МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ВЛАДИВОСТОКСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНСТИТУТ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И АНАЛИЗА ДАННЫХ
КАФЕДРА ИНФОРМАЦИОННОЙ БЕЗОПАСНОСТИ

КУРСОВАЯ ПРОЕКТИРОВАНИЕ 1

Обеспечение информационной безопасности при разработке ПО для манипулятора

Студент _____ Е.А. Чаркин

Руководитель

канд. экон. наук, доцент _____ Е.Г. Шумик

Нормоконтролер

канд. экон. наук, доцент _____ Е.Г. Шумик
(нормоконтролер руководитель)

Владивосток 2026

Аннотация

Курсовая работа на тему: «Обеспечение информационной безопасности при разработке ПО для промышленного манипулятора».

Автор: студент гр. ИБ-22-1, Чаркин Е.А.

Руководитель: кандидат экономических наук, доцент Е.Г. Шумик.

Работа посвящена анализу требований информационной безопасности к программному обеспечению промышленных манипуляторов на значимых объектах критической информационной инфраструктуры, а также практическому применению ключевых мер защиты на экспериментальном образце.

Ключевыми аспектами работы являются анализ нормативной базы (федеральное законодательство о безопасности КИИ, требования ведомства ФСТЭК, национальные стандарты ГОСТ Р ИСО / МЭК 27002-2021 и ГОСТ Р 56939-2016), выявление основных классов уязвимостей для промышленных манипуляторов, а также изучение отечественной и международной практики обеспечения безопасности промышленных роботов на примере автомобильных заводов (АвтоВАЗ, УАЗ, VAG, Тойота).

В практической части работы был разработан и изготовлен макет промышленного манипулятора из ABS-пластика с использованием 3D-печати на базе микроконтроллера Arduino Nano с двумя шаговыми двигателями, сервоприводом, потенциометрами и модулем Wi-Fi ESP8266. Программное обеспечение разработано на C++ в среде Ардуино IDE, реализуя локальное и удаленное управление. Реализованы меры информационной безопасности: таймер, аварийная остановка, белый список команд, проверка входных данных, ограничение буфера приема и скорость обработки команд. Были выявлены ограничения платформы Ардуино для реализации контроля целостности прошивки и предложены способы их устранения.

Предполагается, что разработанная экспериментальная установка и реализованные защитные механизмы могут быть использованы при проектировании безопасных систем управления промышленными манипуляторами с учетом категории важности объекта критической информационной инфраструктуры.

Работа содержит 35 страниц, включает 15 источников и 6 рисунков.

Содержание

Введение

Современное промышленное производство отличается высоким уровнем роботизации технологических процессов. Промышленные манипуляторы, интегрированные в автоматизированные системы управления, являются обязательной частью любого крупного предприятия. Их устойчивое функционирование имеет важнейшее значение для оборонной, металлургической, химической промышленности, транспорта и энергетики.

Рост числа компьютерных атак на объекты КИИ создаёт серьёзные угрозы безопасности. Программное обеспечение промышленных манипуляторов и других станков с ЧПУ, становится одной из обилия уязвимых целей, для злоумышленников.

Несанкционированное изменение прошивки или подмена управляющих команд могут иметь катастрофические последствия.

Действующая нормативно-правовая база - федеральный закон о безопасности КИИ, ведомственные требования регулятора, национальные стандарты по информационной безопасности и безопасной разработке ПО - устанавливает обязательные требования к защите информации. Однако практическая реализация этих требований осложняется, особенно на ограниченных по ресурсам платформах.

Объект исследования - программное обеспечение управления промышленным манипулятором как часть АСУ ТП.

Предмет исследования - требования информационной безопасности к разработке и эксплуатации ПО промышленных манипуляторов, а также способ его реализации.

Цель работы - анализ требований к безопасности ПО промышленных манипуляторов, изучение отечественной и мировой практики, а также практическая реализация ключевых мер защиты на, собственноручно созданной, экспериментальной базе.

Практическая значимость, заключается, в создании действующего образца манипулятора с реализованными механиками защиты, который может служить платформой, для отработки мер по обеспечению ИБ на предприятии.

Работа состоит из введения, двух глав и заключения. В первой главе проведён анализ нормативной базы, требований и мировой практики. Во второй главе описана разработка прототипа, для тестов, и внедрение мер защиты.

1 Анализ требований, уязвимостей и практики эксплуатации ПО промышленных манипуляторов

1.1 Анализ требований в нормативно правовых актах

Функционирование манипуляторов на производстве требует соблюдения норм и правил в информационной безопасности. Для того что бы обозначить эти нормы и правила, необходимо установить перечень правовых документов. Правовой основой можно считать ФСТЭК России от 25 декабря 2017 г. № 239 «Об утверждении Требований по обеспечению безопасности значимых объектов критической информационной инфраструктуры Российской Федерации», промышленный манипулятор, интегрированный в автоматизированную систему управления (АСУ ТП) значимого объекта критической информационной инфраструктуры, подпадает под действие установленных требований, к защите: «В значимых объектах объектами, подлежащими защите от угроз безопасности информации (объектами защиты), являются: в) в автоматизированных системах управления: информация (данные) о параметрах (состоянии) управляемого (контролируемого) объекта или процесса (в том числе входная (выходная) информация, управляющая (командная) информация, контрольно-измерительная информация, иная критически важная (технологическая) информация); программно-аппаратные средства (в том числе автоматизированные рабочие места, промышленные серверы, телекоммуникационное оборудование, линии связи, программируемые логические контроллеры, производственное, технологическое оборудование (исполнительные устройства); программные средства (в том числе микропрограммное, общесистемное, прикладное программное обеспечение); средства защиты информации; архитектура и конфигурация автоматизированной системы управления.» [1, п. 17]

Перечень мер приведён в п. 22 Требований и детализирован в Приложении к ним, в том же нормативном документе: «В значимых объектах в зависимости от их категории значимости и угроз безопасности информации посредством исполнения организационных и технических мер, определяемых в соответствии с приложением к настоящим Требованиям, осуществляются: а) идентификация и аутентификация субъектов доступа и объектов доступа; б) управление доступом субъектов доступа к объектам доступа; в) ограничение программной среды; г) защита машинных носителей информации; д) аудит безопасности; е) антивирусная защита; ж) предотвращение вторжений (компьютерных атак); з) обеспечение целостности; и) обеспечение доступности; к) защита технических средств и систем; л) защита информационной (автоматизированной) системы и ее компонентов; м) планирование мероприятий по обеспечению безопасности; н) управление конфигурацией; о) управление обновлениями программного обеспечения; п) реагирование на инциденты информационной

безопасности; р) обеспечение действий в нештатных ситуациях; с) информирование и обучение персонала; т) защита значимых объектов от атак, направленных на отказ в обслуживании.» [1, п. 22].

Таким образом, разрабатываемое программное обеспечение для управления промышленным манипулятором, в соответствии с рассматриваемым ФСЭК № 239, относится к категории «прикладное программное обеспечение», а сам манипулятор - к категории «исполнительные устройства», что налагает на процесс разработки и эксплуатации совокупность обязательных к исполнению организационных и технических мер.

Соответственно, для разрабатываемого ПО промышленного манипулятора ключевое значение имеют меры по обеспечению целостности программного обеспечения, контролю доступа, регистрации событий безопасности, антивирусной защите, а также требования к безопасности разработки самого программного обеспечения.

Не только в ведомственных требованиях ФСТЭК указываются, базовые правовые основы обеспечения безопасности системы управления промышленным манипулятором, а также и в Федеральном законе от 26 июля 2017 г. № 187-ФЗ «О безопасности критической информационной инфраструктуры Российской Федерации».

Согласно ст. 2 пункту 1 данного закона: «автоматизированная система управления - комплекс программных и программно-аппаратных средств, предназначенных для контроля за технологическим и (или) производственным оборудованием (исполнительными устройствами) и производимыми ими процессами, а также для управления такими оборудованием и процессами» [2, ст. 2, п. 1]. Это положение соответствует выводу о том, что промышленный манипулятор выступает в роли исполнительного устройства, а разрабатываемое ПО - частью АСУ.

Далее, ст. 2 закона определяет компьютерную атаку: «компьютерная атака - целенаправленное воздействие программных и (или) программно-аппаратных средств на объекты критической информационной инфраструктуры, сети электросвязи, используемые для организации взаимодействия таких объектов, в целях нарушения и (или) прекращения их функционирования и (или) создания угрозы безопасности обрабатываемой такими объектами информации» [2, ст. 2, п. 4]. Следовательно, угрозы нарушения целостности и доступности для такого ПО относятся к категории наиболее критичных.

В соответствии со ст. 10 этого же закона: «В целях обеспечения безопасности значимого объекта критической информационной инфраструктуры субъект критической информационной инфраструктуры в соответствии с требованиями к созданию систем безопасности таких объектов и обеспечению их функционирования, утвержденными федеральным органом исполнительной власти, уполномоченным в области обеспечения

безопасности критической информационной инфраструктуры Российской Федерации, создает систему безопасности такого объекта и обеспечивает ее функционирование.» [1, ст. 10 п. 1], а задачи для этой системы в свою очередь указаны ниже в этом же документе: «Основными задачами системы безопасности значимого объекта критической информационной инфраструктуры являются: 1) предотвращение неправомерного доступа к информации, обрабатываемой значимым объектом критической информационной инфраструктуры, уничтожения такой информации, ее модифицирования, блокирования, копирования, предоставления и распространения, а также иных неправомерных действий в отношении такой информации; 2) недопущение воздействия на технические средства обработки информации, в результате которого может быть нарушено и (или) прекращено функционирование значимого объекта критической информационной инфраструктуры; 3) восстановление функционирования значимого объекта критической информационной инфраструктуры, обеспечиваемого в том числе за счет создания и хранения резервных копий необходимой для этого информации; 4) непрерывное взаимодействие с государственной системой обнаружения, предупреждения и ликвидации последствий компьютерных атак на информационные ресурсы Российской Федерации.» [2, ст. 10, п. 2].

