение непредвиденных задержек: после включения ПЛК может перейти на начальный шаг машины состояний. Для перехода к следующим шагам может потребоваться выполнение определенных внутренних условий, на которые оператор повлиять не в состоянии. В этом случае инженеру придется подключиться к контроллеру и вручную изменять состояния переменных. Это приведет к задержкам и потерям прибыли из-за недопроизведенной продукции

Ссылки

Стандарт	Раздел
MITRE ATT&CK for ICS	**Tactic:** TA009 - Inhibit Response Function **Technique:** T0816 - Device Restart/Shutdown
ISA 62443-3-3	**SR 3.6:** Deterministic Output
ISA 62443-4-2	**CR 3.6:** Deterministic Output
ISA 62443-4-1	**SVV-1:** Security requirements testing

16. Измеряйте время цикла ПЛК и отображайте его на HMI в виде трендов

Измеряйте время цикла ПЛК каждые 2-3 секунды и отображайте его на HMI в виде тренда.

Компонент, для которого обеспечивается безопасность / надежность	Ответственный
Мониторинг	Системный интегратор / сервисный инженер

Рекомендация

Время цикла обычно доступно в ПЛК в виде системной переменной. Необходимо периодически считывать время цикла ПЛК, чтобы иметь возможность рассчитать его минимальное, максимальное и среднее значение. Эти значения должны отображаться на HMI в виде тренда, чтобы предупреждать оператора об отклонениях в работе ПЛК.

Время цикла – это время, за которое происходит однократное выполнение пользовательского кода (программ на языках стандарта МЭК 61131-3: LD, FBD, IL, ST, SFC).

Время цикла может быть приблизительно постоянным, если не происходит никаких изменений в:

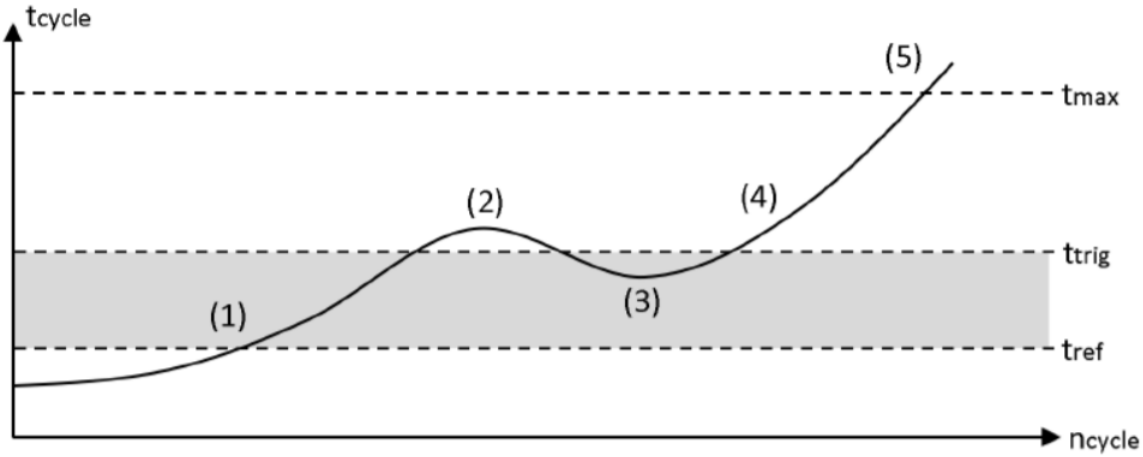
- работе сети (например, резкого увеличения трафика);
- работе приложении ПЛК (например, перехода в другое состояние, в котором выполняется другой набор операций);
- процессе управления.

Следовательно, отклонения значений времени цикла ПЛК от средних значений могут быть признаком преднамеренного изменения приложения ПЛК или нарушения его целостности. Визуализация истории этих значений в виде тренда на HMI позволяет привлечь внимание оператора к этому факту (что было бы гораздо сложнее сделать путем отображения только текущего значения времени цикла).

Пример

Многие ПЛК имеют аппаратный сторожевой таймер, который срабатывает при превышении временем цикла максимального допустимого значения. В результате ПЛК переключается в состояние STOP (5[1]). Естественно, злоумышленники об этом знают – и поэтому стараются минимизировать влияние вредоносного кода на время цикла ПЛК. Поэтому требуется организовать в ПЛК мониторинг времени цикла. Поскольку небольшие отклонения в его значениях являются естественными, то следует определить допустимые пределы (1, 3), при выходе за которые должно формироваться предупреждение для оператора (2, 4).