Статья 11 закона, в свою очередь устанавливает: «Требования по обеспечению безопасности значимых объектов критической информационной инфраструктуры, устанавливаемые федеральным органом исполнительной власти, уполномоченным в области обеспечения безопасности критической информационной инфраструктуры Российской Федерации, дифференцируются в зависимости от категории значимости объектов критической информационной инфраструктуры и этими требованиями предусматриваются» [2, ст. 11, п. 1]. Применительно к разрабатываемому программному обеспечению манипулятора, это означает обязанность внедрения, на этапе разработки, таких механизмов, как: контроль целостности исполняемых модулей при загрузке и в рантайме; разграничение доступа к интерфейсам программирования и настройкам; аудит всех команд управления, особенно аварийных и критических перемещений.

Таким образом, требования Федерального закона № 187-ФЗ формируют верхнеуровневый правовой контекст, а Приказ ФСТЭК № 239 и Приложения к нему задают конкретные меры защиты, которые должны быть реализованы в разрабатываемом ПО промышленного манипулятора.

Но для практической реализации мер ИБ целесообразно огласить и положения ГОСТ Р ИСО/МЭК 27002-2021 (далее - ГОСТ 27002), который содержит детальные рекомендации по обеспечению информационной безопасности, в том числе применительно к разработке и

эксплуатации прикладного программного обеспечения, что детализирует, ранее полученные сведения.

В соответствии с 14 разделом: «Необходимо управлять изменениями в системах в течение жизненного цикла разработки посредством применения формализованных процедур управления изменениями.» [3, раздел 14.2.1]. А также в соответствии с этим же разделом, раскрывается понятие безопасной разработки: «Безопасная разработка - это требование для создания безопасного сервиса, архитектуры, программного обеспечения и системы. В рамках политики безопасной разработки должны быть учтены следующие аспекты: a) безопасность среды разработки; b) руководство по безопасности в жизненном цикле разработки программного обеспечения» [3, раздел 14.2.2].

Кроме того, раздел 14.2 ГОСТ 27002 требует применения принципов безопасного проектирования: «Принципы безопасного проектирования систем должны быть установлены, задокументированы, поддерживаться и применяться к любым работам по реализации информационной системы.»[3, раздел 14.2.5] а ее механизмы должны быть встроены в архитектуру: «Процедуры безопасного проектирования информационных систем, основанные на указанных выше принципах, должны быть установлены, задокументированы и применены к внутренним процессам по проектированию информационных систем. Безопасность должна проектироваться на всех уровнях архитектуры (бизнес, данные, приложения и технологии), чтобы сбалансировать потребность в ИБ и удобстве. Новые технологии должны быть проанализированы на предмет угроз безопасности, а решения должны быть рассмотрены с точки зрения известных шаблонов атак.» [3, раздел 14.2.5]. В завершении: «Для новых информационных систем, обновлений и новых версий должны быть разработаны программы приемо-сдаточных испытаний и установлены связанные с ними критерии.» [3, раздел 14.2.9].

Для защиты промышленного манипулятора от вредоносного вмешательства, раздел 12.2 ГОСТ 27002 предписывает: «Для защиты от вредоносных программ должны быть реализованы меры обеспечения ИБ, связанные с обнаружением, предотвращением и восстановлением, в сочетании с соответствующим информированием пользователей.» [3, раздел 12.2.1]. А далее идет конкретное руководство по применению: «установление формальной политики, запрещающей использование неавторизованного программного обеспечения; реализация мер обеспечения ИБ, которые предотвращают или обнаруживают использование неавторизованного программного обеспечения (например, белый список приложений); внедрение мер обеспечения ИБ, которые предотвращают или обнаруживают обращение к известным или предполагаемым вредоносным веб-сайтам (например, ведение черного списка); установление формальной политики для защиты от рисков, связанных с получением файлов и программного обеспечения из внешних сетей или через них, либо с

помощью других способов, с указанием мер по защите; снижение числа уязвимостей, которые могут быть использованы вредоносными программами, например через менеджмент технических уязвимостей : проведение регулярных проверок программного обеспечения и содержимого систем, поддерживающих критические бизнес-процессы» [3, раздел 12.2.1, п. a-e].

Раздел 12.4 ГОСТ 27002 устанавливает требования к регистрации событий безопасности: «Журналы событий, где это применимо, должны включать в себя: a) пользовательские идентификаторы; b) действия в системе; c) дату, время и детали ключевых событии, например вход и выход из системы; d) идентификатор устройства или местоположения, если возможно, и системный идентификатор; e) записи об успешных и отклоненных попытках доступа к системе; f)записи об успешных или отклоненных попытках доступа к данным и иным ресурсам; g) изменения конфигурации системы; h) использование привилегий; i) использование системных служебных программ и приложений; j) файлы, к которым был запрошен доступ, а также вид доступа» [3, раздел 12.4.1]. Особое внимание уделяется суперпользователям: «Владельцы привилегированных учетных записей могут иметь возможность манипулировать журналами на средствах обработки информации, находящимися под их непосредственным управлением, следовательно, необходимо защищать и проверять журналы для обеспечения подотчетности привилегированных пользователей.» [3, раздел 12.4.3].

Раздел 9.2 ГОСТ 27002 так же, регламентирует управление доступом пользователей: «Процесс управления идентификаторами пользователей должен включать в себя:

a) использование уникальных идентификаторов пользователей, позволяющих отследить действия пользователей, чтобы они несли ответственность за свои действия; использование групповых идентификаторов должно быть разрешено только в тех случаях, когда это необходимо для бизнеса или в силу операционных причин и должно быть утверждено и документировано; b) немедленное отключение или удаление идентификаторов пользователей, покинувших организацию; c) периодическое выявление и удаление или отключение избыточных идентификаторов пользователей; d) обеспечение того, чтобы избыточные идентификаторы пользователей не были переданы другим пользователям.» [3, раздел 9.2.1]. Права доступа должны регулярно пересматриваться: «Владельцы активов должны регулярно пересматривать права доступа пользователей.» [3, раздел 9.2.5]. Привилегированные права, свою очередь: «d) должны быть определены требования для установления срока действия привилегированных прав доступа; e) привилегированные права доступа должны быть связаны с идентификатором пользователя, отличным от того, что используется для исполнения повседневных должностных обязанностей. Эти обязанности не

должны выполняться под привилегированным идентификатором; f) полномочия пользователей с привилегированными правами доступа должны регулярно пересматриваться с целью убедиться, что они соответствуют их обязанностям;» [3, раздел 9.2.3].

Наконец, раздел 12.6 ГОСТ 27002 требует организации процесса управления техническими уязвимостями: «В ответ на выявление потенциальных технических уязвимостей должны быть предприняты надлежащие и своевременные действия. Необходимо придерживаться следующих указаний для создания эффективного процесса управления техническими уязвимостями. a) организация должна определить и установить роли и обязанности, связанные с управлением техническими уязвимостями, включая их мониторинг, оценку риска, исправление ошибок, отслеживание активов и любые необходимые обязанности по координации процесса; b) для программного обеспечения и других технологий следует идентифицировать информационные ресурсы, которые будут использоваться для выявления соответствующих технических уязвимостей и поддержания осведомленности о них; эти информационные ресурсы должны обновляться в соответствии с изменениями в перечне активов или при обнаружении новых и полезных ресурсов; c) следует определить сроки реагирования на уведомления о потенциально значимых технических уязвимостях; d) после выявления потенциальной технической уязвимости организация должна определить связанные с ней риски и действия, которые необходимо предпринять; такие действия могут включать применение пакетов исправлений к уязвимым системам или применение компенсирующих мер обеспечения ИБ; e) в зависимости от того, насколько срочно необходимо устранить техническую уязвимость, предпринимаемые действия должны проводиться в соответствии с мерами по управлению изменениями) или в соответствии с процедурами реагирования на инциденты ИБ; f) если доступно официальное исправление, следует оценить риски, связанные с его установкой (риски, связанные с уязвимостью, следует сравнить с рисками, которые могут возникнуть вследствие установки исправления); g) пакеты исправлений должны быть проверены и оценены до их установки, чтобы быть уверенным в том, что они не приведут к недопустимым побочным эффектам; если исправлений не выпущено, следует рассмотреть иные меры обеспечения информационной безопасности, такие как» [3, раздел 12.6.1].

Таким образом, совокупность требований разделов 9.2, 12.2, 12.4, 12.6 и 14.2 ГОСТ Р ИСО/МЭК 27002-2021 создает, практическую основу для обобщения мер обеспечения целостности, контроля доступа, антивирусной защиты, аудита безопасности и безопасной разработки применительно к программному обеспечению промышленного манипулятора как исполнительного устройства, значимого объекта критической информационной инфраструктуры.

Далее следует ввести еще один национальный стандарт, ГОСТ Р 56939-2016, который вносит подробности в разработку: «При выполнении квалификационного тестирования ПО разработчик ПО должен реализовать следующие меры: функциональное тестирование программы; тестирование на проникновение; динамический анализ кода программы; фаззинг-тестирование программы.» [4, раздел 5.4.1]. Так же в программе должно быть: «Разработчик ПО должен обеспечить проведение тестирования на проникновение в отношении программы с целью выявления ее уязвимостей. Тесты, выполняемые в рамках тестирования на проникновение.» [4, п. 5.4.3.2] и так же: «Разработчик ПО должен обеспечить проведение динамического анализа кода программы с целью выявления уязвимостей программы. Тесты, выполняемые в рамках динамического анализа кода программы» [4, п. 5.4.3.3] По результатам каждого, составляются отчёты: «отчеты, содержащие список выявленных несоответствий требованиям безопасности, описание действий, направленных на их устранение, либо обоснование невозможности или отсутствия необходимости в устранении несоответствия требованиям.» [4, п. 5.4.3.1].