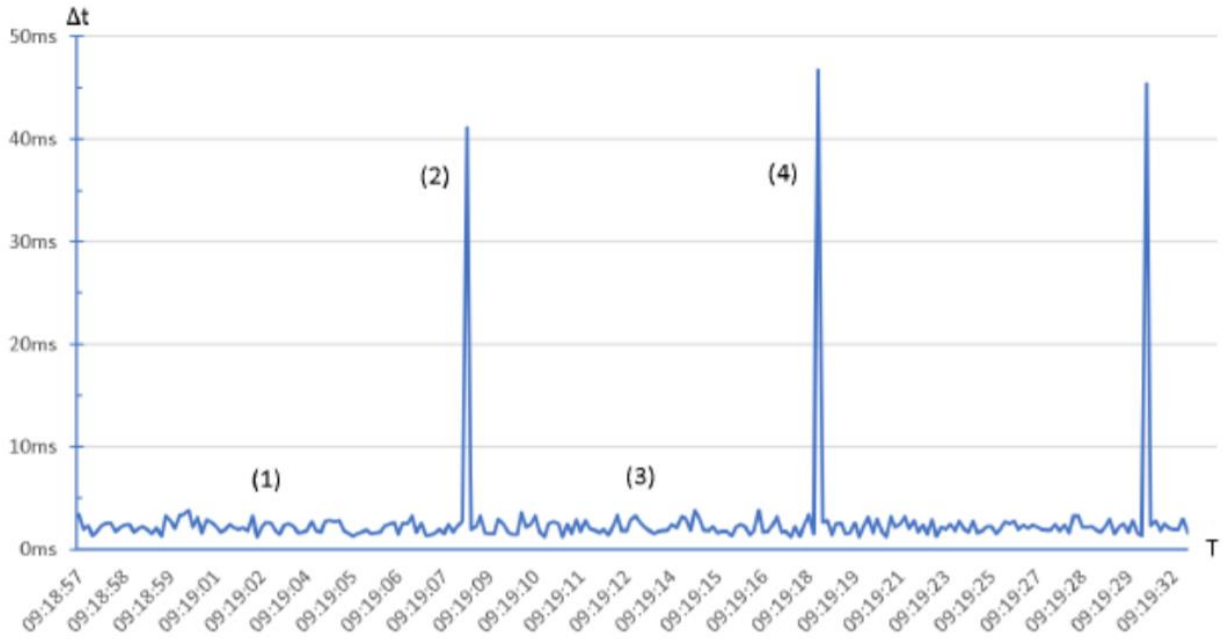
[1] См. отметки на рисунке на следующей странице



Любые отклонения от базового (заранее полученного в ходе пусконаладки) времени цикла должны сохраняться в журнале сообщений с меткой времени:

SeqNo	Date	UTC Time	Abweichung
1	2019-11-22	09:05:50.021	40,821ms
2	2019-11-22	09:06:00.069	44,391ms
3	2019-11-22	09:06:10.120	44,994ms
4	2019-11-22	09:06:20.166	40,561ms
5	2019-11-22	09:06:30.211	40,725ms

Если время цикла отображается на HMI в виде тренда, то любые отклонения в работе приложения будут сразу заметны. На рисунке ниже приведен пример программы с периодически выполняемым вредоносным кодом. В фазах 1 и 3 отклонения являются допустимыми, а пики 2 и 4 появляются в моменты выполнения вредоносного кода.



Обоснование

Причина	Комментарий
Безопасность	Атаки на ПЛК могут включать изменение его приложения, загрузку другой программы, загрузку новых рецептов, добавление конкретных фрагментов кода для отправки сообщений или выполнения других функций. Поскольку большинство ПЛК не поддерживают традиционные криптографические функции – то отследить такие события явным образом не получится. Но неявно их можно определить по отклонениям во времени цикла ПЛК – это хороший индикатор того, что приложение контроллера могло быть заменено
Надежность	Все описанное выше также справедливо не только в контексте преднамеренных атак, но и случайных изменений (например, случайная загрузка программы с ошибкой или некорректным рецептом)

Ссылки

Стандарт	Раздел
MITRE ATT&CK for ICS	**Tactic:** TA002 - Execution **Technique:** T0873 - Project File Infection
ISA 62443-3-3	**SR 3.4:** Software and information integrity
ISA 62443-4-2	**EDR 3.2:** Protection from malicious code
MITRE CWE	**CWE-754:** Improper Check for Unusual or Exceptional Conditions

17. Регистрируйте время непрерывной работы ПЛК и отображайте его на HMI

Регистрируйте время непрерывной работы ПЛК, чтобы определить моменты, когда он перезапускался. Отображайте это время на HMI для диагностики.

Компонент, для которого обеспечивается безопасность / надежность	Ответственный
Мониторинг	Системный интегратор / сервисный инженер

Рекомендация

Регистрируйте время непрерывной работы ПЛК. Обычно это можно сделать:

- с помощью системной переменной;
- с помощью MIB-объекта (если ПЛК поддерживает протокол SNMP в режиме Agent).

Если ПЛК поддерживает протокол SNMP в режиме Agent – то для получения его времени непрерывной работы можно использовать объект **sysUpTimeInstance(0)**, который имеет OID **1.3.6.1.2.1.1.3.** Сброс счетчика непрерывной работы свидетельствует о перезапуске ПЛК. На HMI в этом случае должно быть отображено соответствующее сообщение.

Контроль времени непрерывной работы для генерации аварийных сообщений на HMI при перезапуске ПЛК является хорошим способом диагностики ошибок.

Обоснование

Причина	Комментарий
Безопасность	Устранение непредвиденных ситуаций: самый простой способ атаки на ПЛК – заставить его аварийно остановить работу и/или перезагрузиться. Для многих ПЛК это не так сложно сделать, потому что они не могут обработать некорректные данные или, например, слишком интенсивный поток сетевых запросов. Поэтому перезагрузка ПЛК может являться признаком атаки
Надежность	Регистрация перезагрузок ПЛК является хорошим способом диагностики сбоев в работе приложения, а также позволяет определить, что с ПЛК проводятся сервисные работы

Ссылки

Стандарт	Раздел
MITRE ATT&CK for ICS	**Tactic:** TA009 - Inhibit Response Function **Technique:** T0816 - Device Restart/Shutdown
ISA 62443-3-3	**SR 7.6:** Network and security configuration settings
ISA 62443-4-2	**CR 7.6:** Network and security configuration settings
MITRE CWE	**CWE-778:** Insufficient Logging

18. Организуйте логирование остановок работы ПЛК по исключению и отображайте информацию об этом на HMI

Организуйте логирование остановок работы ПЛК по исключению и отображайте на HMI соответствующие аварийные сообщения, чтобы помочь оператору понять, какие действия необходимо совершить перед перезапуском ПЛК. Настройте в ПЛК синхронизацию времени для отображения точных меток времени этих событий.

Компонент, для которого обеспечивается безопасность / надежность	Ответственный
Мониторинг	Системный интегратор / сервисный инженер

Рекомендация

Остановка ПЛК указывает на наличие проблемы, которую необходимо решить до перезапуска контроллера.

Некоторые ПЛК позволяют считать код ошибки, которая привела к останову работы приложения. Сохраните эти коды и предоставьте оператору четкое описание причин, по которым могла возникнуть данная ошибка. Хорошим подходом является отображение этих ошибок на HMI или, например, на syslog-сервере (если он присутствует в системе автоматизации).

Многие ПЛК имеют функцию «первого цикла» (first scan) или позволяют ее организовать с помощью пользовательского кода. Эта функция позволяет однократно выполнить при старте приложения определенные действия, а также является индикатором запуска приложения. Информация о выполнении «первого цикла» должна регистрироваться и отображаться на HMI.

Обоснование

Причина	Комментарий
Безопасность	Журналы ошибок позволяют устранить неполадки в случае инцидента. Прежде чем повторно запустить ПЛК в работу – следует убедиться, что причины неисправности были устранены
Надежность	Журналы ошибок упрощают отладку приложения (потому что ошибки могут быть вызваны не только действиями злоумышленника, но и недоработками приложения)

Ссылки

Стандарт	Раздел
MITRE ATT&CK for ICS	**Tactic:** TA009 - Inhibit Response Function **Technique:** T0816 - Device Restart/Shutdown
ISA 62443-3-3	**SR 7.6:** Network and security configuration settings
ISA 62443-4-2	**CR 7.6:** Network and security configuration settings
MITRE CWE	**CWE-778:** Insufficient Logging

19. Отслеживайте потребление памяти ПЛК и отображайте его на HMI

Отслеживайте потребление памяти ПЛК и отображайте его на HMI в виде тренда. Также отображайте усредненный ожидаемый уровень потребления памяти (который должен быть измерен заранее) в качестве точки отсчета для анализа.