Таким образом, до момента передачи ПО оператору, то есть, ввода манипулятора в эксплуатацию, все выявленные уязвимости должны быть либо устранены, либо документально обоснованы как неустранимые с приведением компенсирующих мер, которые их нейтрализуют или минимизируют влияние, что соответствует требованиям безопасной разработки для критической информационной инфраструктуры.

На основании проведённого анализа нормативных документов можно подытожить следующие основные требования.

Во-первых, программное обеспечение манипулятора должно разрабатываться по принципу «безопасность с самого начала». Это означает, что механизмы защиты, такие как контроль доступа, проверка целостности, регистрация событий, закладываются в архитектуру изначально, а не добавляются потом. Разработчик обязан формализовать политику безопасной разработки, определить правила программирования и контрольные точки проверки безопасности на всех этапах создания ПО.

Во-вторых, до того, как манипулятор будет введён в эксплуатацию, его программное обеспечение должно пройти обязательное тестирование на уязвимости и ошибки. А именно: тестирование на проникновение, динамический анализ кода, фаззинг-тестирование и функциональное тестирование требований безопасности. Все выявленные уязвимости должны быть устранены, либо документально обоснованы, почему, то или иное исправление невозможно, с предложением компенсирующих мер минимизации влияния.

В-третьих, в самом манипуляторе и в связанных с ним компьютерах, должна быть реализована антивирусная защита с регулярным обновлением сигнатур. Должна действовать

политика «белых списков» - запрет на установку любого неавторизованного программного обеспечения.

В-четвёртых, система должна вести подробные журналы событий безопасности: пуск, остановка, изменение параметров. Так же должно вестись, были ли попытки несанкционированного доступа, не нарушалась ли целостность прошивки. Действия администраторов и наладчиков с привилегированными правами должны регистрироваться с особым вниманием, иными словами, детальное логирование.

В-пятых, доступ к управлению манипулятором должен быть строго разграничен по ролям. Каждый пользователь получает уникальный идентификатор. Права доступа регулярно пересматриваются, особенно при увольнении сотрудников или смене их должностей. Привилегированные права, предоставляются минимально необходимому числу доверенных лиц.

В-шестых, разработчик обязан организовать процесс управления уязвимостями: отслеживать информацию о новых уязвимостях в используемых компонентах (прошивках контроллеров, библиотеках управления движением, операционных системах), своевременно устанавливать исправления или применять компенсирующие меры.

1.2 Применение требований безопасности при разработке ПО для промышленных манипуляторов в мировой и отечественной практике

Сбор информации о конкретных автоматизированных системах на предприятиях, к примеру, на заводах «Лада», «УАЗ» и «ГАЗ» усложняется тем фактом, что внутренние правила автоматизированных систем управления процессами и протоколы информационной безопасности промышленных контроллеров и станков с ЧПУ обычно являются коммерческой или промышленной тайной. Открытые источники, такие как финансовые отчеты, новостные ленты и официальные заявления, описывают окончательные результаты развертывания автоматизированных линий или технологических инцидентов, но не раскрывают технические детали развертывания инструментов информационной безопасности на конкретных манипуляторах. Тем не менее, на основе отраслевого контекста, известных тенденций цифровой трансформации и публичных данных о типах используемого оборудования можно воссоздать типовую картину и сделать обоснованные выводы об организации безопасности на этих предприятиях. В мировой практике обеспечение безопасности промышленных роботов регламентируется комплексом стандартов, среди которых ключевыми являются IEC 61508 (функциональная безопасность электрических, электронных и программируемых электронных систем) и ISO 13849 (безопасность машин - части систем управления, связанные с безопасностью). Эти стандарты устанавливают требования к надёжности и

отказоустойчивости систем управления, включая программное обеспечение промышленных контроллеров.

Современные манипуляторы ведущих мировых производителей, таких как KUKA, проектируются с использованием принципа «безопасность с самого начала» (security-by-design). Согласно сертификационной документации KUKA, механизмы контроля целостности прошивки, аутентификации команд и защиты каналов связи устройства поступают с завода, а не добавляются в процессе работы. [7].

Одним из ключевых элементов функциональной безопасности манипуляторов является определение требуемого уровня безопасности (SIL - Safety Integrity Level). В соответствии с сертификационной документацией производителя, для типичных промышленных роботов KUKA уровень SIL 2/PL d это стандартное требование, особенно для функции аварийной остановки, когда человек приближается или обнаруживается препятствие. [7]. Сертификаты определяют требования к программному обеспечению системы безопасности, включая необходимость проверки и проверки драйверов. [7].

В технической документации KUKA примечательно, что контроллеры роботов используют безопасные протоколы связи, а критические обновления прошивки имеют цифровую подпись, что предотвращает возможность несанкционированного изменения алгоритмов управления. [7]. Эти меры напрямую соотносятся с требованиями к обеспечению целостности программного обеспечения, установленными для значимых объектов КИИ.

На Волжском, же автозаводе (АВТОВАЗ) в городе Тольятти работают промышленные роботы, выполняющие сварочные операции. Согласно открытым источникам, в цехе сварки LADA Vesta задействованы манипуляторы KUKA, которые осуществляют сварку корпусов кроссовера LADA [5]. Немецкие роботы KUKA взяли на себя задачу выполнения сложных сварочных операций, требующих высокой точности позиционирования и повторяющихся движений. [5].

Использование на заводе АВТОВАЗ роботов KUKA, имеющих сертификацию по стандартам безопасности [7], косвенно свидетельствует о том, что их программное обеспечение соответствует требованиям для контроля целостности и управления изменениями, хотя подтвердить это сложно, поскольку нет документальных доказательств. В сертификационных документах KUKA указывается, что системы управления роботами, прошли тест на отказоустойчивость и несанкционированное воздействие [7].

На основе анализа открытых публикаций можно предположить, что архитектура управления роботами на АВТОВАЗе построена по принципу распределение сети: управляющие контроллеры манипуляторов изолированы от корпоративной сети, а обмен данными между уровнями АСУ ТП осуществляется через ограниченные шлюзы с

контролируемыми протоколами передачи. Такая архитектура соответствует требованиям по ограничению программной среды и контролю целостности: ни одна новая прошивка или конфигурационный файл не могут быть переданы на контроллер без прохождения процедуры авторизации и дополнительной проверки.

Производство, же внедорожников УАЗ-3163 «Patriot» и коммерческой модели «Профи», также использует промышленные манипуляторы для сварочных и покрасочных операций. На УАЗе применяются сварочные манипуляторы производства китайской компании Shenzhen Guanhong Automation Co. [6]. Данная техника представляет собой шестиосевые технологические манипуляторы, предназначенные для автоматизации сварочных процессов [6].

Компания Shenzhen Guanhong Automation Co., если ссылаться на информацию с официального сайта, предоставляет сертификацию своей продукции [8]. Однако, в отличие от АВТОВАЗа, открытая информация о конкретных реализованных мерах информационной безопасности на УАЗе практически отсутствует, что подтверждает тезис о закрытости подобных сведений для широкой публики.

Основной акцент в защите, основанный на общих принципах построения автоматизированных систем управления процессами, основан на предотвращении физического доступа к контроллерам через последовательные порты и применении политики Application Control, белых списков, запрещающих запуск любых сценариев, драйверов или прошивок, не имеющих цифровой подписи предприятия.

На производстве есть не только внутренний рынок, но и необходимо обратить внимание на европейских автопроизводителей. Volkswagen в Ганновере. Касаемо зарубежных компаний, информация о средствах сборки, более доступна. До 2020 года сотрудники использовали промышленные роботы KUKA, просто переключая режимы на панели управления. Такая система, основанная на паролях, имела известные недостатки: она поощряла злоупотребления и часто не соответствовала требуемым стандартам безопасности. Неправильные действия недостаточно квалифицированного персонала приводили к авариям и простоям, что сказывалось на производительности завода[9].

Для решения этой проблемы Volkswagen модернизировал систему контроля доступа, внедрив электронную ключевую систему EKS от EUCHNER, основанную на транспондерной RFID-технологии. Каждый ключ имеет встроенную память, которая позволяет настраивать индивидуальные параметры доступа для каждого сотрудника. Только персонализированная система чтения и записи с индивидуально установленными параметрами может гарантировать эффективную защиту от несанкционированного доступа и помех.

Новая система позволяет: четко дифференцировать права доступа к роботам, делая их не подменяемыми и отказоустойчивыми; регистрировать все события, включая инициатора доступа, идентификатор робота, дату и время изменения параметров или режима работы; документировать соответствующие параметры процесса для повышения производительности и надёжности производства[9].

Таким образом, Volkswagen на своём производстве реализовал требование пункта 9.2.3 ГОСТ Р ИСО/МЭК 27002-2021 об управлении привилегированными правами доступа: доступ к управлению роботами предоставляется только авторизованным лицам на основе физического аутентификатора, транспондерного ключа, но при этом, используя местные нормативные документы, ISO/IEC 27002, TISAX, ISO 27001, а также отраслевые стандарты для роботов, такие как ISO 10218[9].

Если смотреть на Японский рынок сборочных линий, Toyota Motor Corporation, как крупнейший автопроизводитель в мире, он внедрил комплексную систему управления информационной безопасностью, охватывающую не только собственные заводы, но и всю цепочку поставок. Ключевым элементом этой системы являются универсальные руководящие принципы безопасности Toyota (All Toyota Security Guidelines, ATSG), которые распространяются на дочерние и аффилированные компании, дилерские центры, а также компании по аренде и лизингу автомобилей [10]. ATSG обеспечивает информационную безопасность посредством многомерного подхода, охватывающего организационное управление, управление человеческими ресурсами, техническую безопасность, физическую безопасность и реагирование на инциденты [10].