Компонент, для которого обеспечивается безопасность / надежность	Ответственный
Мониторинг	Системный интегратор / сервисный инженер

Рекомендация

Поскольку увеличение числа строк кода приложения может привести к увеличению потребляемой памяти, разработчик должен предусмотреть отслеживание потребления памяти и генерацию предупредительных сообщений в тех случаях, если оно превышает допустимые пределы.

Пример

Для ПЛК Rockwell Allen Bradley можно установить допустимый уровень потребления памяти в настройках контроллера, а отслеживать текущий уровень потребления можно с помощью модуля RSLogix 5000 Task Monitor Tool. Можно отслеживать не только общую память, но и память области ввода-вывода, а также память программы и тегов.

Обоснование

Причина	Комментарий
Безопасность	Повышенное потребление памяти может быть индикатором предумышленного изменения приложения ПЛК
Надежность	Отслеживание потребления памяти может быть полезно для обнаружения утечек памяти и предотвращения перехода ПЛК в состояние отказа
Обслуживание	Отслеживание потребления памяти может быть полезно для подбора параметров ПЛК (например, времени цикла) и диагностики неисправностей

Ссылки

Стандарт	Раздел
MITRE ATT&CK for ICS	**Tactic:** TA002 - Execution **Technique**: T0873 - Project File Infection
ISA 62443-3-3	**SR 3.4**: Software and information integrity
ISA 62443-4-2	**EDR 3.2**: Protection from malicious code

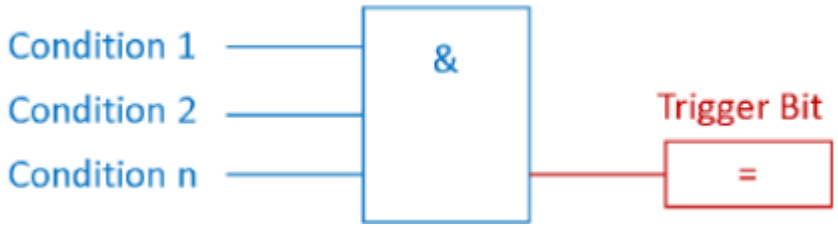
20. Отслеживайте несрабатывания и ложные срабатывания для сигналов критических тревог

Определите условия возникновения критических тревог и отслеживайте их возникновение. Также отслеживайте условия срабатывания и состояния битов тревог для любого отклонения в производственном процессе.

Компонент, для которого обеспечивается безопасность / надежность	Ответственный
Мониторинг	Системный интегратор / сервисный инженер

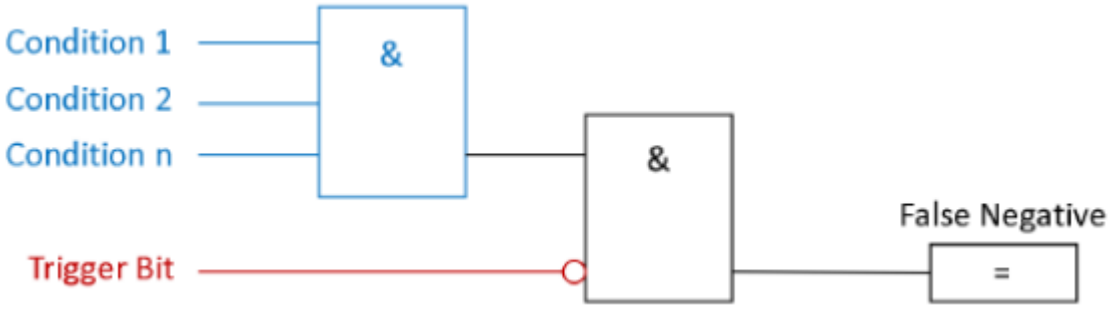
Рекомендация

В большинстве случаев сигналы тревог представляют собой переменные логического типа (boolean), которые формируются на основании определенного набора условий. Например, сигнал тревоги «избыточное давление» формируется в том случае, если биты «реле давления 1», «значение датчика давления выше критического порога» и т.д. имеют значение TRUE.

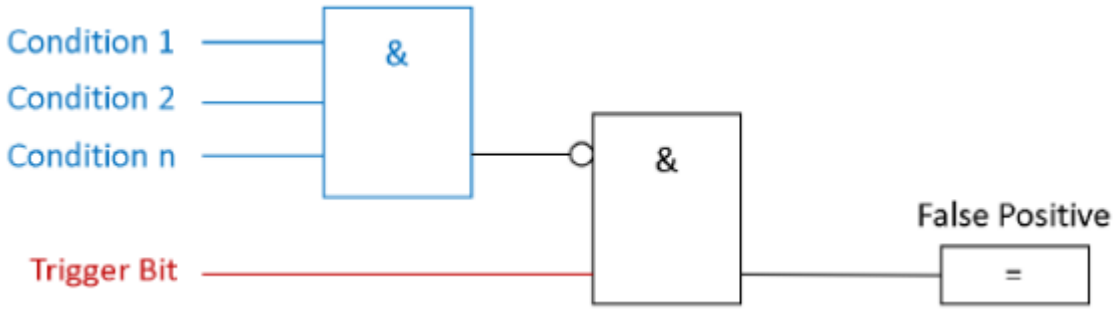


Для маскировки кибератаки злоумышленник может перезаписать бит тревоги (trigger bit), чтобы оператор не получил информацию о наличии аварийной ситуации в системе управления.

Детектировать такую ситуацию можно с помощью добавления к списку условий инвертированного значения самого сигнала тревог. Очевидно, что если условия тревоги имеют значение TRUE, а сигнал тревоги при этом имеет значение FALSE – то произошло несрабатывание тревоги (false negative).



В других ситуациях злоумышленник специально можно сгенерировать ложные сигналы тревоги, чтобы отвлечь внимание оператора. Детектирование таких случаев осуществляется аналогичным образом: если условия тревоги имеют значение FALSE, а сигнал тревоги при этом имеет значение TRUE – то произошло ложное срабатывание тревоги (false positive).



Примеры

1. ПЛК Siemens Simatic S7-1200/1500 имеют встроенный web-сервер с широким набором функций для диагностики и мониторинга. В том числе, имеется возможность просмотра и изменения значений переменных ПЛК. Права доступа настраиваются на вкладке PLC – Hardware Settings. В случае некорректной настройки прав доступа злоумышленник может получить возможность изменять значения переменных ПЛК – в том числе, битов тревог.

2. В кибератаках с использованием вируса Triton / Trisys / HatMan злоумышленники подавляли формирование сигналов тревог.

3. Кибератака на промышленную сеть может сопровождаться генерацией ложных срабатываний сигналов тревог в SCADA для отвлечения внимания оператора.

Обоснование

Причина	Комментарий
Безопасность	Детектирование несрабатываний и ложных срабатываний сигналов тревог позволяет своевременно определить начало кибератаки

Ссылки

Стандарт	Раздел
MITRE ATT&CK for ICS	**Tactic**: TA009 - Inhibit Response Function **Technique**: T0878 - Alarm Suppression
ISA 62443-3-3	**SR 3.5**: Input Validation
ISA 62443-4-2	**CR 3.5**: Input Validation
ISA 62443-4-1	**SI-1**: Security implementation review
MITRE CWE	**CWE-754**: Improper Check for Unusual or Exceptional Conditions

О проекте «Безопасное программирование ПЛК»

В течение длительного времени разработчики ПЛК не уделяли внимания вопросам безопасности. В последние годы адаптация практик из области IT в АСУ ТП привела к появлению безопасных протоколов, зашифрованных каналов связи, внедрению практики разделения полевой сети и сети предприятия и т.д. Но, тем не менее, до сих пор никто не задавался вопросом о том, как использовать типовой функционал ПЛК (или SCADA/РСУ) для обеспечения безопасности – иными словами, как программировать ПЛК с расчетом на устойчивость к воздействию кибератак. Данный проект, вдохновленный практиками безопасного программирования, призван заполнить этот пробел.

Для кого написано это руководство?

Данный документ предназначен для инженеров. Целью нашего проекта является предоставление рекомендаций для специалистов, которые разрабатывают приложения для систем автоматизации (на языке LD, FBD и т.д.), для повышения безопасности систем управления. Эти рекомендации основаны на использовании стандартного функционала, доступного в большинстве ПЛК/SCADA; для их использования не требуется дополнительного оборудования или ПО. Все они могут быть включены в стандартизированный процесс разработки приложений. Для применения этих рекомендаций необходимы не только опыт в информационной безопасности, но и хорошее знание применяемых ПЛК, их логики и особенностей автоматизируемого процесса управления.

Что рассматривается в рамках этого руководства?

Рекомендации данного документа описывают изменения, которые должны быть внесены в приложение для повышения уровня его безопасности. Документ содержит 20 наиболее важных рекомендаций. Существуют также рекомендации по архитектуре системы управления, HMI и документации. Они не относятся к безопасному программированию, но могут быть включены в состав будущего документа, посвященного обеспечению безопасности систем управления в целом.