При разработке ATSG Toyota опирается на международные стандарты и национальные руководящие документы. В частности, ATSG основан на ISO/IEC 27001/27002, что является, международными стандартами систем менеджмента информационной безопасности и сводами практик, NIST Cybersecurity Framework, что в переводе, структура кибербезопасности Национального института стандартов и технологий США, а также на Cyber-Physical Security Measures Framework, выпущенном Министерством экономики, торговли и промышленности Японии (METI) [10]. Данные документы определяют перечень организационных, технических и физических мер контроля, а также устанавливают требования к созданию структур реагирования в случае инцидентов и аварий [10].

Особенностью подхода Toyota является также членство в Automotive Information Sharing & Analysis Center (Auto-ISAC) - отраслевом центре обмена информацией о киберугрозах в автомобильной промышленности, действующем в Японии и США. Через Auto-ISAC Toyota получает оперативную информацию об инцидентах в отрасли и использует её для совершенствования собственных разработок [10].

Однако, несмотря на наличие выстроенной системы кибербезопасности, Toyota столкнулась с серьёзным инцидентом, который наглядно продемонстрировал уязвимость глобальных производственных цепочек. 28 февраля 2022 года Toyota была вынуждена остановить все 28 производственных линий на 14 внутренних заводах [11]. Причиной стала атака программы-вымогателя (ransomware) на компанию Kojima Industries - поставщика третьего уровня, производящего пластиковые детали и электронные компоненты для производственной системы Toyota «точно вовремя» (just-in-time) [11].

События развивались следующим образом: 26 февраля 2022 года в файловом сервере Kojima Industries была обнаружена ошибка. После перезагрузки сервера была выявлена вредоносная программа и «угрожающее сообщение» с требованием расширить атаку на IT-систему Toyota через систему управления производством «Kanban». На следующее утро, 27 февраля, Kojima Industries отключила сетевое соединение с Toyota и внешними сетями, приостановила работу всех серверов и уведомила соответствующие государственные органы Японии и полицию [11]. В результате остановки производства за один день не было произведено около 13 000 автомобилей, что составило примерно 4–5% от ежемесячного объёма производства Toyota в Японии [11].

Министр экономики, торговли и промышленности Японии Хагиуда Коити, комментируя инцидент, отметил, что, хотя крупные компании имеют средства кибербезопасности, Правительство обеспокоено ситуацией с субподрядчиками малого и среднего бизнеса[13]. Цепочка поставок Toyota в Японии насчитывает 60 000 компаний по четырём уровням (уровням субподряда), и атака на одного поставщика третьего уровня оказалась достаточной, чтобы остановить производство крупнейшего автопроизводителя в мире [13].

Этот инцидент наглядно демонстрирует, что безопасность промышленных роботов и АСУ ТП не ограничивается контролем внутрифирменного доступа, но требует управления рисками всей цепочки поставок, включая удаленный доступ, поставщиков и производителей оборудования, к роботизированным контроллерам и системам управления производством. В ответ на этот инцидент Toyota активизировала свои усилия по внедрению избыточных сетевых подключений к ключевым поставщикам, что позволило возобновить производство уже второго марта 2022 года. [13]. Кроме того, Toyota усилила требования к поставщикам в рамках ATSG, проводя ежегодные проверки (включая выездные аудиты силами специализированной команды Toyota) состояния информационной безопасности в дочерних компаниях, дилерских центрах и у ключевых контрагентов [10].

Таким образом, Toyota на своем примере показала, что эффективная система кибербезопасности производства должна основываться на следующих принципах,

многоуровневой нормативной базе, объединяющая требования ISO 27001/27002, NIST CSF и отраслевых стандартов (Cyber-Physical Security Framework METI) [10], далее распространение требований на всю цепочку поставок (ATSG для поставщиков и дочерних компаний) с регулярными аудитами соответствия [10], отраслевая кооперация через Auto-ISAC для оперативного обмена информацией об угрозах [10].

Так же следует огласить, наличие резервных механизмов (back-up network) для обеспечения непрерывности производственного процесса при возникновении инцидентов у поставщиков.

Данный подход согласуется с требованиями о менеджменте информационной безопасности во взаимоотношениях с поставщиками, который предписывает распространять требования безопасности на всю цепочку поставок и проводить регулярный мониторинг их выполнения.

В результате анализа нормативно-правовой базы, технических стандартов и практических примеров эксплуатации программного обеспечения промышленных манипуляторов и схожих станков, сформулированы следующие положения.

Требования к безопасности программного обеспечения промышленных манипуляторов формируются на трёх уровнях. Правовой контекст задаётся Федеральным законом № 187-ФЗ, который определяет автоматизированную систему управления как комплекс программных и программно-аппаратных средств управления исполнительными устройствами, а также обязанность субъектов критической информационной инфраструктуры по созданию системы безопасности, включающей предотвращение несанкционированного доступа, обеспечение целостности и восстановление функционирования. Специальные меры защиты деталей в приказе ФСТЭК России № 239, который определяет восемнадцать обязательных к исполнению организационных и технических мер. Ключевыми из них применительно к программному обеспечению промышленного манипулятора являются идентификация и аутентификация, управление доступом, ограничение программной среды, аудит безопасности, антивирусная защита, обеспечение целостности и доступности, управление конфигурацией и обновлениями, реагирование на инциденты.

Практическую методологию реализации этих мер предоставляют национальные стандарты. ГОСТ Р ИСО/МЭК 27002-2021 устанавливает требования к политике безопасной разработки, управлению доступом, антивирусной защите, регистрации событий и управлению техническими уязвимостями. ГОСТ Р 56939-2016 дополняет эти требования обязательным проведением квалификационного тестирования программного обеспечения до ввода в эксплуатацию - тестирования на проникновение, динамического анализа кода, фаззинг-тестирования и функционального тестирования требований безопасности.

На основе анализа перечисленных документов определены ключевые требования к разработке программного обеспечения для промышленного манипулятора. Программное обеспечение должно разрабатываться по принципу «безопасность с самого начала», что означает закладку механизмов защиты в архитектуру изначально, формализацию политики безопасной разработки, определение правил программирования и контрольных точек проверки на всех этапах. До ввода манипулятора в эксплуатацию программное обеспечение обязано пройти тестирование на уязвимости, включающее имитацию атак злоумышленника, проверку работающей программы, подачу случайных или некорректных входных данных и проверку выполнения требований безопасности. Все выявленные уязвимости подлежат устранению либо документальному обоснованию невозможности устранения с предложением компенсирующих мер.

В самом манипуляторе и в управляющих им компьютерах должна быть реализована антивирусная защита с регулярным обновлением сигнатур и политика «белых списков», запрещающая установку любого неавторизованного программного обеспечения. Система обязана вести подробные журналы событий безопасности. Действия пользователей с привилегированными правами подлежат регистрации с особым вниманием. Доступ к управлению манипулятором должен быть строго разграничен. Разработчик обязан организовать процесс управления уязвимостями.

Определены основные классы уязвимостей, которые необходимо учитывать в случае разработки, во-первых, нарушение целостности прошивки или исполняемых файлов, во-вторых, несанкционированный подход к управлению, в-третьих, внедрение вредоносных программ, используемых через съемные носители или несанкционированные приложения, впоследствии отсутствие или отсутствие аудита, невозможность определить виновника инцидента, не менее важные, но не исправленные уязвимости разработки, необходимо описать и минимизировать ущерб от их наличия, и, наконец, использование устаревших или неподдерживаемых компонентов.

Анализ отечественной практики показал, что на российских автозаводах информация о конкретных реализованных системах информационной безопасности промышленных манипуляторов является в основном закрытой и представляет собой коммерческую тайну. На АВТОВАЗе используются роботы KUKA, имеющие сертификацию по стандартам функциональной безопасности, что косвенно свидетельствует о соответствии их программного обеспечения требованиям по контролю целостности и управлению изменениями. Архитектура управления построена по принципу сегментации сети: управляющие контроллеры изолированы от корпоративной сети, обмен данными осуществляется через ограниченные шлюзы. На УАЗе применяются китайские манипуляторы

Shenzhen Guanhong Automation Co., что создаёт специфические риски, связанные с разнородностью программного обеспечения и необходимостью адаптации мер защиты к конкретным типам контроллеров.

Мировая практика демонстрирует, что ведущие автопроизводители строят системы безопасности промышленных роботов на основе международных стандартов ISO/IEC 27001/27002, NIST Cybersecurity Framework, отраслевых стандартов ISO 10218 (безопасность роботов) и IEC 61508 (функциональная безопасность). На заводе Volkswagen в Ганновере внедрена транспондерная система физической аутентификации доступа к роботам KUKA, позволяющая разграничивать права доступа, регистрировать все события и документировать параметры процесса. Пример Toyota Motor Corporation демонстрирует критическую важность управления рисками цепочки поставок: в 2022 году атака программы-вымогателя на поставщика третьего уровня привела к остановке всех 28 производственных линий в Японии. Этот инцидент показал, что безопасность манипуляторов не ограничивается контролем доступа внутри предприятия, а требует распространения требований на всю цепочку поставок с регулярными аудитами и наличием резервных сетевых соединений.

Таким образом, требования к безопасности программного обеспечения промышленных манипуляторов представляют собой сложную многоуровневую систему, объединяющую законодательные акты, ведомственные требования и национальные стандарты. Ключевыми требованиями являются обеспечение целостности, контроль доступа, антивирусная защита, аудит безопасности, безопасная разработка и управление уязвимостями. Мировая практика подтверждает необходимость комплексного подхода, включающего физическую аутентификацию доступа, управление рисками цепочки поставок и соответствие международным стандартам. Невыполнение любого из перечисленных требований создаёт уязвимости, которые могут быть использованы для компьютерных атак с целью нарушения или остановки работы манипулятора, что влечёт сбои технологического процесса, материальный ущерб и травматизм персонала.

2 Проектирование архитектуры защищённого ПО и интеграция DevSecOps-практик в процесс разработки на Arduino

2.1 Создание макета для проектирования и тестирования ПО

Для разработки архитектуры по обеспечению информационной безопасности программного управления промышленным манипулятором была изготовлена масштабная модель из ABS-пластика методом печати на 3D принтере. Модель представляет собой трёхзвенную конструкцию, имитирующую основные степени подвижности реального промышленного робота: подъём стрелы, наклон руки и сжатие клешней.

В состав экспериментальной системы входят следующие компоненты. Потенциометры Wh148 B500K в количестве 3 штук. Технические характеристики: номинальное сопротивление 500 кОм, рабочее напряжение до 5 В, линейная характеристика изменения сопротивления. Используются для аналогового управления каждым исполнительным механизмом.

Шаговые двигатели 28BYJ-48 в количестве 2 штук. Технические характеристики: напряжение питания 12 В, ток 0,32 А, угол шага 5,625°, наличие встроенного редуктора с передаточным отношением 1:64. Для коммутации обмоток используются драйверы ULN2003. Двигатели обеспечивают подъём стрелы и наклон руки через червячную передачу.

Сервопривод SG-90 в количестве 1 штуки. Технические характеристики: напряжение питания 4,8–6 В, усилие 1,8 кг/см, угол поворота 180 градусов. Используется для управления захватным устройством (крюком).

Микроконтроллер Arduino Nano. Технические характеристики: микроконтроллер ATmega328P, тактовая частота 16 МГц, flash-память 32 Кбайт, оперативная память 2 Кбайт, 14 цифровых выводов (6 из которых с ШИМ), 8 аналоговых входов.

Для удобства распайки и подключения периферийных устройств используется плата расширения (шилд), устанавливаемая непосредственно на Arduino Nano. Плата расширения выполнена в форм-факторе, повторяющем расположение всех выводов микроконтроллера, и имеет гнёзда с шагом 2,54 мм для подключения проводов и компонентов без необходимости пайки. На плате расширения также выведены отдельные контакты для питания (5 В и 3,3 В) и общей земли, что позволяет организовать аккуратную и надёжную разводку соединений. Использование платы расширения исключает риск механического повреждения дорожек и контактов самой платы Arduino Nano при частых переподключениях, что особенно важно при отладке и проведении экспериментов.

Модуль Wi-Fi ESP8266. Технические характеристики: напряжение питания 3,3 В, потребляемый ток до 200 мА в пиках, поддержка стандартов 802.11 b/g/n. Используется для организации удалённого беспроводного управления.

Вспомогательные компоненты. В схеме предусмотрены две кнопки (включение питания и аппаратная перезагрузка), а также два резистивных делителя для согласования уровней напряжения между Arduino Nano (5 В) и ESP8266 (3,3 В) [15].

Подключение компонентов к микроконтроллеру Arduino Nano выполнено в соответствии со следующей схемой.

Первый шаговый двигатель, отвечающий за подъём стрелы, подключается к драйверу ULN2003. Цифровой вывод 2 микроконтроллера соединён с катушкой A+ драйвера, вывод 4 - с катушкой A-, вывод 3 - с катушкой B+, вывод 5 - с катушкой B-.

Второй шаговый двигатель, отвечающий за наклон руки, также подключается через драйвер ULN2003. Цифровой вывод 6 соединён с катушкой A+ драйвера, вывод 8 - с катушкой A-, вывод 7 - с катушкой B+, вывод 9 - с катушкой B-.

Сервопривод, управляющий захватным устройством, подключается к цифровому выводу 10, который является сигнальным проводом.

Потенциометры подключаются к аналоговым входам микроконтроллера. Потенциометр управления сервоприводом соединён с аналоговым входом A0. Потенциометр управления первым шаговым двигателем (подъём стрелы) соединён с аналоговым входом A1. Потенциометр управления вторым шаговым двигателем (наклон руки) соединён с аналоговым входом A2.

Модуль Wi-Fi ESP8266 подключается к цифровым выводам 11 и 12 для обмена данными. Вывод 11 (TX) микроконтроллера через резистивный делитель соединён со входом RX модуля. Вывод 12 (RX) микроконтроллера соединён с выходом TX модуля.

Резистивные делители, состоящие из резисторов номиналом 1 кОм и 2 кОм, установлены на линии передачи данных от Arduino к ESP8266 для понижения уровня сигнала с 5 В до 3,3 В, что необходимо для защиты модуля [15]. Кнопка перезагрузки подключена между выводом Reset и общей землёй. Общая земля объединяет все компоненты системы.

Созданный макет предназначен для отработки следующих задач: реализация базового алгоритма управления двигателями по сигналам от потенциометров; организация удалённого управления через веб-интерфейс ESP8266; внедрение контроля целостности прошивки; реализация аутентификации оператора для удалённого доступа; ведение журнала событий безопасности; отработка безопасного режима (аварийная остановка при потере связи, выходе параметров за допустимые пределы или получении некорректной команды) [14].

Полная принципиальная электрическая схема подключения всех компонентов к микроконтроллеру Arduino Nano, а так же изображение цепей питания, представлена на рисунке 1.

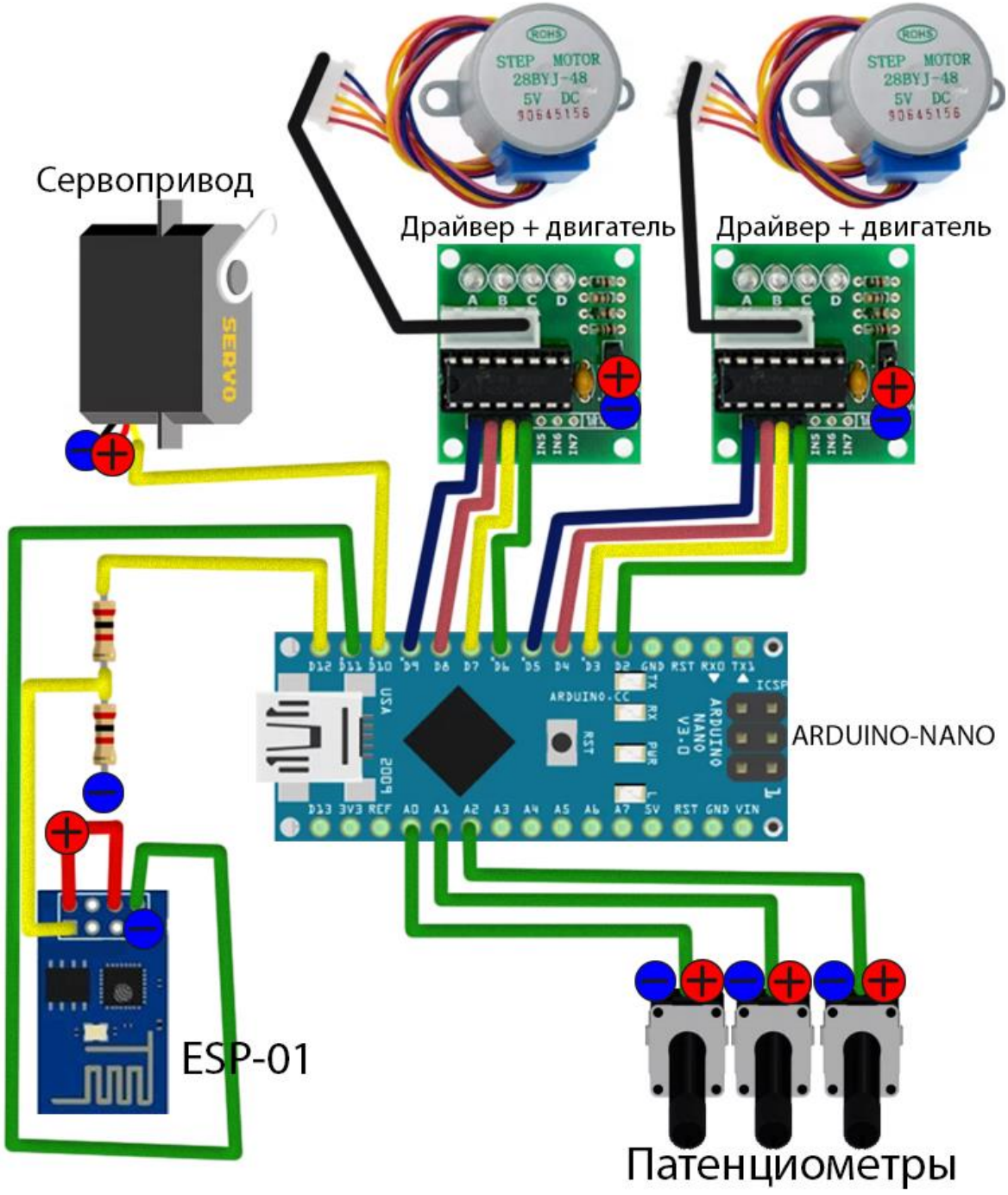

Рисунок 1 - Электрическая схема подключения

После сборки электрической схемы в соответствии с рисунком 1 необходимо закрепить все компоненты на корпусе изготовленного манипулятора, обеспечив надёжную фиксацию плат, двигателей и потенциометров. Далее следует подключить внешний источник питания напряжением 12 В к соответствующим входам драйверов шаговых двигателей, а также подать

питание 5 В на шину, обозначенную на схеме знаком «+», и общий провод (землю), обозначенный знаком «-».

Внешний вид полностью собранной экспериментальной установки, готовой к загрузке программного обеспечения, представлен на рисунке 2.

Рисунок 2 – Собранная экспериментальная установка

На рисунке 2 отчётливо видны расположение платы Arduino Nano с платой расширения, модуля ESP8266, двух шаговых двигателей с драйверами, сервопривода, трёх потенциометров и кнопок управления, закреплённых на пластиковом корпусе манипулятора.

2.2 Разработка программного обеспечения для управления экспериментальной установкой

Для реализации алгоритмов управления и последующего внедрения мер информационной безопасности было разработано программное обеспечение на языке C++ в среде Arduino IDE.

В начале программного кода определены выводы микроконтроллера, к которым подключены исполнительные устройства и датчики. Для каждого шагового двигателя заданы четыре вывода, соответствующие катушкам A+, A-, B+ и B- драйвера ULN2003. Для сервопривода выделен один ШИМ-вывод. Три аналоговых входа назначены для считывания сигналов с потенциометров, управляющих положением сервопривода и двух шаговых двигателей. Два цифровых вывода выделены для обмена данными с модулем Wi-Fi ESP8266 через программный последовательный порт SoftwareSerial.

Затем в коде вводятся константы, определяющие зону нечувствительности (мёртвую зону) потенциометров для исключения дребезга управления, а также задержку между шагами двигателя, влияющую на скорость его вращения.

В целях обеспечения информационной безопасности, а именно предотвращения подачи некорректных или опасных управляющих команд, в программу встроены механизмы валидации всех входных данных. В явном виде это реализовано через определение максимальных и минимальных диапазонов значений, которые могут быть переданы на исполнительные устройства. Использование констант, задающих зону нечувствительности потенциометров («мёртвую зону»), позволяет исключить ложные срабатывания от шумов и дребезга, которые злоумышленник мог бы интерпретировать как команды. Таким образом, архитектура кода с самого начала содержит защиту от выхода параметров управления (скорости вращения шаговых двигателей и положения сервопривода) за технологически допустимые пределы, что является практической реализацией требований к обеспечению целостности и доступности, сформулированных в главе 1.

Определяется перечисление ControlMode, содержащее два режима управления: локальный (MODE_LOCAL), при котором команды формируются по сигналам с потенциометров, и удалённый (MODE_REMOTE), при котором управляющие воздействия поступают через Wi-Fi модуль. Текущий режим сохраняется в переменной currentMode, начальное значение устанавливается в локальный режим.

Для хранения параметров удалённого управления введены переменные: целевое положение сервопривода в градусах, а также флаги включения и направления вращения для каждого из двух шаговых двигателей.

Создаётся объект SoftwareSerial для работы с Wi-Fi модулем на выбранных выводах. Также объявляется строковый буфер для приёма данных по последовательному каналу.

Далее определены функции низкоуровневого управления шаговыми двигателями: _write1 и _write2. Каждая из них принимает четыре логических значения, соответствующих состояниям обмоток двигателя, и устанавливает соответствующие выводы микроконтроллера. Данные функции используются для формирования шаговых последовательностей при вращении двигателей. Часть листинга программного кода, описанного выше представлена на рисунке 3.

06_KURS.ino

```
// ========== ПИНЫ ==========
#define DRV1_PIN_1 2
#define DRV1_PIN_2 4
#define DRV1_PIN_3 3
#define DRV1_PIN_4 5

#define DRV2_PIN_1 6
#define DRV2_PIN_2 8
#define DRV2_PIN_3 7
#define DRV2_PIN_4 9

#define SERVO_PIN 10
#define POT_SERVO_PIN A0
#define POT_MOTOR1_PIN A1
#define POT_MOTOR2_PIN A2

#define WIFI_RX_PIN 11
#define WIFI_TX_PIN 12

#define DEAD_ZONE 100
#define MOTOR_SPEED_DELAY 3

// ========== РЕЖИМЫ ==========
enum ControlMode { MODE_LOCAL, MODE_REMOTE };
ControlMode currentMode = MODE_LOCAL;

// Удалённое управление
int remoteServo = 90;
bool m1_on = false, m1_dir = false;
bool m2_on = false, m2_dir = false;

SoftwareSerial wifiSerial(WIFI_RX_PIN, WIFI_TX_PIN);
String buffer = "";

// ========== ДВИГАТЕЛИ ==========
void _write1(bool a, bool b, bool c, bool d) {
    digitalWrite(DRV1_PIN_1, a);
    digitalWrite(DRV1_PIN_2, b);
    digitalWrite(DRV1_PIN_3, c);
    digitalWrite(DRV1_PIN_4, d);
}

void _write2(bool a, bool b, bool c, bool d) {
    digitalWrite(DRV2_PIN_1, a);
    digitalWrite(DRV2_PIN_2, b);
    digitalWrite(DRV2_PIN_3, c);
    digitalWrite(DRV2_PIN_4, d);
}
```

Рисунок 3 – Фрагмент кода управления, инициализация

Для пошагового вращения двигателей разработаны функции step1 и step2, каждая из которых принимает направление вращения. Внутри хранится текущий шаг, который

увеличивается или уменьшается, после чего вызывается соответствующая процедура _write1 или _write2, подающая напряжение на обмотки двигателя.

Процедура setServo преобразует угол поворота в управляющий импульс и формирует сигнал на выводе сервопривода. Функция localControl считывает значения с трёх потенциометров. Сигнал с потенциометра сервопривода передаётся в процедуру setServo. Для шаговых двигателей значение с потенциометра определяет направление вращения или остановку с учётом зоны нечувствительности. Далее процедура remoteControl устанавливает сервопривод в положение, полученное через Wi-Fi, и при активных флагах запускает соответствующие шаговые двигатели. После чего, функция handleCmd обрабатывает команды Wi-Fi: переключает локальный и удалённый режимы, задаёт угол сервопривода, управляет направлением вращения и остановкой шаговых двигателей. Данный фрагмент программного кода с описанными элементами представлен на рисунке 4.

```
_06_KURS.ino
52  void step1(bool dir) {
53      static uint8_t step = 0;
54      dir ? step++ : step--;
55      step &= 3;
56      switch(step) {
57          case 0: _write1(1,0,1,0); break;
58          case 1: _write1(0,1,1,0); break;
59          case 2: _write1(0,1,0,1); break;
60          case 3: _write1(1,0,0,1); break;
61      }
62  }
63
64  void step2(bool dir) {
65      static uint8_t step = 0;
66      dir ? step++ : step--;
67      step &= 3;
68      switch(step) {
69          case 0: _write2(1,0,1,0); break;
70          case 1: _write2(0,1,1,0); break;
71          case 2: _write2(0,1,0,1); break;
72          case 3: _write2(1,0,0,1); break;
73      }
74  }
75
76  void setServo(int angle) {
77      int pulse = map(angle, 0, 180, 500, 2500);
78      digitalWrite(SERVO_PIN, HIGH);
79      delayMicroseconds(pulse);
80      digitalWrite(SERVO_PIN, LOW);
81  }
82
_06_KURS.ino
83  // ========== УПРАВЛЕНИЕ ==========
84  void localControl() {
85      setServo(map(analogRead(POT_SERVO_PIN), 0, 1023, 0, 180));
86
87      int p1 = analogRead(POT_MOTOR1_PIN);
88      if (p1 < 512 - DEAD_ZONE) step1(0);
89      else if (p1 > 512 + DEAD_ZONE) step1(1);
90
91      int p2 = analogRead(POT_MOTOR2_PIN);
92      if (p2 < 512 - DEAD_ZONE) step2(0);
93      else if (p2 > 512 + DEAD_ZONE) step2(1);
94  }
95
96  void remoteControl() {
97      setServo(remoteServo);
98      if(m1_on) step1(m1_dir);
99      if(m2_on) step2(m2_dir);
100 }
101
102 // ========== КОМАНДЫ ==========
103 void handleCmd(String cmd) {
104     cmd.trim();
105     if(cmd == "L") currentMode = MODE_LOCAL;
106     else if(cmd == "R") currentMode = MODE_REMOTE;
107     else if(cmd.startsWith("S")) remoteServo = constrain(cmd.subs
108     else if(cmd == "1F") { m1_on = true; m1_dir = true; }
109     else if(cmd == "1B") { m1_on = true; m1_dir = false; }
110     else if(cmd == "1S") { m1_on = false; }
111     else if(cmd == "2F") { m2_on = true; m2_dir = true; }
112     else if(cmd == "2B") { m2_on = true; m2_dir = false; }
113     else if(cmd == "2S") { m2_on = false; }
114 }
115
```

Рисунок 4 – Фрагмент кода элементов управления

После 115 строки в разработанном программном обеспечении реализован ряд мер, направленных на обеспечение информационной безопасности промышленного манипулятора в соответствии с требованиями, сформулированными в главе 1. Ниже представлено описание внедрённых механизмов защиты, а также отмечены ограничения платформы Arduino, не позволившие реализовать некоторые из требований.

Для обеспечения удалённого управления манипулятором через Wi-Fi была разработана веб-страница, формируемая процедурой webpage. Данная страница генерируется непосредственно микроконтроллером и передаётся по запросу в браузер пользователя.

Страница содержит минималистичный тёмный интерфейс с центральным расположением элементов управления. Верхняя панель предоставляет две кнопки для переключения режимов: «L» (локальный режим) и «R» (удалённый режим). При нажатии на эти кнопки отправляется GET-запрос с соответствующей командой.

Следующая панель содержит кнопки для управления первым шаговым двигателем: «▲1» (вращение вперёд) и «▼1» (вращение назад). Аналогичная панель расположена ниже для второго двигателя: «▲2» и «▼2». Реализована логика непрерывного движения: при касании или нажатии кнопки отправляется команда на включение двигателя с соответствующим направлением, а при отпускании кнопки-команда на остановку. Это достигается с помощью событий ontouchstart и ontouchend для сенсорных экранов и onmousedown и onmouseup для управления с помощью мыши.

В нижней части страницы расположен ползунковый регулятор (диапазон от 0 до 180), предназначен для управления сервоприводом. Когда вы меняете положение ползунка, команда с буквой управляется «S» и текущим значением угла.

Таким образом, онлайн-интерфейс предоставляет полный набор команд для управления манипулятором без необходимости обновления страницы, что делает его удобным для использования на настольных компьютерах и мобильных устройствах. В этом случае все отправленные команды проходят ранее описанные проверки безопасности: фильтрация символов, ограничение длины, белый список и анти-flood защиту. Листинг разработанной веб-страницы представлен на рисунке 5.

```
// ========== ВЕБ-СТРАНИЦА ==========
String webpage() {
    return "<!DOCTYPE html><html><head><meta charset='UTF-8'><meta name='viewport' content='width=device-width,
    initial-scale=1'><style>body{background:#000;margin:0;display:flex;justify-content:center;align-items:center
    ;min-height:100vh}.panel{display:flex;flex-direction:column;gap:30px}.row{display:flex;gap:15px;
    justify-content:center}button{width:80px;height:80px;font-size:24px;border:none;border-radius:50%;
    cursor:pointer;background:#333;color:#fff}.active{background:#0a0}.stop{background:#a00}
    input{width:300px;margin:10px}</style></head><body><div class='panel'><div class='row'><button id='L' onclick='fetch(\"?L\")'>ПУЛЬТ
    </button><button id='R' onclick='fetch(\"?R\")'>ВЕБ</button></div><div class='row'><button id='1F' ontouchstart='fetch(\"?1F\")'
    onmousedown='fetch(\"?1F\")' ontouchend='fetch(\"?1S\")' onmouseup='fetch(\"?1S\")'>▲1</button><button id='1B' ontouchstart='fetch(\"?1B\")'
    onmousedown='fetch(\"?1B\")' ontouchend='fetch(\"?1S\")' onmouseup='fetch(\"?1S\")'>▼1</button></div><div class='row'><button id='2F'
    ontouchstart='fetch(\"?2F\")' onmousedown='fetch(\"?2F\")' ontouchend='fetch(\"?2S\")' onmouseup='fetch(\"?2S\")'>▲2</button><button id='2B'
    ontouchstart='fetch(\"?2B\")' onmousedown='fetch(\"?2B\")' ontouchend='fetch(\"?2S\")' onmouseup='fetch(\"?2S\")'>▼2</button></div><div
    class='row'><input type='range' min='0' max='180' onchange='fetch(\"?S\"+this.value)'></div></div></body></html>";
}
```

Рисунок 5 – Листинг кода Веб-страницы

Контроль целостности прошивки: данная мера не внедрена в связи с аппаратными ограничениями платформы Arduino. На микроконтроллере ATmega328P отсутствует аппаратная защита flash-памяти для хранения эталонной контрольной суммы, которую невозможно было бы перезаписать вместе с самой прошивкой. Любой злоумышленник, имеющий физический доступ к USB-порту, может загрузить новую модифицированную

прошивку без каких-либо препятствий. Вычисление контрольной суммы (например, CRC32) технически выполнимо, но отсутствие защищённого хранилища для эталонного значения делает эту меру неэффективной. В промышленных условиях данная проблема решается применением специализированных микроконтроллеров с аппаратной поддержкой защищённых областей памяти, что подтверждает обоснованность выбора более дорогих промышленных контроллеров для критической инфраструктуры.

Watchdog-таймер: в коде реализован аппаратный сторожевой таймер (Watchdog Timer, WDT) с временем срабатывания 2 секунды. В начале каждой итерации основного цикла вызывается команда сброса таймера. Если программа зависает (например, из-за бесконечного цикла или зависания при обмене данными), сторожевой таймер не получает сигнал сброса и автоматически перезагружает микроконтроллер. Это гарантирует, что система не останется в неработоспособном состоянии на неопределённое время.

Fail-safe режим (безопасное состояние): реализована процедура emergencyStop, которая переводит систему в безопасное состояние. При её вызове происходит отключение всех фаз шаговых двигателей (подача напряжения на обмотки прекращается), сервопривод переводится в нейтральное положение (90 градусов), переключатели включения двигателя сбрасывают, а режим управления принудительно переключается на локальный. Эта процедура вызывается при запуске системы, при смене режима управления, а также при переполнении буфера приема данных, что исключает возможность несанкционированного перемещения манипулятора в аварийных ситуациях.

Белый список разрешённых команд: в программном коде реализована проверка isCommandAllowed, которая пропускает только команды определённого формата. В белый список включены: команды переключения режимов «L» и «R», команды управления двигателями «1F», «1B», «1S», «2F», «2B», «2S», а также команды установки угла сервопривода, начинающиеся с буквы «S» и содержащие числовое значение от 0 до 180. Любая команда, не соответствующая этим форматам, отклоняется с отправкой диагностического сообщения об ошибке. Данный механизм предотвращает выполнение произвольных команд, которые мог бы передать злоумышленник.

Валидация входных данных: в коде реализованы три уровня проверки входных данных. Во-первых, с помощью константы MAX_CMD_LEN ограничена максимальная длина команды (32 символа), что предотвращает атаки переполнением буфера. Во-вторых, функция isValidString проверяет каждый символ строки: допускаются только буквы, цифры и служебные символы ('S', 'F', 'B', 'L', 'R'). Это защищает от инъекций произвольных символов и управляющих последовательностей. В-третьих, для команд сервопривода выполняется проверка диапазона углов: допустимы только значения от 0 до 180, а функция constrain

дополнительно ограничивает полученное значение. Для шаговых двигателей в локальном режиме зона нечувствительности исключает ложные срабатывания от шумов и дребезга потенциометров.

Аварийный стоп по сторожевому таймеру: при срабатывании сторожевого таймера происходит аппаратная перезагрузка микроконтроллера. В процедуре setup сразу после старта вызывается emergencyStop, которая переводит все исполнительные механизмы в безопасное состояние. Таким образом, даже при критическом зависании программы манипулятор гарантированно останавливается.

Лимит на размер буфера приёма данных: для защиты от переполнения памяти при приёме данных по Wi-Fi введено ограничение на размер буфера (256 символов). При попытке записи данных сверх лимита буфер сбрасывается, а система переводится в безопасное состояние вызовом emergencyStop. Это предотвращает атаки, направленные на исчерпание памяти или переполнение буфера.

Ограничение скорости обработки команд (Anti-flood): в коде реализована защита от чрезмерно частых запросов. Константа CMD_RATE_LIMIT_MS задаёт минимальный интервал между командами (50 миллисекунд, что соответствует не более 20 команд в секунду). При превышении этого лимита команда отклоняется с отправкой сообщения «ERR: rate limit». Данная мера предотвращает атаки типа отказ в обслуживании, направленные на перегрузку системы обработки команд.

Разработанное программное обеспечение демонстрирует практическую реализацию ключевых требований информационной безопасности, сформулированных в главе 1. Несмотря на аппаратные ограничения платформы Arduino, что не позволило реализовать контроль целостности прошивки, все остальные меры успешно реализованы и обеспечивают защиту манипулятора от основных классов атак: подмена команд, флуда, переполнение буфера, запись опасных символов и зависание системы. Полный листинг кода приведенного выше представлен на рисунке 6.

Как было отмечено ранее, контроль целостности прошивки не был реализован в рамках экспериментальной установки на базе Arduino Nano в связи с аппаратными ограничениями платформы. На микроконтроллере ATmega328P отсутствует аппаратная защита flash-памяти для хранения эталонной контрольной суммы, которую невозможно было бы перезаписать вместе с самой прошивкой. Любой злоумышленник, имеющий физический доступ к USB-порту, может загрузить новую модифицированную прошивку без каких-либо препятствий. Вычисление контрольной суммы (например, CRC32) технически выполнимо, но отсутствие защищённого хранилища для эталонного значения делает эту меру неэффективной.

Для микроконтроллеров семейства AVR, к которому относится ATmega328P, существует возможность частичной защиты прошивки с использованием fuse-битов и лок-битов. Данные биты позволяют ограничить доступ к памяти: BLB12 и BLB11 управляют доступом к загрузочному сектору, а LB2 и LB1 обеспечивают защиту всей flash-памяти. При программировании лок-битов в соответствующее положение дальнейшее чтение и запись памяти через ISP-программатор становятся невозможными, и единственным способом перепрограммирования становится полное стирание чипа.

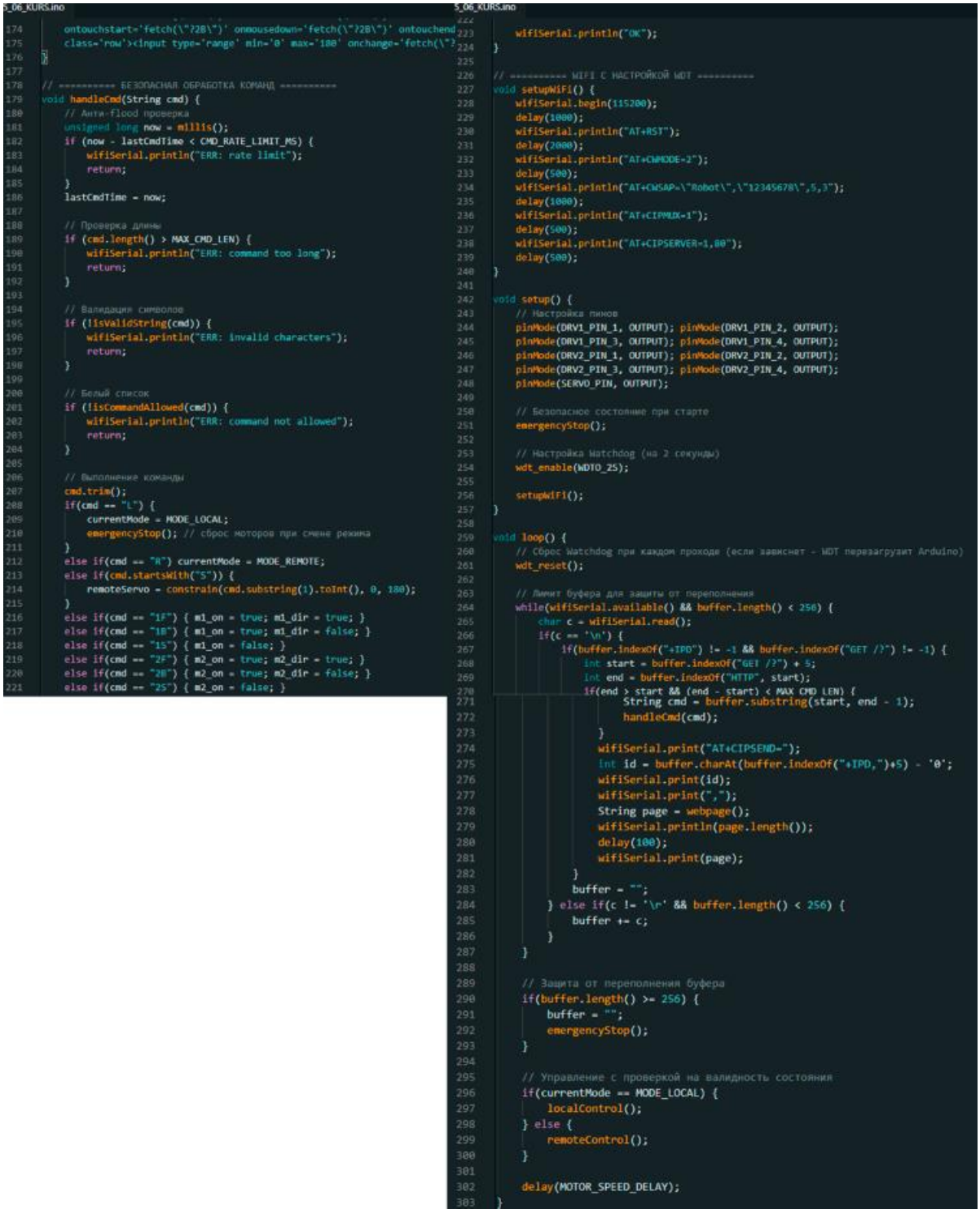

Рисунок 6 – Реализация ключевых требований ИБ

Arduino Nano построена на микроконтроллере ATmega328P, который поддерживает аналогичные lock-биты. Однако стандартная среда разработки Arduino IDE не предоставляет удобных средств для их программирования. Для записи lock-битов потребуется использовать фирменный программатор (например, Atmel-ICE) или загрузку через ISP-интерфейс с помощью утилиты avrdude с соответствующими параметрами. Кроме того, после установки lock-битов перепрограммирование микроконтроллера через стандартный USB-порт (загрузчик Optiboot) станет невозможным - потребуется полное стирание через ISP.

Таким образом, даже на платформе Arduino Nano теоретически возможно обеспечить базовую защиту от несанкционированного чтения и модификации прошивки, но ценой потери удобства перепрошивки через USB.

Заключение

В ходе выполнения работы, был произведен анализ и последующая обработка требований информационной безопасности, применимых к программному обеспечению промышленных манипуляторов с ЧПУ, на значимых объектах критической информационной инфраструктуры.

Выявлено, из нормативной документации, что промышленный манипулятор, в данном случае относиться к категории «исполнительные устройства», а его программное обеспечение к категории «прикладное программное обеспечение». Законодательство о безопасности КИИ обязывает создавать систему безопасности, интегрировать в нее предотвращение неправомерного доступа, обеспечение целостности и восстановление функционирования. Ведомственные требования регулятора детализируют меры применяемые выше. Национальные стандарты же, предоставляют методы реализации этих мер.

Сформулированы ключевые требования к разработке программного обеспечения, а конкретно, безопасная разработка по принципу «безопасность с самого начала», обязательное тестирование на уязвимости, антивирусная защита и политика «белых списков», регистрация событий безопасности, строгое разграничение доступа, управление техническими уязвимостями. Определены основные классы уязвимостей: нарушение целостности прошивки, несанкционированный доступ, внедрение вредоносного ПО, отсутствие аудита, ошибки разработки и использование устаревших компонентов.

Анализ отечественной и зарубежной практики, показал, фактическое внедрение физической аутентификации доступа, строгие разграничения, журналы аудита, безопасную разработку, применяют повсеместно, при чем, на столько успешно, что ввиду секретности, не выходит выяснить мельчайших подробностей.

В практической части работы спроектирована и реализована модель для тестирования ПО, во главе с микроконтроллером Arduino Nano, разработано программное обеспечение на языке «C++» в среде Arduino IDE, реализующее локальное и удалённое управление. Внедрены следующие меры информационной безопасности: аппаратный сторожевой таймер с автоматической перезагрузкой при зависании; процедура аварийной остановки emergencyStop; белый список разрешённых команд; трёхуровневая валидация входных данных, ограничение размера буфера приёма данных; ограничение скорости обработки команд для защиты от flood-атак. Полученные результаты могут быть использованы при проектировании защищённых систем управления промышленными манипуляторами с учётом категории значимости объекта информационной инфраструктуры.

Список использованных источников

1. Приказ ФСТЭК России от 25 декабря 2017 г. № 239 [Электронный ресурс] // Федеральная служба по техническому и экспортному контролю: официальный сайт. – Режим доступа: https://fstec.ru/dokumenty/vse-dokumenty/prikazy/prikaz-fstek-rossii-ot-25-dekabrya-2017-g-n-239 [Дата обращения 10.05.2026]

2. Федеральный закон от 26 июля 2017 г. № 187-ФЗ [Электронный ресурс] // Федеральная служба по техническому и экспортному контролю: официальный сайт. – Режим доступа: https://fstec.ru/dokumenty/vse-dokumenty/zakony/federalnyj-zakon-ot-26-iyulya-2017-g-n-187-fz [Дата обращения 11.05.2026]

3. ГОСТ Р ИСО/МЭК 27002-2021 [Электронный ресурс] // Meganorm: библиотека стандартов. – Режим доступа: https://meganorm.ru/mega_doc/norm/gost-r_gosudarstvennyj-standart/10/gost_r_iso_mek_27002-2021_natsionalnyy_standart_rossiyskoy.html

4. ГОСТ Р 56939-2016 [Электронный ресурс] // Meganorm. – Режим доступа: https://meganorm.ru/Data/626/62688.pdf [Дата обращения 12.05.2026]

5. В LADA сваркой занимаются манипуляторы KUKA [Электронный ресурс] // Robotrends. – Режим доступа: http://robotrends.ru/pub/2037/promyshlennye-manipulyatory-kuka-vzyali-na-sebya-svarku-korpusov-vnedorozhnika-lada-na-avtovaz [Дата обращения 12.05.2026]

6. Сварочные манипуляторы завода УАЗ от Shenzhen Guanhong Automation Co [Электронный ресурс] // Made-in-China. – Режим доступа: https://ru.made-in-china.com/co_szghauto/product_Industrial-Automatic-Robot-Welder-with-6-Axis-Technology_ysyeusyerg.html [Дата обращения 12.05.2026]

7. Сертификаты манипуляторов KUKA [Электронный ресурс] // Promautomatic. – Режим доступа: https://www.promautomatic.ru/certificate/ [Дата обращения 12.04.2026]

8. Сертификаты манипуляторов Shenzhen Guanhong Automation Co [Электронный ресурс] // Made-in-China. – Режим доступа: https://ru.made-in-china.com/co_szghauto/company_info.html?pv_id=1jq89079n436&faw_id=null&bv_id=1jq890oln90f&pbv_id=1jq8905pnb88p [Дата обращения 12.04.2026]

9. Volkswagen: Clearly defined access rights. Modern safety and security systems [Электронный ресурс] // EUCHNER : [сайт]. – Режим доступа: https://www.euchner.com.cn/zh-cn [Дата обращения 15.05.2026]

10. Information Security | ESG Activities | Sustainability | Toyota Motor Corporation Official Global Website [Электронный ресурс] // Toyota Global : [сайт]. – Режим доступа: https://global.toyota/en/sustainability/esg/information-security/ [Дата обращения 16.05.2026]

11. Toyota stops production in Japan after a cyberattack at a supplier [Электронный ресурс] // The New York Times : [сайт]. – Режим доступа: https://www.nytimes.com/2022/02/28/business/toyota-stoppage-cyberattack.html [Дата обращения 16.05.2026]

12. Toyota to restart Japan production after halt caused by cyberattack on supplier [Электронный ресурс] // Malay Mail : [сайт]. – Режим доступа: https://www.malaymail.com/news/money/2022/03/01/toyota-to-restart-japan-production-after-halt-caused-by-cyberattack-on-supp/2044773 [Дата обращения 17.05.2026]

13. Toyota Motor: Information Security [Электронный ресурс] // MarketScreener : [сайт]. – Режим доступа: https://www.marketscreener.com/quote/stock/TOYOTA-MOTOR-CORPORATION-6492484/news/Toyota-Motor-Information-Security-41136280/ [Дата обращения 17.05.2026]

14. Уроки Ардуино и робототехники [Электронный ресурс] // AlexGyver : [сайт]. – Режим доступа: https://alexgyver.ru/lessons/ [Дата обращения 17.05.2026]

15. ESP8266 и Arduino, подключение, распиновка [Электронный ресурс] // Habr : [сайт]. – Режим доступа: https://habr.com/ru/articles/390593/ [Дата обращения 15.05.2026]