Каковы преимущества использования данных рекомендаций?

Очевидно, что использование этих рекомендаций повышает уровень безопасности ПЛК – в основном либо за счет уменьшения поверхности атаки, либо за счет уменьшения времени на устранение неполадок в случае нарушения безопасности. Однако этим дело не ограничивается – следование рекомендациям также позволяет повысить надежность приложения, упростить его отладку и поддержку, сделать его более компактным и облегчить настройку обмена с ПЛК. Кроме того, рекомендации позволяют не только избежать или уменьшить последствия предумышленной атаки, но и сделать приложение ПЛК более устойчивым к ошибкам, связанным с человеческим фактором (например, вводу некорректных данных).

Кто участвует в проекте?

Все началось с выступления Джейка Бродского на конференции S4x20, которое называлось «Практики безопасного программирования для ПЛК».

После конференции Дейл Петерсон начал проект, который привел к появлению этого документа. Джейк Бродский и Сара Флючс потратили несколько часов на телефонный разговор, в

ходе которого зафиксировали предложенные Джейком рекомендации по безопасному программированию ПЛК. После этого Дейл, Джейк и Сара при поддержке организации ISA GCA создали платформу top20.isa.org, чтобы собрать и структурировать дополнительную информацию от специалистов по информационной безопасности и инженеров АСУ ТП.

Обобщение и обсуждение рекомендаций, а также составление списка 20 наиболее важных их них, заняло около года; процесс был ускорен Вивеком Поннада, который не только внес свой вклад в содержимое документа, но и постоянно организовывал созвоны участников для устранения всех возникших замечаний. Мохамед Абдельмез Сакесли добавил в документ все ссылки на стандарты. Команда MITRE CWE предоставляла ссылки из MITRE ATT&CK for ICS вплоть до релиза документа. Сара составила документ, который вы сейчас читаете, а Джейк, Дейл, Джон Кузимано, Дирк Ротермунд, Джош Рафф, Томас Рабенштейн, Гас Серино, Уолтер Спет, Августин Валенсия Хиль-Ортега, Марсель Рик-Сен и Аль Ратиш Р. внесли вклад в его содержимое.

Кто участвует в проекте?

Проект «Безопасное программирование ПЛК» появился и существует за счет усилий сообщества, вложившего свои силы и знания в данный документ. В общей сложности на онлайн-платформе зарегистрировалось 943 пользователя. Ниже приведен список тех, кто согласился быть упомянутым. Спасибо всем, кто нашел время поддержать этот проект!

Aagam Shah	Josie Houghton
Adam Paturej	Jozef Sulwinsk
Agustin Valencia Gil-Ortega	Juan Pablo Angel Espejo
Aitor García Almiñana	Khalid Ansari
Alec Summers	Marc Weber
Al Ratheesh. R	Marcel Rick-Cen
Andreas Falk	Martin Huddleston
Anton Shipulin	Massimiliano Zonta
Arkaitz Gamino	Matthew Loong
Carlos Olave	Matthias Müller
Chris van den Hooven	Michael Thompson
Chris Sistrunk	Michal Stepien
Christos Alexopoulos	Miguel Angel Frias
Cris DeWitt	Mohamed Abdelmoez Sakesli
Dale Peterson	Moon Eluvangal Chandran
Dene Yandle	Nahuel Iglesias
Dennis Verschoor	Nalini Kanth
Dirk Rotermund	Narasimha S. Himakuntala
Edorta Echave García	Omar Morando
Gananand Kini	Oscar J. Delgado-Melo
George Alex Holburn	Päivi Brunou
Gus Serino	Peter Donnelly
Hakija Agic	Peter Jackson
Hector Medrano	Ravindra Deshakulakarni
Heiko Rudolph	Rick Booij
Isiah Jones	Robert Albach
Jacob Brodsky	Rushi Purohit
Javier Perez Quezada	Sarah Fluchs
J-D Bamford	Sergei Biberdorf
Joe Weiss	Stephan Beirer
John Cusimano	Steve Christey Coley
John Hoyt	Thomas Rabenstein
John Powell	Tim Gale
John Kingsley	Vivek Ponnada
Joseph J. Januszewski	Vytautas Butrimas
Josh Ruff	Walter Speth

Мы приносим особую благодарность организациям, которые помогли нам с инфраструктурой для нашего проекта – доменом, хостингом, графическим дизайном и web-дизайном: