САМАРСКИЙ ПОЛИТЕХ
Опорный университет

МИНОБРНАУКИ РОССИИ
федеральное государственное бюджетное образовательное
учреждение высшего образования
«Самарский государственный технический университет»
(ФГБОУ ВО «СамГТУ»)

Методические указания по выполнению лабораторных работ по дисциплине:

КИБЕРБЕЗОПАСНОСТЬ И КРИПТОГРАФИЯ

для обучающихся очной и заочной форм обучения
направления подготовки 13.04.02 Электроэнергетика и электротехника
профиль Цифровая трансформация и управление проектами в электроэнергетике

Новокуйбышевск, 2026

Методические указания разработаны на кафедре «Информатика и системы управления» в соответствии с требованиями ФГОС ВО по направлению подготовки **13.04.02 Электроэнергетика и электротехника**, утвержденного приказом Министерства образования и науки РФ от № 147 от 28.02.2018 и рабочей программой дисциплины «Кибербезопасность и криптография».

Методические указания предназначены для обучающихся очной, заочной и очно-заочной форм обучения и содержат указания по выполнению лабораторных работ, а также требования к оформлению отчетов по их выполнению.

Разработчик(и): Лада А.Н.

Содержание

1. Общие положения 4
2. Правила работы в лаборатории 5
3. Алгоритм проведения лабораторной работы 7
4. Материально-техническое обеспечение выполнения лабораторных работ 8
5. Учебно-методическое обеспечение выполнения лабораторных работ 8
6. Содержание лабораторных работ 11
7. Критерии и показатели оценки результата выполнения лабораторных работ 25
Приложение А 27
Приложение Б 28

1. Общие положения

Лабораторные занятия – одна из форм практического занятия, являющаяся эффективной формой учебных занятий в образовательной организации. Лабораторные занятия имеют выраженную специфику в зависимости от учебной дисциплины, углубляют и закрепляют теоретические знания. Лабораторные занятия дают наглядное представление об изучаемых процессах, обучающиеся осваивают постановку и ведение эксперимента, учатся оценивать полученные результаты, делать выводы и обобщения.

Лабораторная работа – это форма организации учебного процесса, в рамках которой обучающиеся по заданию и под руководством преподавателя самостоятельно выполняют специально разработанные задания. Лабораторная работа как вид учебного занятия должна проводиться в специально оборудованных учебных лабораториях. Продолжительность – не менее двух академических часов. Необходимыми структурными элементами лабораторной работы, помимо самостоятельной деятельности обучающегося, являются инструктаж, проводимый преподавателем, а также организация обсуждения итогов выполнения лабораторной работы.

Цели лабораторного занятия:

- углубление и закрепление знания теоретического курса путем практического изучения в лабораторных условиях изложенных в лекциях законов и положений;
- приобретение навыков в научном экспериментировании, анализе полученных результатов;
- формирование первичных навыков организации, планирования и проведения практических работ и исследований.

В зависимости от задач, решаемых на лабораторных занятиях, различают:

- ознакомительные лабораторные занятия, которые проводятся с целью закрепления и конкретизации изученного теоретического материала, а также для изучение конструктивных особенностей, устройство средств производственной деятельности (оборудования, инструментов приспособлений и т.д.) и средств исследовательской деятельности (испытательных установок, приборов и т.д.), а также их наладки и настройки;
- экспериментальные лабораторные занятия, которые проводятся с целью получение новой информации на основе формализованных методов, обеспечивающих накопление знаний, умений и практического опыта и включают экспериментальные и исследовательские задания (по изучению и отработке методики проведения различных исследований, по конструированию, переконструированию и доконструированию различных схем и приспособлений, по исследованию влияния различных факторов на свойства объектов, по определению степени соответствия экспериментальных и расчетных данных, по проверке, иллюстрации, подтверждению законов, закономерностей и т.д.;
- творческие лабораторные занятия (проблемно-поисковые работы), которые ставят соей целью получение новой информации на основе формализованных методов и обеспечивают накопление знаний, умений и практического опыта, а также включают постановку и проведение экспериментов и отличаются они только степенью проблемности экспериментальных задач, при этом речь идет об уровнях проблемности этих задач: новизне объектов, условий, в которых про-водится эксперимент по сравнению с известными ранее (к этой группе лабораторных работ относятся и работы по про-верке различных гипотез учебного и научного уровня проблемности).

При проведении лабораторных занятий учебная группа может делиться на подгруппы численностью не менее 8 человек, а в случае индивидуальной подготовки и менее.

2. Правила работы в лаборатории

Правила работы в лаборатории обязательны для исполнения всеми обучающимися, преподавателями и сотрудниками, находящимися в лаборатории. Нарушение правил влечет за собой предупреждение, отстранение от работы и/или другие дисциплинарные меры, предусмотренные уставом образовательной организации. Администрация лаборатории не несет ответственности за несчастные случаи, произошедшие в результате несоблюдения настоящих правил.

Одежда и защитные средства при выполнении лабораторной работы

При проведении лабораторных работ по дисциплине «Кибербезопасность и криптография» не предусмотрены специальная одежда и защитные средства. Обучающимся запрещается находиться в аудитории в верхней одежде при работе за компьютером.

Инструктаж перед выполнением лабораторной работы

Перед началом выполнения лабораторной работы обучающиеся должны пройти инструктаж по работе в лаборатории, оснащенной персональными компьютерами. Инструктаж может быть как общим (в начале семестра), так и индивидуальным (перед каждой работой, при необходимости). Инструктаж включает:

1. Общие правила работы в лаборатории.
2. Меры безопасности при работе с компьютерной техникой (электробезопасность, правильная посадка за рабочим местом, перерывы).
3. Правила работы с программным обеспечением.
4. Правила хранения и использования данных (конфиденциальность, резервное копирование).
5. Подтверждение прохождения инструктажа – подпись обучающегося в журнале (при необходимости).

1. Общие правила работы в лаборатории:

- поддерживать чистоту и порядок на рабочем месте;
- не оставлять личные вещи на проходах и рабочих столах;
- не употреблять пищу и напитки за компьютерами;
- не использовать постороннее программное обеспечение без разрешения преподавателя;
- сообщать о любых неисправностях оборудования или программного обеспечения лаборанту или преподавателю.
- соблюдать правильную осанку при работе за компьютером;
- регулярно делать перерывы для отдыха глаз и разминки.

2. Меры безопасности при работе с компьютерной техникой:

- запрет на эксплуатацию поврежденного оборудования: необходимо немедленно прекратить использование компьютера, если имеются повреждения корпуса или силовых кабелей, а также если в розетке отсутствует заземление;
- избежание посторонних предметов на системном блоке: на корпусе системного блока не должно быть посторонних предметов, так как это может привести к вибрациям и сбоям в работе оборудования;
- работа в сухих условиях: следует избегать работы с компьютером условиях повышенной влажности или при открытом корпусе;

- правильное расположение проводов и кабелей: провода должны располагаться так, чтобы исключить риск наступления на них или перегрузки тяжелыми предметами;
- не пытаться самостоятельно ремонтировать компьютерную технику;
- использовать средства защиты от излучения монитора (при необходимости);
- не перекрывать вентиляционные отверстия на системном блоке и мониторе;
- не прикасаться к экрану и корпусу монитора, а также изменять местоположение системного блока и монитора.

3. Правила работы с программным обеспечением:
- необходимо использовать только лицензионное программное обеспечение;
- запрещается устанавливать программное обеспечение без разрешения преподавателя;
- запрещается модернизировать или изменять параметры (настройки) операционной системы.

4. Правила хранения и использования данных
- при необходимости создавать резервные копии своих данных;
- не распространять персональные данные без разрешения;
- не посещать сайты, содержащие вирусы или вредоносное программное обеспечение.

Меры безопасности при выполнении лабораторной работы

При выполнении лабораторной работы обучающиеся обязаны соблюдать следующие меры безопасности:
- включать, отключать, работать за компьютером без разрешения преподавателя или лаборанта;
- подключать и отключать любые периферийные устройства, за исключением флэш-накопителей; прикасаться к соединительным кабелям и их разъемам.
- прикасаться к соединительным кабелям и их разъемам;
- открывать корпус системного блока, монитора, периферийных устройств;
- размещать какие-либо предметы (тетради, дискеты, книги и др.) на элементах оборудования персонального компьютеру;
- продолжать работу при наличии сбоев и неполадок функционирования, нехарактерного шума компьютера или признаков возникновения пожара (запаха гари).
- открывать корпус системного блока, монитора, периферийных устройств;

Перед началом работы обучающиеся должны убедиться в отсутствии видимых повреждений на компьютерах (нарушении целостности корпуса, нарушении изоляции проводов, неисправность индикации включения питания, признаки электрического напряжения на корпусе и т. д.), начинать работу с персональным компьютером только по указанию преподавателя.

В процессе работы обучающиеся должны при непроизвольном отключении персонального компьютера, сбоев в работе, нехарактерного шума или запаха гари необходим о немедленно сообщить преподавателю или лаборанту.

По окончании работы необходимо выполнить операцию выхода из системы (компьютер не выключать!), поставить преподавателя в известность об окончании работы с персональным компьютером.

Ответственность обучающихся при выполнении лабораторной работы

Обучающиеся несут ответственность за сохранность оборудования и программного обеспечения, предоставленных для выполнения лабораторной

работы. В случае повреждения по вине обучающегося, он обязан возместить ущерб в установленном порядке. Обучающиеся несут ответственность за нарушение конфиденциальности данных.

3. Алгоритм проведения лабораторной работы

Цель проведения лабораторных работ по дисциплине «Кибербезопасность и криптография» заключается в формировании у обучающихся практических навыков применения технологий программирования, при решении задач профессиональной деятельности, разработки алгоритмов и программ, пригодных для практического применения, использования технологий программирования при оснащении отделов, лабораторий, офисов компьютерным и сетевым оборудованием, тестирования работоспособности программ.

Содержание лабораторных работ определяется требованиями к результатам обучения по дисциплине «Кибербезопасность и криптография» (таблица 1) в виде умений и навыков в соответствии с **компетенциями**.

Таблица 1

Планируемые результаты обучения по дисциплине «Кибербезопасность и криптография»

Код и наименование компетенции	**Код и наименование индикатора достижения компетенции**	**Результаты обучения (знать, уметь, владеть, соотнесенные с индикаторами достижения компетенции)**
ПК-1 Способен участвовать в управлении проектами и цифровым развитием в сфере электроэнергетик и	**ПК-1.6** Использует методы обеспечения кибербезопасности	**Знать** методы обеспечения кибербезопасности и криптографии **Уметь** использовать методы обеспечения кибербезопасности и криптографии **Владеть** навыками использования методов обеспечения кибербезопасности и криптографии

Алгоритм проведения лабораторной работы по дисциплине «Кибербезопасность и криптография» включает шесть основных этапов. Последовательность и содержание выполнения лабораторной работы представлено в таблице 2.

Таблица 2

Последовательность и содержание выполнения лабораторной работы по дисциплине «Кибербезопасность и криптография»

Этап	**Содержание**
1. Подготовка к лабораторной работе	Получение задания от преподавателя (описание задачи, данные, требования к отчету). Изучение теоретического материала (конспекты лекций, учебная литература, методические указания). Ознакомление с используемым программным обеспечением. Подготовка плана выполнения работы.
2. Начало лабораторной работы	Запуск необходимого программного обеспечения. Загрузка предоставленных данных или подготовка данных самостоятельно. Создание новых файлов для хранения результатов работы.
3. Выполнение работы	Выполнение анализа данных в соответствии с заданием. Очистка и подготовка данных. Выбор и применение подходящих методов выполнения поставленных задач. Интерпретация результатов анализа. Фиксация промежуточных результатов. Создание таблиц, графиков, диаграмм для визуализации данных. Формулировка выводов и рекомендаций на основе результатов анализа.
4. Завершение работы	Сохранение всех рабочих файлов. Закрытие программного обеспечения. Удаление временных файлов (при необходимости).

Этап	Содержание
5. Оформление отчета	В отчете должны быть четко сформулированы цель работы, описание использованных методов, результаты анализа (с таблицами, графиками), выводы и рекомендации. Отчет должен быть оформлен в соответствии с требованиями методических указаний. Указание источников данных и использованного программного обеспечения.
6. Защита лабораторной работы	Демонстрация преподавателю результатов работы. Ответы на вопросы по методике анализа, интерпретации результатов и сделанным выводам. Объяснение ограничений использованных методов. Предложения по дальнейшему развитию анализа. Обоснование практической значимости полученных результатов лабораторной работы.

4. Материально-техническое обеспечение выполнения лабораторных работ

Для проведения лабораторных занятий и выполнения лабораторных работ по дисциплине «Кибербезопасность и криптография», образовательная организация располагает материально-технической базой, соответствующей действующим санитарным и противопожарным правилам и нормам. Помещения оснащены оборудованием персональными компьютерами, программным обеспечением, комплектом мебели для обучающихся и преподавателя, а также другими техническими средствами обучения.

Каждый обучающийся обеспечен индивидуальным неограниченным доступом к электронной информационно-образовательной среде образовательной организации из любой точки, в которой имеется доступ к информационно-телекоммуникационной сети «Интернет», как на территории образовательной организации, так и вне ее.

Компьютерная техника оснащена подключения к информационно-телекоммуникационной сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду образовательной организации.

Образовательная организация обеспечена необходимым комплектом лицензионного и свободно распространяемого программного обеспечения, в том числе отечественного производства (в соответствии с программой дисциплины) и подлежит обновлению при необходимости.

Оборудование и программное обеспечение, используемое при проведении лабораторных работ по дисциплине «Кибербезопасность и криптография»:

- программное обеспечение: Visual Studio Code (VS Code), браузер, доступ в ЭИОС и Интернет;
- образовательная платформа «Юрайт».

5. Учебно-методическое обеспечение выполнения лабораторных работ

Учебно-методическое обеспечение выполнения лабораторных работ по дисциплине «Кибербезопасность и криптография» включает:

1. Перечень литературы по дисциплине.
2. Методические указания по выполнению лабораторных работ.
3. Описание процесса проведения лабораторных работ.

1. Перечень литературы по дисциплине.

Основная литература:

1. Губарева, К.В. Системы мониторинга и управления инженерной инфраструктурой: учебное пособие / К. В. Губарева; Самарский государственный технический университет, Промышленная теплоэнергетика. - Самара, 2025.- 93 с..- Режим доступа: https://elib.samgtu.ru/getinfo?uid=els_samgtu||elib||6465

2. Криптография и безопасность сетей: учебное пособие / Фороузан Б.А., Профобразование, ред. Берлина А.Н.: 2024. - Режим доступа: https://elib.samgtu.ru/getinfo?uid=els_samgtu||iprbooks||139752

3. Основы криптографии: учебное пособие / Басалова Г.В., ИнтернетУниверситет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа: 2024. - Режим доступа: https://elib.samgtu.ru/getinfo?uid=els_samgtu||iprbooks||133959

4. Технологии искусственного интеллекта и кибербезопасность: монография / Менисов А.Б., Ай Пи Ар Медиа: 2022. - Режим доступа: https://elib.samgtu.ru/getinfo?uid=els_samgtu||iprbooks||123570

Дополнительная литература:

5. Кибербезопасность: стратегии атак и обороны: монография / Диогенес Ю., Озкайя Э., ДМК Пресс, пер. Беликов Д.А.: 2020. - Режим доступа: https://elib.samgtu.ru/getinfo?uid=els_samgtu||iprbooks||124557

6. Криптография – наука о тайнописи: учебное пособие / Фомичев В.М., Прометей: 2020. - Режим доступа: https://elib.samgtu.ru/getinfo?uid=els_samgtu||iprbooks||125666

7. Основы криптографии: учебное пособие / Басалова Г.В., ИнтернетУниверситет Информационных Технологий (ИНТУИТ), Ай Пи Ар Медиа: 2024. - Режим доступа: https://elib.samgtu.ru/getinfo?uid=els_samgtu||iprbooks||133959

8. Современные методы криптографии и кодирования: учебное пособие / Данилов С.Н., Инфра-Инженерия: 2025. - Режим доступа: https://elib.samgtu.ru/getinfo?uid=els_samgtu||iprbooks||154449

Доступ обучающихся к ЭР НТБ СамГТУ (elib.samgtu.ru) осуществляется посредством электронной информационной образовательной среды университета и сайта НТБ СамГТУ по логину и паролю.

Методические материалы размещены на сайте филиала ФГБОУ ВО «СамГТУ» в г. Новокуйбышевске, в разделе «Сведения об образовательной организации», подраздел «Образование», таблица «Информация по образовательным программам» в ячейке «Ссылка на иные компоненты, оценочные и методические материалы, а также в предусмотренных ФЗ от 29.12.2012 № 273-ФЗ «Об образовании в РФ» случаях в виде рабочей программы воспитания, календарного плана воспитательной работы, форм аттестации в виде электронного документа».

2. Методические указания по выполнению лабораторных работ.

Методические указания по выполнению лабораторных работ по дисциплине «Кибербезопасность и криптография» содержат общие положения, правила работы в лаборатории, алгоритм проведения лабораторной работы, материально-техническое обеспечение выполнения лабораторных работ, учебно-методическое обеспечение выполнения лабораторных работ, содержание лабораторных работ, критерии и показатели оценки результата выполнения лабораторных работ.

3. Описание процесса проведения лабораторных работ

Процесс проведения лабораторных работ по дисциплине «Кибербезопасность и криптография» направлен на формировании у обучающихся практических навыков в соответствии с указанной выше целью. Процесс проведения лабораторных работ состоит из шести этапов, перечисленных выше. Рассмотрим содержание и ход выполнения работ в соответствии с этапами проведения лабораторной работы.

1. Подготовка к лабораторной работе

Преподаватель предоставляет обучающим описание лабораторной работы, включающее цель работы, постановку задачи, требования к программному обеспечению, требования к отчету по результатам проведения лабораторной

работы, критерии оценки (параметры, по которым будет оцениваться выполненная работа).

Обучающиеся выбирают и осваивают программные средства, которые будут использоваться для выполнения работы, знакомятся с интерфейсом, основными функциями и инструментами, примерами выполнения учебных заданий, осуществляют поиск дополнительных материалов и документации для выполнения лабораторных задач. Обучающиеся разрабатывают план выполнения лабораторной работы, включающий определение последовательности действий, распределение времени на выполнение каждого этапа работы, подготовку необходимых ресурсов (данные, программное обеспечение, инструменты).

2. Выполнение лабораторной работы (начало лабораторной работы, выполнение работы, завершение работы.

Обучающиеся собирают необходимые для выполнения лабораторной работы данные из указанных источников, подготавливают данные для анализа, осуществляет преобразование данных, масштабирование данных. В процессе анализа данных обучающиеся применяют выбранные методы для решения поставленной задачи, интерпретируют результаты анализа данных, делая выводы о решении поставленной задачи. Выводы должны быть обоснованы результатами анализа. Обучающиеся оформляют результаты работы в виде таблиц, графиков, диаграмм. Формат представления результатов должен быть понятным и наглядным.

3. Оформление и сдача отчета

Обучающиеся оформляют отчет по лабораторной работе. Отчет включает следующие элементы:

- титульный лист (Приложения А);
- цель работы;
- индивидуальный вариант задания;
- листинг программы (исходный код с комментариями, с указанием структуры и специальной функции);
- результаты выполнения программы (скриншот консоли с выводом данных и результата);
- выводы по лабораторной работе.

Отчет должен быть оформлен в соответствии с требованиями. Текст должен быть четким и лаконичным. Таблицы и графики должны быть подписаны и пронумерованы. Формулы должны быть набраны в редакторе формул. Отчет должен быть предоставлен на проверку преподавателю в установленный срок. Образец отчета по лабораторной работе представлен в Приложении Б.

4. Защита лабораторной работы

Обучающиеся готовятся к защите, повторяя материал и анализируя результаты своей работы. Необходимо быть готовым ответить на вопросы преподавателя, касающиеся цели и задач работы, методики выполнения работы, результатов работы и сформулированных выводов.

В процессе защиты, обучающийся кратко излагает цель, задачи и результаты своей работы, отвечает на вопросы преподавателя, демонстрирует понимание результатов анализа и их практической значимости. Преподаватель оценивает качество выполненной работы, уровень знаний обучающегося и его способность применять эти знания на практике.

Обучающиеся должны активно участвовать в выполнении лабораторных работ, проявлять самостоятельность и инициативу. При анализе данных необходимо проявлять критическое мышление, оценивать достоверность информации и обоснованность выводов.

6. Содержание лабораторных работ

В рамках дисциплины «Кибербезопасность и криптография» предусмотрено 16 академических часов на выполнение лабораторных работ согласно учебному плану. Лабораторные работы предусмотрены в рамках разделов дисциплины:

- кибербезопасность в электроэнергетике;
- основы криптографии.

Темы лабораторных работ и соответствующие им планируемые результаты обучения представлены в таблице 3.

Таблица 3

Темы лабораторных работ и соответствующие планируемые результаты обучения

Темы лабораторных работ	Кол-во часов	Код индикатора достижения компетенции	Планируемые результаты обучения (умения, навыки)
Раздел 1. Кибербезопасность в электроэнергетике			
1. Особенности организации кибербезопасности в электроэнергетике	2	ПК-1.6	**Знать** методы обеспечения кибербезопасности и криптографии **Уметь** использовать методы обеспечения кибербезопасности и криптографии **Владеть** навыками использования методов обеспечения кибербезопасности и криптографии
2. Организационное обеспечение и технические средства обеспечения кибербезопасности на объектах электроэнергетики	2	ПК-1.6	**Знать** методы обеспечения кибербезопасности и криптографии **Уметь** использовать методы обеспечения кибербезопасности и криптографии **Владеть** навыками использования методов обеспечения кибербезопасности и криптографии
Раздел 2. Основы криптографии			
3. Симметричное и асимметричное шифрование	2	ПК-1.6	**Знать** методы обеспечения кибербезопасности и криптографии **Уметь** использовать методы обеспечения кибербезопасности и криптографии **Владеть** навыками использования методов обеспечения кибербезопасности и криптографии
4. Хеш-функции и электронная подпись	2	ПК-1.6	**Знать** методы обеспечения кибербезопасности и криптографии **Уметь** использовать методы обеспечения кибербезопасности и криптографии **Владеть** навыками использования методов обеспечения кибербезопасности и криптографии

Содержание лабораторных работ отражает цель, код и наименование индикатора достижения компетенции, виды работ, время выполнения лабораторной работы, информацию о подготовке к лабораторной работе, общее задание для всех вариантов ответов, индивидуальные варианты заданий, методику выполнения, содержание отчета (шаблон), выводы, контрольные вопросы (для допуска и защиты).

Лабораторная работа №1. Особенности организации кибербезопасности в электроэнергетике

Цель: сформировать навык использования методов обеспечения кибербезопасности и криптографии для защиты автоматизированных систем управления технологическими процессами (АСУ ТП) в электроэнергетике. Научиться: анализировать архитектуру SCADA-систем, настраивать шифрование каналов связи между центром управления и подстанцией, проверять целостность конфигураций устройств РЗА, выявлять аномальный трафик.

Код и наименование индикатора достижения компетенции: ПК-1.6 Использует методы обеспечения кибербезопасности

Время выполнения: 2 часа.

Подготовка к работе (повторить):

- понятие АСУ ТП, SCADA-системы, архитектура подстанции (уровни: поле, БМРЗ, КП, АРМ, серверы);
- базовые протоколы электроэнергетики: Modbus, IEC 60870-5-104, IEC 61850 (GOOSE, SV);
- методы шифрования (симметричное и асимметричное) и хэширования (MD5, SHA, ГОСТ Р 34.11-2012);
- механизмы аутентификации, контроля целостности прошивок и обнаружения вторжений (IDS/IPS);
- нормативные требования: Приказ Минэнерго №300, стандарты NERC CIP, СО 153-34.

Общее задание для всех вариантов:

Разработайте программу (на C++ или Python), которая:

- определяет структуру PowerEquipment (Энергооборудование) для описания элемента электроэнергетической системы. Поля структуры приведены в вашем варианте;
- в функции main() создает массив (std::vector в C++ или list в Python) из 3–5 структур PowerEquipment с данными согласно вашему варианту;
- реализует функцию printPowerEquipment(const PowerEquipment& item) (или print_power_equipment(item)), которая выводит информацию об одном объекте в удобочитаемом виде;
- реализует функцию для решения задачи, специфичной для вашего варианта (например, проверка сертификата устройства, шифрование телеметрии, фильтрация аномального трафика);
- выводит исходный список оборудования и результат работы специальной функции.

Индивидуальные варианты заданий (базовые поля для всех вариантов):

```
cpp
int id;                 // Уникальный идентификатор устройства
std::string name;       // Наименование (например, "МРЗ Сириус-2М")
std::string substation;// Подстанция (например, "ПС 220 кВ Западная")
int year_commissioned; // Год ввода в эксплуатацию
```

Вариант 1. Микропроцессорное устройство РЗА (Релейная защита и автоматика)

Доп. поле в структуре: `std::string firmware_hash;` // Хэш прошивки (SHA-256)

Специальная задача: проверить и вывести список устройств, у которых хэш прошивки не соответствует эталонному (признак возможной модификации).

Пример данных:

(1, "Сириус-2М-Л", "ПС Западная", 2020, "a3f5c1..."),

(2, "БМРЗ-100", "ПС Восточная", 2019, "ff00a2...")

Вариант 2. Контроллер телемеханики (RTU)

Доп. поле: `bool is_encrypted;` // Шифрование канала активно (true/false)

Специальная задача: подсчитать количество устройств, работающих в незашифрованном режиме, и выдать предупреждение.

Пример данных:

(1, "RTU-3000", "ПС Северная", 2021, true),

(2, "МТК-50", "ПС Южная", 2018, false)

Вариант 3. Сервер SCADA

Доп. поле: `int failed_auth_attempts;` // Количество неудачных попыток аутентификации за последние сутки

Специальная задача: найти серверы с числом неудачных аутентификаций выше порога (например, >10) – признак атаки подбора пароля.

Пример данных:

(1, "SCADA Master", "Диспетчерская", 2022, 3),

(2, "SCADA Backup", "Диспетчерская", 2022, 127)

Вариант 4. Устройство синхронизации времени (GPS/IRIG-B)

Доп. поле: `double time_error_ms;` // Ошибка синхронизации, мс

Специальная задача: вывести устройства с ошибкой синхронизации > 5 мс (опасность нарушения последовательности событий).

Пример данных:

(1, "GPS-Clock-100", "ПС Центральная", 2023, 0.5),

(2, "IRIG-B-GEN", "ПС Новая", 2020, 12.3)

Вариант 5. Программно-аппаратный криптошлюз

Доп. поле: `std::string cipher_algorithm;` // Алгоритм шифрования (ГОСТ 28147-89, AES-256, отсутствует)

Специальная задача: подсчитать количество устройств, использующих разрешённый алгоритм (например, ГОСТ 28147-89).

Пример данных:

(1, "Криптошлюз КШ-1", "ЦУС", 2021, "ГОСТ 28147-89"),

(2, "CryptoGate-500", "ЦУС", 2022, "AES-256"),

(3, "Безопасный маршрутизатор", "ЦУС", 2020, "отсутствует")

Вариант 6. Коммутатор уровня ядра КСПД

Доп. поле: `int vlan_sec;` // Номер защищённого VLAN (0 – не настроен)

Специальная задача: найти коммутаторы с незащищённым управляющим VLAN (vlan_sec == 0).

Пример данных:

(1, "Moxon NPort", "ПС Западная", 2019, 100),

(2, "Cisco IE-2000", "ПС Восточная", 2020, 0)

Вариант 7. Сервер сбора данных (Historian)

Доп. поле: `double encrypt_ratio;` // Доля зашифрованных архивируемых данных (0.0 – 1.0)

Специальная задача: найти среднюю долю шифрования архивов по всем серверам.

Пример данных:

(1, "PI Server", "ДЦ", 2019, 0.95),

(2, "GE Historian", "ДЦ", 2021, 0.88)

Вариант 8. Устройство противоаварийной автоматики (ПА)

Доп. поле: `bool has_digital_signature;` // Наличие цифровой подписи на уставках

Специальная задача: вывести устройства без цифровой подписи (уязвимы к подмене команд).

Пример данных:

(1, "ПА Блок-200", "ПС Энергетическая", 2020, true),

(2, "Сириус-ПА", "ПС Индустриальная", 2022, false)

Вариант 9. Удалённый терминал сбора данных (IED)

Доп. поле: std::string protocol; // Используемый протокол (Modbus, 104, GOOSE)

Специальная задача: отфильтровать устройства, использующие небезопасный протокол (Modbus без шифрования или 104 без TLS).

Пример данных:

(1, "SEL-751", "ПС Речная", 2018, "GOOSE"),

(2, "IED-2000", "ПС Лесная", 2020, "Modbus")

Вариант 10. Система мониторинга переходных режимов (СМПР)

Доп. поле: double anomaly_score; // Оценка аномальности трафика (0 – норма, 1 – критично)

Специальная задача: вывести устройства с anomaly_score > 0.7 (вероятная кибератака или нештатный режим).

Пример данных:

(1, "PMU-400", "ПС Горная", 2021, 0.2),

(2, "Монитор СМПР", "ПС Прибрежная", 2020, 0.89)

Методика выполнения:

1. Выберите свой вариант задания (по номеру в журнале или указанию преподавателя).
2. Создайте новый проект в среде разработки (C++ / Python).
3. Объявите структуру PowerEquipment с полями, указанными в вашем варианте (базовые + доп.).
4. Напишите функцию printPowerEquipment.
5. Напишите функцию, решающую специальную задачу вашего варианта.
6. В функции main() инициализируйте массив/вектор/список данных (минимум 3 записи). Данные должны быть реалистичными для объектов электроэнергетики.
7. Протестируйте программу. Убедитесь, что специальная функция работает корректно.
8. Выведите результаты на экран.

Содержание отчета (шаблон):

1. Титульный лист (по форме Приложения А).
2. Цель работы.
3. Индивидуальный вариант задания (скопируйте текст своего варианта).
4. Листинг программы (исходный код с комментариями, особенно где объявлена структура и специальная функция).
5. Результаты выполнения программы (скриншот консоли с выводом данных и результата кибербезопасностной задачи).
6. Выводы.

Выводы (примерная структура для заполнения обучающимся):

1. В чем заключается особенность обеспечения кибербезопасности в электроэнергетике по сравнению с обычными корпоративными сетями?
2. Опишите, как в вашей программе реализована проверка критерия кибербезопасности (например, контроль хэша прошивки или шифрования). Почему это важно для АСУ ТП?
3. Какие криптографические методы могут быть добавлены для защиты протоколов (Modbus, IEC 104) в реальной системе?

Контрольные вопросы (для допуска и защиты):

1. Объясните, почему в электроэнергетике недопустимо отключать устройство для установки обновления безопасности без специального разрешения.
2. Что такое «глубокоэшелонированная защита» (defense in depth) и как она применяется на подстанции?

3. Какие стандарты IEC 61850 (GOOSE, SV) делают системы РЗА уязвимыми для кибератак, и как криптография может снизить риски?

4. Предложите, как можно модифицировать вашу программу для имитации атаки подмены команды телеуправления (spoofing). Какое поле структуры для этого добавить?

5. Почему для устройств РЗА критична проверка целостности прошивки (firmware hash)? Какой алгоритм хэширования вы бы рекомендовали?

6. Перечислите три основных метода криптографической защиты, обязательных для систем диспетчерского управления согласно Приказу Минэнерго №300.

Лабораторная работа №2. Организационное обеспечение и технические средства обеспечения кибербезопасности на объектах электроэнергетики

Цель: сформировать навык использования методов обеспечения кибербезопасности и криптографии для защиты автоматизированных систем управления технологическими процессами (АСУ ТП) объектов электроэнергетики. Научиться: применять организационные меры (разграничение доступа, политики безопасности) и настраивать технические средства (сетевые экраны, криптошлюзы) для защиты каналов связи и данных.

Код и наименование индикатора достижения компетенции: ПК-1.6 Использует методы обеспечения кибербезопасности

Время выполнения: 2 часа.

Подготовка к работе (повторить):

- основные угрозы для объектов электроэнергетики (согласно ФСТЭК, Приказ №239);
- организационное обеспечение: политика парольной защиты, управление доступом к АСУ ТП, журналы событий;
- технические средства: межсетевые экраны промышленного назначения (NGFW), криптошлюзы (ViPNet, Континент-АП), средства обнаружения вторжений (COB/IDS);
- базовые алгоритмы криптографии: хэширование (HMAC), симметричное шифрование (AES-256), формирование электронной подписи для технологических команд.

Общее задание для всех вариантов (симуляция на языке Python):

Разработайте программу, которая:

- моделирует объект электроэнергетики (подстанция, ТЭЦ, ГЭС, АЭС, диспетчерский пункт);
- реализует организационные меры: проверку прав доступа оператора по роли (диспетчер, инженер, аудитор) и логирование событий в журнал;
- реализует технические средства: эмуляцию криптографической защиты канала передачи команд между АСУ ТП и исполнительным устройством (выключатель, регулятор, защита);
- в функции main() создает не менее 3 различных устройств (согласно варианту), демонстрирует попытки передачи команд от легитимного и нелегитимного оператора, а также проверку целостности команды с использованием хэша или имитовставки;
- выводит на экран: попытку доступа, результат проверки прав, сгенерированный ключ/имитовставку, сообщение об успешной/отклонённой команде.

Индивидуальные варианты заданий:

Обязательные общие поля для всех вариантов (в программной модели):

- device_id – идентификатор устройства АСУ ТП
- device_name – наименование (напр. «Выключатель В-110», «Регулятор Р-ТЭЦ»)
- operator_role – роль оператора (диспетчер, инженер, аудитор)
- command – команда управления («Отключить», «Включить», «Установить уставку 50%», «Запросить телеметрию»)

Вариант 1. Пункт управления подстанцией 110 кВ

- Доп. атрибут модели: allowed_commands – список разрешённых команд для роли «диспетчер».
- Специальная задача: реализовать проверку, что команда, переданная диспетчером, входит в список разрешённых. Если нет – отклонить с записью в журнал событий.

Вариант 2. ТЭЦ (теплоэлектроцентраль), цех автоматизированных систем защиты

- Доп. атрибут: emergency_mode_flag (True/False) – признак аварийного режима.
- Специальная задача: при аварийном режиме разрешить выполнение команд только от роли «диспетчер» (даже если у инженера есть права в обычном режиме).

Вариант 3. ГЭС, система управления затворами

- Доп. атрибут: hmac_key – симулированный ключ для имитовставки (целостность команды).
- Специальная задача: перед выполнением команды вычислить HMAC от команды, сравнить с эталонным; если не совпадает – отклонить (защита от подмены команды в канале связи).

Вариант 4. АЭС, система блокировки аварийной защиты

- Доп. атрибут: digital_signature – эмуляция электронной подписи оператора.
- Специальная задача: разрешить выполнение критической команды («Сброс защиты», «Ввод в эксплуатацию») только при наличии корректной подписи оператора с ролью не ниже «Инженер АСУ ТП».

Вариант 5. Солнечная электростанция, SCADA-сервер

- Доп. атрибут: session_id – идентификатор сеанса связи.
- Специальная задача: блокировать выполнение команды, если session_id не соответствует активному сеансу (защита от replay-атак).

Вариант 6. Дизельная электростанция (резервный источник)

- Доп. атрибут: last_command_time – время последней команды.
- Специальная задача: запрещать команды, поступающие чаще 1 раза в 10 секунд от одного оператора (защита от DoS-атак на контроллер).

Вариант 7. Ветропарк, удаленный диспетчерский пункт

- Доп. атрибут: vpn_status (0/1) – симуляция защищённого VPN-канала.
- Специальная задача: передавать команду только если vpn_status == 1 (криптошлюз активен), иначе – отказ с уведомлением о небезопасном канале.

Вариант 8. Тяговая подстанция (ж/д транспорт)

- Доп. атрибут: encryption_key – симулированный ключ AES.
- Специальная задача: реализовать простейшее симметричное шифрование команды (можно эмуляцией: xor с ключом) и расшифровку перед выполнением. Показать, что без ключа команда невыполнима.

Вариант 9. Промышленная ТЭЦ, система сбора данных (телеметрия)

- Доп. атрибут: log_file – симулированный журнал событий (список строк).

- Специальная задача: при попытке несанкционированного доступа записывать в журнал: время, роль оператора, команду. В конце работы вывести журнал.

Вариант 10. Мини-ГЭС, автоматический регулятор частоты

- Доп. атрибут: two_factor_required – требует ли команда двухфакторной аутентификации.
- Специальная задача: для команд «Аварийный сброс нагрузки» и «Пуск генератора» требовать подтверждения вторым фактором (симуляция: ввод кода 123456). Без кода – отказ.

Методика выполнения:

1. Выберите свой вариант задания (по номеру в журнале или указанию преподавателя).
2. Создайте новый проект в среде разработки (Python 3.x).
3. Объявите класс или именованный кортеж для моделирования оборудования (структуру можно через dataclass или namedtuple). Добавьте обязательные и дополнительные поля согласно вашему варианту.
4. Напишите функцию check_organizational_security(operator_role, device, command) – реализует организационное обеспечение (проверку прав доступа).
5. Напишите функцию apply_cryptography(command, key_or_flag) – эмулирует криптографическую защиту (вычисление имитовставки, шифрование, проверку подписи согласно варианту).
6. В функции main() создайте список из 3-5 устройств (структур) с реалистичными названиями для электроэнергетики (например, «Выключатель ВЛ-110», «Регулятор Р-ТЭЦ №2»).
7. Протестируйте программу: выполните легитимную передачу команды и нелегитимную (не тот оператор, подмена данных, отсутствие шифрования).
8. Выведите на экран:
   - исходные данные об устройстве;
   - результат организационной проверки доступа;
   - результат криптографической проверки;
   - итоговое сообщение (команда выполнена / отклонена) + запись в журнал.

Содержание отчета (шаблон):

1. Титульный лист (по форме Приложения А).
2. Цель работы.
3. Индивидуальный вариант задания (скопируйте текст своего варианта).
4. Листинг программы (исходный код с комментариями, обязательно выделите функции, реализующие **организационное** и **криптографическое** обеспечение).
5. Результаты выполнения программы (скриншот консоли с выводом: попытки доступа, результаты проверок, отклонённые/выполненные команды).
6. Выводы.

Выводы:

1. Какие преимущества даёт разделение кибербезопасности на организационную (роли, политики, журналы) и **техническую** (криптография, сетевые экраны) составляющие на объекте электроэнергетики? Почему нельзя обойтись только одним из подходов?
2. Опишите, как в вашей программе реализована защита канала передачи команды. Какой метод криптографии (имитовставка, шифрование, подпись) был применён и почему он подходит для АСУ ТП?
3. Как бы вы изменили программу, если бы команда передавалась не в локальной симуляции, а по реальному промышленному протоколу (Modbus, IEC

60870-5-104, IEC 61850)? Какие технические средства (криптошлюз, межсетевой экран) добавились бы?

Контрольные вопросы (для допуска и защиты):

1. Объясните, чем отличается организационное обеспечение от технического на примере запрета выполнения опасной команды на подстанции.
2. В чём разница между симметричным шифрованием (AES) и хэшированием (HMAC) для целостности команды в АСУ ТП? Какой метод лучше для проверки отсутствия подмены команды?
3. Предложите, как можно модифицировать программу, чтобы она использовала реальную криптографическую библиотеку (например, hashlib, cryptography в Python) вместо эмуляции. Как это повлияет на безопасность?
4. Почему на объектах электроэнергетики критически важно разграничивать роли «диспетчер» и «инженер»? Приведите реальный сценарий атаки, если такое разграничение отсутствует.
5. Какие нормативные документы РФ (ФСТЭК, Минэнерго) требуют применения криптошлюзов на каналах связи между диспетчерским центром и подстанцией?

Лабораторная работа №3. Симметричное и асимметричное шифрование

Цель: сформировать навык использования методов симметричного и асимметричного шифрования для обеспечения конфиденциальности и целостности информации в автоматизированных системах. Научиться: генерировать ключи, шифровать и расшифровывать данные с использованием алгоритмов AES (симметричное) и RSA (асимметричное), выбирать подходящий алгоритм в зависимости от задачи.

Код и наименование индикатора достижения компетенции: ПК-1.6 Использует методы обеспечения кибербезопасности

Время выполнения: 2 часа.

Подготовка к работе (повторить):

- принципы симметричного шифрования (общий секретный ключ, алгоритмы AES, ГОСТ 28147-89);
- принципы асимметричного шифрования (открытый и закрытый ключ, алгоритмы RSA, ГОСТ Р 34.10-2012);
- различия областей применения: симметричное – для больших объёмов данных (скорость), асимметричное – для безопасной передачи ключей и электронной подписи;
- гибридные схемы (например, TLS/SSL: асимметричное шифрование для обмена сеансовым ключом, симметричное – для самих данных);
- базовые операции в Python с библиотеками cryptography, hashlib, pycryptodome.

Общее задание для всех вариантов (реализация на языке Python):

Разработайте программу, которая:

1. Генерирует ключи для симметричного алгоритма (AES-256) и асимметричного алгоритма (RSA-2048).
2. Шифрует и расшифровывает заданное текстовое сообщение (строку) двумя способами:
   - с использованием симметричного шифрования (AES в режиме GCM или CBC);
   - с использованием асимметричного шифрования (RSA).
3. Демонстрирует ключевые отличия:

- скорость выполнения операций (замер времени);
- возможность безопасной передачи открытого ключа по незащищённому каналу;
- ограничение на длину сообщения для RSA (длина ≤ размер ключа).

4. В функции main() выполняет:
- генерацию ключей;
- ввод сообщения пользователем (или задание сообщения согласно варианту);
- шифрование и расшифровку обоими методами;
- вывод исходного текста, шифротекста (в hex-формате), результата расшифровки;
- сравнение времени выполнения.

5. Программа должна корректно обрабатывать ошибки (некорректный ключ, подмена шифротекста, превышение длины сообщения для RSA).

Индивидуальные варианты заданий:

Базовый набор операций для всех вариантов (обязательная часть):
- AES-256 (симметричное шифрование)
- RSA-2048 (асимметричное шифрование)

Вариант 1. Передача конфиденциальных параметров АСУ ТП
- Доп. требование: сообщение содержит технологические параметры (например, "Уставка_P=5.5 МПа; T=450°C").
- Специальная задача: для RSA вывести размер открытого ключа (в байтах) и показать, что сообщение, превышающее 190 байт (для RSA-2048), шифровать нельзя. Реализовать проверку длины.

Вариант 2. Обмен сеансовым ключом (гибридная схема)
- Доп. требование: сгенерировать случайный сеансовый ключ (32 байта), зашифровать его алгоритмом RSA, а сами данные зашифровать AES с этим сеансовым ключом.
- Специальная задача: продемонстрировать, что злоумышленник, имея только RSA-зашифрованный сеансовый ключ, не может расшифровать данные без закрытого RSA-ключа.

Вариант 3. Сравнение производительности
- Доп. требование: зашифровать и расшифровать одно и то же сообщение длиной 1 Кбайт (сгенерировать искусственно) методами AES и RSA (если RSA позволяет длина, иначе разбить на блоки).
- Специальная задача: измерить время выполнения (использовать time.perf_counter()) и вывести соотношение скоростей. Объяснить, почему AES значительно быстрее.

Вариант 4. Цифровая подпись сообщения (асимметричное шифрование наоборот)
- Доп. требование: подписать сообщение закрытым ключом RSA (эмуляция ЭЦП), а проверить подпись – открытым ключом.
- Специальная задача: реализовать генерацию подписи (хэш сообщения + шифрование хэша закрытым ключом) и проверку подписи. Показать, что при изменении сообщения подпись становится недействительной.

Вариант 5. Защита журнала событий подстанции
- Доп. требование: журнал состоит из 3–5 записей (например, "2025-05-24 10:00:01, Оператор Иванов, Отключение В-110").
- Специальная задача: каждую запись зашифровать симметрично (AES) с одним ключом, а сам ключ зашифровать асимметрично (RSA) для каждого из двух получателей (диспетчер, аудитор). Показать, что каждый получатель может расшифровать ключ и прочитать журнал.

Вариант 6. Передача кода аварийной защиты (короткое сообщение)

- Доп. требование: сообщение фиксированной длины – 16 символов (например, "EMERGENCY_STOP").
- Специальная задача: зашифровать его и RSA, и AES. Сравнить длину шифротекста. Объяснить, почему для коротких сообщений RSA может быть удобен (не нужен предварительный обмен ключами).

Вариант 7. Атака "человек посередине" (MITM) – демонстрация уязвимости

- Доп. требование: эмулировать ситуацию, когда открытый ключ RSA подменён на публичный ключ злоумышленника.
- Специальная задача: программа создаёт два набора ключей (легитимный и атакующий), шифрует сообщение «чужим» открытым ключом, затем показывает, что легитимный получатель не может расшифровать сообщение (ошибка). Сделать вывод о необходимости сертификатов.

Вариант 8. Шифрование файла конфигурации контроллера

- Доп. требование: вместо строки – чтение небольшого текстового файла (config.txt) до 1 Кбайт.
- Специальная задача: зашифровать файл симметрично (AES-GCM) с аутентификацией. Расшифровать и проверить целостность. При повреждении файла (изменении 1 байта) – выдать ошибку.

Вариант 9. Асимметричное шифрование с сохранением ключей в PEM-файлы

- Доп. требование: сохранить сгенерированные открытый и закрытый ключи RSA в файлы (public.pem, private.pem) в формате PEM.
- Специальная задача: загрузить ключи из файлов и использовать их для шифрования/расшифровки. Объяснить, как это применяется в реальных системах (например, защита каналов АСУ ТП по протоколу IEC 62351).

Вариант 10. Гибридное шифрование с разными режимами AES (CBC vs GCM)

- Доп. требование: реализовать шифрование в двух режимах: CBC (с отдельным HMAC для целостности) и GCM (встроенная аутентификация).
- Специальная задача: показать, что в режиме CBC подмена шифротекста без подписи HMAC приводит к успешной расшифровке (но с искажёнными данными), а в GCM – расшифровка не выполняется. Демонстрация важности аутентификации.

Методика выполнения:

1. Выберите свой вариант задания (по номеру в журнале или указанию преподавателя).
2. Установите библиотеку cryptography или pycryptodome:

```
pip install cryptography
```

3. Создайте новый проект в среде разработки (Python 3.x).
4. Реализуйте функции:

- generate_symmetric_key() – генерация ключа AES-256 (32 байта) и вектора инициализации (IV, 12 байт для GCM);
- encrypt_aes(plaintext, key, iv) – шифрование AES (режим GCM или CBC);
- decrypt_aes(ciphertext, key, iv, tag) – расшифровка AES;
- generate_rsa_keys() – генерация пары RSA-2048 (открытый и закрытый ключи);
- encrypt_rsa(plaintext, public_key) – шифрование RSA (с предварительным хэшированием или OAEP);
- decrypt_rsa(ciphertext, private_key) – расшифровка RSA.

5. В функции main() выполните задание согласно вашему варианту. Обязательно выведите:

- исходное сообщение;
- зашифрованные данные (AES и RSA) в hex-формате;
- расшифрованные данные;
- время выполнения (если требуется вариантом);
- диагностику ошибок.

6. Протестируйте программу на корректных и некорректных данных (изменённый шифротекст, неправильный ключ).

Содержание отчета (шаблон):

1. Титульный лист (по форме Приложения А).
2. Цель работы.
3. Индивидуальный вариант задания (скопируйте текст своего варианта).
4. Листинг программы (исходный код с комментариями, обязательно укажите функции шифрования и расшифровки для обоих методов).
5. Результаты выполнения программы (скриншот консоли с выводом: исходный текст, шифротексты, результаты расшифровки, сравнение времени или иные демонстрации согласно варианту).
6. Выводы.

Выводы:

1. Сравните симметричное (AES) и асимметричное (RSA) шифрование по следующим параметрам: скорость, безопасность передачи ключей, максимальная длина сообщения, вычислительные затраты. Какой метод и в каком сценарии предпочтительнее для защиты данных в электроэнергетике?
2. Опишите, как в вашей программе решена проблема аутентификации данных (целостности). Почему в режиме CBC без HMAC злоумышленник может модифицировать шифротекст, а в GCM – нет?
3. Как, по вашему мнению, можно комбинировать симметричное и асимметричное шифрование в реальной SCADA-системе для безопасного обмена командами между диспетчерским центром и подстанцией?

Контрольные вопросы (для допуска и защиты):

1. Объясните, почему асимметричное шифрование не применяют для больших объёмов данных (например, для передачи видео с камер наблюдения на подстанции)? Какую гибридную схему используют на практике?
2. В чём заключается проблема передачи симметричного ключа по незащищённому каналу? Как асимметричная криптография решает эту проблему?
3. Что такое "размер блока" для RSA и почему нельзя зашифровать сообщение длиннее ключа? Как в реальных протоколах (например, TLS) обходят это ограничение?
4. Назовите российские стандарты (ГОСТ Р 34.10-2012, ГОСТ Р 34.12-2015) и укажите, какой из них соответствует симметричному, а какой – асимметричному шифрованию. Какой алгоритм из вашей работы является их аналогом?
5. Предложите, как модифицировать программу, чтобы злоумышленник не мог подменить открытый ключ при передаче (проблема доверия публичным ключам). Что такое инфраструктура открытых ключей (PKI)?
6. Почему в режиме AES-GCM не требуется отдельный HMAC? Зачем нужен аутентификационный тег (tag) и что произойдёт, если его изменить при передаче?

Лабораторная работа №4. Хеш-функции и электронная подпись

Цель: сформировать навык использования хеш-функций и электронной подписи (ЭП) для обеспечения целостности, аутентичности и неотказуемости информации в автоматизированных системах управления технологическими процессами (АСУ ТП) объектов электроэнергетики. Научиться: вычислять хеш-

значения данных, формировать и проверять электронную подпись, обнаруживать несанкционированные изменения в технологических параметрах и командах.

Код и наименование индикатора достижения компетенции: ПК-1.6 Использует методы обеспечения кибербезопасности

Время выполнения: 2 часа.

Подготовка к работе (повторить):

- свойства хеш-функций: односторонность, детерминированность, устойчивость к коллизиям, лавинный эффект;
- основные хеш-алгоритмы: MD5 (нерекомендуемый), SHA-1, SHA-256, ГОСТ Р 34.11-2012 (Стрибог);
- принципы электронной подписи: хеширование сообщения → шифрование хеша закрытым ключом подписанта → проверка открытым ключом;
- различия между ЭП и просто шифрованием: подпись обеспечивает целостность и неотказуемость, но не конфиденциальность;
- библиотеки Python: hashlib, cryptography, rsa.

Общее задание для всех вариантов (реализация на языке Python):

Разработайте программу, которая:

1. Вычисляет хеш-значение заданного сообщения (строки или файла) с использованием минимум двух хеш-алгоритмов (например, SHA-256 и ГОСТ-подобного через hashlib).
2. Демонстрирует свойства хеш-функций:
   - лавинный эффект (изменение одного символа → кардинальное изменение хеша);
   - детерминированность (одинаковое сообщение → одинаковый хеш);
   - невозможность восстановить сообщение по хешу.
3. Генерирует пару ключей (закрытый и открытый) для электронной подписи (RSA или ГОСТ Р 34.10-2012).
4. Формирует электронную подпись для сообщения (хеш сообщения шифруется закрытым ключом).
5. Выполняет проверку подписи открытым ключом и демонстрирует:
   - успешную проверку для неизменённого сообщения;
   - неуспешную проверку при изменении сообщения (даже на 1 символ).
6. В функции main() выводит исходное сообщение, хеши разными алгоритмами, сгенерированные ключи, подпись (в hex-формате), результаты проверки подписи для корректного и модифицированного сообщения.

Индивидуальные варианты заданий:

Базовый набор операций для всех вариантов (обязательная часть):

- вычисление хеша SHA-256;
- формирование и проверка ЭП (RSA-2048 + SHA-256);
- демонстрация лавинного эффекта.

Вариант 1. Контроль целостности файла конфигурации контроллера (PLC)

- Доп. требование: программа должна вычислить хеш (SHA-256) от содержимого файла config.cfg (создать заранее небольшой текстовый файл с параметрами: "PV=100; SV=50; MODE=AUTO").
- Специальная задача: сохранить хеш в отдельный файл (hash.txt). Затем изменить один байт в файле конфигурации, пересчитать хеш и сравнить. Сделать вывод о том, как хеш-функции помогают обнаружить несанкционированное изменение настроек контроллера.

Вариант 2. Электронная подпись команды диспетчера (АСУ ТП)

- Доп. требование: сообщение – технологическая команда (например, "CMD:OPEN_VALVE_V-110; TIMESTAMP=2025-05-24 12:00:00; OPERATOR=Ivanov").

- Специальная задача: подписать команду закрытым ключом диспетчера. На стороне исполнительного устройства (функция проверки) проверить подпись. Показать, что при подмене оператора в строке (например, "Petrov" вместо "Ivanov") подпись становится недействительной.

Вариант 3. Хеширование паролей операторов (безопасное хранение)

- Доп. требование: программа хранит словарь с паролями операторов ({"dispetcher": "P@ssw0rd", "engineer": "Qwerty123", "auditor": "SecurePass"}).
- Специальная задача: вместо хранения паролей в открытом виде вычислять и хранить их хеши (SHA-256) с солью (случайная строка). При аутентификации оператора программа должна вычислить хеш введённого пароля с той же солью и сравнить с сохранённым. Объяснить, почему хеширование с солью безопаснее.

Вариант 4. ЭП для журнала событий (обеспечение неизменности)

- Доп. требование: журнал событий – список из 3–5 записей (например, ["2025-05-24 08:00:01, Вход оператора Иванов", "2025-05-24 08:15:30, Отключение В-110", ...]).
- Специальная задача: подписать весь журнал (или каждую запись) электронной подписью. При попытке добавить или изменить запись после подписания – проверка подписи должна провалиться. Реализовать функцию verify_log_integrity(log, signature, public_key).

Вариант 5. Сравнение хеш-алгоритмов (MD5, SHA-1, SHA-256) для технологических данных

- Доп. требование: взять длинное сообщение (например, строку с данными телеметрии из 500 символов, включая "Ток=150А; Напряжение=10.5кВ; Мощность=1575кВт; ...").
- Специальная задача: вычислить хеши MD5, SHA-1, SHA-256. Показать, что MD5 и SHA-1 уязвимы к коллизиям (теоретически). Сделать вывод, какой алгоритм рекомендован ФСТЭК для объектов КИИ.

Вариант 6. ЭП с двумя ключами (инженер + диспетчер) – двойное подписание критической команды

- Доп. требование: критическая команда "ACTIVATE_EMERGENCY_SHUTDOWN" должна быть подписана двумя разными операторами (сначала инженером, затем диспетчером).
- Специальная задача: реализовать последовательное наложение двух подписей или подпись агрегированного хеша. Проверка должна проходить только если обе подписи корректны. Объяснить, зачем нужна двойная подпись на АЭС или ГЭС.

Вариант 7. Подпись временной метки (Timestamp) – защита от replay-атак

- Доп. требование: команда содержит временную метку "CMD=SET_POWER_50%; TS=2025-05-24 12:00:00".
- Специальная задача: подписать команду вместе с меткой. При проверке также проверять, что метка отличается от текущего времени не более чем на 5 минут. Показать, что старая записанная команда (replay) будет отклонена даже при корректной подписи.

Вариант 8. Хеширование и подпись обновления ПО (прошивки) для микропроцессорной защиты

- Доп. требование: файл прошивки (firmware.bin) – создать искусственно (например, текстовый файл "version=2.1.0; checksum=0xAB12").
- Специальная задача: производитель ПО вычисляет хеш файла и подписывает его. Устройство перед установкой проверяет подпись. Продемонстрировать, что любое изменение прошивки (даже 1 байта) делает

подпись недействительной. Объяснить, как это предотвращает установку вредоносного ПО.

Вариант 9. Использование хеш-функций для контроля целостности сетевого пакета (Modbus)

- Доп. требование: эмулировать Modbus-пакет в виде структуры: {"slave_id": 1, "function_code": 3, "address": 0x1000, "data": [0x01, 0x02]}.
- Специальная задача: вычислить хеш от пакета, включить хеш в пакет как поле hash. При приёме пересчитать хеш и сравнить. Показать, что при подмене данных (data изменён на [0x03, 0x04]) проверка не проходит. Аналог имитовставки, но на основе хеша.

Вариант 10. ЭП на основе ГОСТ Р 34.10-2012 (через библиотеку cryptography **или** gostcrypto**)**

- Доп. требование: вместо RSA использовать алгоритм электронной подписи по ГОСТ Р 34.10-2012 (кривые на эллиптических кривых).
- Специальная задача: сгенерировать ключи согласно ГОСТ, подписать сообщение, проверить подпись. Сравнить длину ключа и подписи с RSA-2048 (подпись ГОСТ значительно короче). Сделать вывод о применимости на встраиваемых контроллерах с ограниченными ресурсами.

Методика выполнения:

1. Выберите свой вариант задания (по номеру в журнале или указанию преподавателя).
2. Установите необходимые библиотеки:

pip install cryptography hashlib (при необходимости для ГОСТ – pip install gostcrypto).

3. Создайте новый проект в среде разработки (Python 3.x).
4. Реализуйте функции:

- calculate_hash(data, algorithm="sha256") – вычисление хеша (поддерживает MD5, SHA-1, SHA-256);
- demonstrate_avalanche_effect(message) – демонстрация лавинного эффекта (исходное сообщение и сообщение с изменённым 1 символом);
- generate_signature_keys() – генерация пары ключей (RSA или ГОСТ);
- sign_message(message, private_key) – формирование электронной подписи;
- verify_signature(message, signature, public_key) – проверка подписи.

5. В функции main() выполните задание согласно вашему варианту. Обязательно выведите:

- исходное сообщение;
- хеши (минимум 2 алгоритма);
- демонстрацию лавинного эффекта;
- сгенерированные ключи (открытый – в PEM-формате);
- подпись (в hex-формате);
- результаты проверки подписи для корректного и модифицированного сообщения.

6. Протестируйте программу на различных сообщениях, включая пограничные случаи (пустая строка, длинное сообщение).

Содержание отчета (шаблон):

1. Титульный лист (по форме Приложения А).
2. Цель работы.
3. Индивидуальный вариант задания (скопируйте текст своего варианта).
4. Листинг программы (исходный код с комментариями, обязательно укажите функции: вычисления хеша, генерации ключей, подписания и проверки подписи).
5. Результаты выполнения программы (скриншот консоли с выводом: исходное сообщение, хеши, лавинный эффект, подпись, результаты проверки).

6. Выводы.

Выводы:

1. Объясните, почему хеш-функции необратимы и как это свойство используется для защиты паролей и контроля целостности. Приведите пример из вашей программы (например, вариант 3 или 1).
2. В чём отличие электронной подписи от простого хеширования? Как ЭП обеспечивает аутентичность (устанавливает автора) и неотказуемость (автор не может отказаться от подписи)? Опишите, как это реализовано в вашей программе.
3. Как, по вашему мнению, можно применить хеш-функции и ЭП для защиты каналов связи между диспетчерским центром и удалённой подстанцией? Предложите конкретный сценарий, включая формирование подписи для каждой команды.

Контрольные вопросы (для допуска и защиты):

1. Что такое коллизия хеш-функции? Почему MD5 и SHA-1 считаются устаревшими для криптографических приложений? Какой алгоритм рекомендуется ФСТЭК России для объектов КИИ?
2. Объясните лавинный эффект на примере из вашей программы. Почему это свойство критически важно для безопасности?
3. В чём разница между электронной подписью (RSA) и асимметричным шифрованием (RSA) с точки зрения использования ключей? Можно ли один и тот же закрытый ключ использовать и для подписи, и для шифрования?
4. Что такое «соль» (salt) при хешировании паролей? Покажите на примере варианта 3, как изменится хеш для одинакового пароля с разной солью.
5. Как можно атаковать систему, если используется ЭП без временной метки (вариант 7)? Что такое replay-атака и как её предотвратить?
6. Почему на объектах электроэнергетики требуется ЭП для обновления прошивок устройств РЗА (релейной защиты и автоматики)? Приведите реальный пример атаки через поддельную прошивку (например, Industroyer, CrashOverride).
7. В каких сценариях предпочтительнее использовать ЭП на основе эллиптических кривых (ГОСТ Р 34.10-2012) вместо RSA? Какое преимущество даёт меньшая длина подписи?

7. Критерии и показатели оценки результата выполнения лабораторных работ

Оценка обучающемуся за выполненную лабораторную работу выставляется по четырехбалльной шкале по итогам проверки отчета и защиты работы. Основными **критериями оценки** лабораторной работы по дисциплине «Кибербезопасность и криптография», являются:

- полнота и правильность выполнения задания (соответствие программы поставленному заданию, обработка всех требуемых случаев);
- качество кода (читаемость, структурированность, наличие комментариев, использование современных возможностей языка);
- оформление отчета (соответствие шаблону, наличие всех разделов, аккуратность, грамотность выводов);
- ответы на вопросы на защите отчета (понимание теоретических основ и практических аспектов выполненной работы);

В таблице 4 представлены критерии и показатели оценки выполнения лабораторной работы

Таблица 4

Критерии и показатели оценки выполнения лабораторной работы

Оценка	**Критерии**			
	Полнота и правильность выполнения	**Качество кода**	**Оформление отчета**	**Ответы на контрольные вопросы/защита**
«Отлично»	Задание выполнено полностью, код корректен.	Код чистый, хорошо структурирован, адекватно прокомментирован.	Отчет оформлен в соответствии с требованиями, выводы обоснованные.	Полные, уверенные, правильные ответы.
«Хорошо»	Задание выполнено с 1-2 незначительными недочетами.	Код читаем, но есть небольшие нарушения стиля.	Отчет оформлен с 1-2 замечаниями, выводы есть.	Ответы в целом правильные, но с неточностями.
«Удовлетвори-тельно»	Задание выполнено частично, есть ошибки в логике.	Код плохо отформатирован, минимальные комментарии.	Отчет оформлен небрежно, выводы поверхностные.	Ответы неполные, требуются наводящие вопросы.
«Неудовлетвори-тельно»	Задание не выполнено или выполнено неверно.	Код нечитаем.	Отчет не оформлен или оформление не соответствует требованиям.	Не может ответить на вопросы по теме.

Некоторые показатели, которые могут использоваться для оценки результата:

- функциональность – работающая реализация всех заявленных функций и возможностей.
- тестирование – наличие тестов и их результаты (покрытие кода тестами).
- креативность – оригинальные решения и подходы к реализации задачи.
- обработка ошибок – корректная обработка исключительных ситуаций и ошибок.
- пользовательский интерфейс – удобство и интуитивность интерфейса (если применимо).
- соответствие стандартам кодирования – следование принятым стандартам и стилям кодирования.

В качестве дополнительных критериев оценки лабораторной работы может быть: скорость выполнения программы, использование памяти и ресурсов, оптимизация алгоритмов и структур данных, выполнение требования к интерфейсу или взаимодействию с пользователем, используемые библиотеки и технологии, также учитывается, предоставлено ли описание работы программы, добавлены ли инструкции по запуску и использованию, документированы функции и классы в коде.

Приложение А
Образец титульного листа отчета по лабораторной работе

МИНОБРНАУКИ РОССИИ
федеральное государственное бюджетное образовательное учреждение высшего образования «Самарский государственный технический университет»
(ФГБОУ ВО «СамГТУ»)

Филиал ФГБОУ ВО «СамГТУ» в г. Новокуйбышевске

Кафедра «Информатика и системы управления»

ОТЧЕТ
по лабораторной работе № ____
по дисциплине «Кибербезопасность и криптография»
Тема: _____

Выполнил:
обучающийся ___ курса _____ группы
ФИО: _____

Принял:
ФИО: _____

Оценка: _____

Дата: _____

Новокуйбышевск, 20___

Приложение Б
Образец отчета по лабораторной работе

---

1. Цель работы: сформировать навык использования хеш-функций и электронной подписи (ЭП) для обеспечения целостности, аутентичности и неотказуемости информации в автоматизированных системах управления технологическими процессами (АСУ ТП) объектов электроэнергетики. Научиться: вычислять хеш-значения данных, формировать и проверять электронную подпись, обнаруживать несанкционированные изменения в технологических параметрах и командах.

2. Индивидуальный вариант задания. Вариант 2. Электронная подпись команды диспетчера (АСУ ТП)

- Доп. требование: сообщение – технологическая команда (например, "CMD:OPEN_VALVE_V-110; TIMESTAMP=2025-05-24 12:00:00; OPERATOR=Ivanov").
- Специальная задача: подписать команду закрытым ключом диспетчера. На стороне исполнительного устройства (функция проверки) проверить подпись. Показать, что при подмене оператора в строке (например, "Petrov" вместо "Ivanov") подпись становится недействительной.

3. Листинг программы

python

```
"""
Лабораторная работа №4
Вариант 2: Электронная подпись команды диспетчера (АСУ ТП)
Выполнил: Иванов И.И.
"""

import hashlib
import time
from cryptography.hazmat.primitives import hashes, serialization
from cryptography.hazmat.primitives.asymmetric import rsa, padding
from cryptography.hazmat.backends import default_backend

# ---------------------------------------------------------------------
# 1. Генерация ключей (закрытый и открытый для диспетчера)
# ---------------------------------------------------------------------
def generate_keys():
    """Генерирует пару ключей RSA-2048"""
    private_key = rsa.generate_private_key(
        public_exponent=65537,
        key_size=2048,
        backend=default_backend()
    )
    public_key = private_key.public_key()
    return private_key, public_key

# ---------------------------------------------------------------------
# 2. Формирование электронной подписи
# ---------------------------------------------------------------------
def sign_command(command_text, private_key):
    """
```

```
    Подписывает команду:
    1. Вычисляет хеш команды (SHA-256)
    2. Шифрует хеш закрытым ключом (RSA-PSS)
    """
    # Хеширование сообщения
    message_hash = hashlib.sha256(command_text.encode('utf-8')).digest()

    # Подпись (шифрование хеша закрытым ключом)
    signature = private_key.sign(
        command_text.encode('utf-8'),
        padding.PSS(
            mgf=padding.MGF1(hashes.SHA256()),
            salt_length=padding.PSS.MAX_LENGTH
        ),
        hashes.SHA256()
    )
    return signature, message_hash

# ---------------------------------------------------------------------
# 3. Проверка электронной подписи
# ---------------------------------------------------------------------
def verify_command(command_text, signature, public_key):
    """
    Проверяет подпись команды
    Возвращает True, если подпись корректна, иначе False
    """
    try:
        public_key.verify(
            signature,
            command_text.encode('utf-8'),
            padding.PSS(
                mgf=padding.MGF1(hashes.SHA256()),
                salt_length=padding.PSS.MAX_LENGTH
            ),
            hashes.SHA256()
        )
        return True
    except Exception as e:
        print(f"  [Ошибка проверки подписи] {e}")
        return False

# ---------------------------------------------------------------------
# 4. Демонстрация лавинного эффекта (обязательная часть)
# ---------------------------------------------------------------------
def demonstrate_avalanche_effect(original_message):
    """Показывает, как изменение 1 символа полностью меняет хеш"""
    print("\n" + "="*60)
    print("ДЕМОНСТРАЦИЯ ЛАВИННОГО ЭФФЕКТА")
    print("="*60)

    hash_orig = hashlib.sha256(original_message.encode('utf-8')).hexdigest()
```

```
    # Изменяем один символ (пробел на точку)
    modified_message = original_message.replace(" ", ".")
    hash_mod = hashlib.sha256(modified_message.encode('utf-8')).hexdigest()

    print(f"Исходное сообщение: {original_message}")
    print(f"SHA-256 (исходное): {hash_orig}")
    print(f"Изменённое (1 символ): {modified_message}")
    print(f"SHA-256 (изменённое): {hash_mod}")
    print(f"Хеши отличаются? {hash_orig != hash_mod} (да)")
    print(f"Доля отличающихся бит ~ 50% (лавинный эффект)")

# ---------------------------------------------------------------------
# 5. Основная функция
# ---------------------------------------------------------------------
def main():
    print("="*60)
    print("ЛАБОРАТОРНАЯ РАБОТА №4. ВАРИАНТ 2")
    print("Электронная подпись команды диспетчера (АСУ ТП)")
    print("="*60)

    # 1. Генерация ключей для диспетчера
    private_key, public_key = generate_keys()
    print("\n[КЛЮЧИ СГЕНЕРИРОВАНЫ]")

    # Сохраняем открытый ключ в PEM (для наглядности)
    public_pem = public_key.public_bytes(
        encoding=serialization.Encoding.PEM,
        format=serialization.PublicFormat.SubjectPublicKeyInfo
    )
    print(f"Открытый ключ (PEM, первые 70 символов):\n{public_pem[:70]}...")

    # 2. Оригинальная команда диспетчера
    original_command = "CMD:OPEN_VALVE_V-110; TIMESTAMP=2025-05-24
12:00:00; OPERATOR=Ivanov"
    print(f"\n[ОРИГИНАЛЬНАЯ КОМАНДА ДИСПЕТЧЕРА]\n{original_command}")

    # 3. Подпись команды
    signature, command_hash = sign_command(original_command, private_key)
    print(f"\n[ПОДПИСЬ СФОРМИРОВАНА]")
    print(f"Хеш команды (SHA-256): {command_hash.hex()}")
    print(f"Электронная подпись (первые 64 байта в hex): {signature[:64].hex()}...")

    # 4. Проверка подписи для корректной команды
    print("\n" + "-"*40)
    print("ПРОВЕРКА 1: Оригинальная команда (без изменений)")
    is_valid = verify_command(original_command, signature, public_key)
    print(f"Результат проверки подписи: {'УСПЕШНО' if is_valid else
'НЕУСПЕШНО'}")

    # 5. Проверка подписи для модифицированной команды (подмена
оператора)
```

```
        tampered_command = "CMD:OPEN_VALVE_V-110; TIMESTAMP=2025-05-24 12:00:00; OPERATOR=Petrov"
        print("\n" + "-"*40)
        print("ПРОВЕРКА 2: Изменённая команда (подмена оператора Ivanov → Petrov)")
        print(f"Изменённая команда: {tampered_command}")
        is_valid_tampered = verify_command(tampered_command, signature, public_key)
        print(f"Результат проверки подписи: {'УСПЕШНО' but 'НЕУСПЕШНО'}[{is_valid_tampered}]")

        if not is_valid_tampered:
            print("  >> Подпись НЕДЕЙСТВИТЕЛЬНА! Обнаружена несанкционированная модификация команды.")

        # 6. Проверка подписи с изменением времени (метка)
        tampered_time_command = "CMD:OPEN_VALVE_V-110; TIMESTAMP=2025-05-24 15:00:00; OPERATOR=Ivanov"
        print("\n" + "-"*40)
        print("ПРОВЕРКА 3: Изменённая команда (подмена временной метки)")
        print(f"Изменённая команда: {tampered_time_command}")
        is_valid_time = verify_command(tampered_time_command, signature, public_key)
        print(f"Результат проверки подписи: {'УСПЕШНО' if is_valid_time else 'НЕУСПЕШНО'}")

        if not is_valid_time:
            print("  >> Подпись НЕДЕЙСТВИТЕЛЬНА! Любое изменение команды разрушает подпись.")

        # 7. Демонстрация лавинного эффекта
        demonstrate_avalanche_effect(original_command)

        # 8. Дополнительно: вычисление хеша другим алгоритмом (для сравнения)
        print("\n" + "="*60)
        print("СРАВНЕНИЕ ХЕШ-АЛГОРИТМОВ")
        print("="*60)
        md5_hash = hashlib.md5(original_command.encode('utf-8')).hexdigest()
        sha1_hash = hashlib.sha1(original_command.encode('utf-8')).hexdigest()
        sha256_hash = hashlib.sha256(original_command.encode('utf-8')).hexdigest()
        print(f"MD5:    {md5_hash}")
        print(f"SHA-1:  {sha1_hash}")
        print(f"SHA-256:{sha256_hash}")
        print("\nВывод: SHA-256 рекомендован для использования в АСУ ТП (устойчив к коллизиям).")

        print("\n" + "="*60)
        print("ЗАВЕРШЕНИЕ РАБОТЫ")
        print("="*60)

    if __name__ == "__main__":
        main()
```

4. Результаты выполнения программы

(Скриншот консоли с выводом)

5. Выводы

1. Обеспечение целостности и аутентичности команды

В ходе выполнения лабораторной работы была реализована система электронной подписи технологической команды диспетчера. Экспериментально подтверждено, что:

1. Оригинальная команда, подписанная закрытым ключом диспетчера, успешно проходит проверку открытым ключом на стороне исполнительного устройства.
2. Любое изменение команды (подмена имени оператора Ivanov → Petrov или изменение временной метки) приводит к недействительности подписи. Это доказывает, что электронная подпись обеспечивает целостность (обнаружение модификации) и аутентичность (подтверждение авторства диспетчера).

2. Лавинный эффект хеш-функций

В программе наглядно продемонстрирован лавинный эффект: изменение одного символа (добавлена точка в конце строки) привело к полностью отличающемуся хешу SHA-256. Это свойство критически важно для криптографии, так как даже минимальное изменение исходных данных делает хеш (а следовательно, и электронную подпись) совершенно иным, что исключает возможность незаметной подмены команды.

3. Преимущества электронной подписи перед хешированием

Хеш-функция сама по себе (например, SHA-256) позволяет только проверить, что данные не изменились (если известен эталонный хеш). Однако она не решает проблему доверия: злоумышленник может заменить и данные, и хеш одновременно. Электронная подпись решает эту проблему, так как подпись вычисляется с использованием закрытого ключа, известного только легитимному диспетчеру. Открытый ключ позволяет любому исполнительному устройству проверить подпись, но никак не позволяет создать корректную подпись без закрытого ключа. Это даёт также неотказуемость – диспетчер не может отказаться от подписанной им команды.

4. Применимость в электроэнергетике

В реальных АСУ ТП подстанций, ТЭС, ГЭС и АЭС электронная подпись команд диспетчера (или двойная подпись – диспетчер + главный инженер) применяется для предотвращения несанкционированного управления выключателями, регуляторами и устройствами релейной защиты. Полученные навыки могут быть использованы при реализации защищённых каналов связи по стандарту IEC 62351, где обязательна аутентификация каждого сообщения.

Контрольные вопросы (для защиты)

1. Коллизия хеш-функции – ситуация, когда двум разным сообщениям соответствует одинаковый хеш. MD5 и SHA-1 имеют известные коллизии, поэтому не рекомендуются. ФСТЭК рекомендует SHA-256 и ГОСТ Р 34.11-2012 (Стрибог).
2. Лавинный эффект – изменение одного бита входных данных приводит к изменению примерно 50% битов хеша. В программе это показано на примере добавления точки.
3. Разница между ЭП и асимметричным шифрованием: при ЭП закрытый ключ используется для подписи (шифрования хеша), а открытый – для проверки. При асимметричном шифровании наоборот: открытым ключом шифруют данные, закрытым – расшифровывают. Один и тот же закрытый ключ не рекомендуется использовать и для подписи, и для шифрования (разные режимы padding и требования).

4. Соль (salt) – случайная строка, добавляемая к паролю перед хешированием, чтобы одинаковые пароли давали разные хеши. Это защищает от атак по радужным таблицам.

5. Replay-атака – повторная передача перехваченной корректной команды. В варианте 7 предотвращается включением временной метки в подписываемое сообщение.

6. Обновление прошивок РЗА – подписанные прошивки исключают установку вредоносного ПО. Пример атаки: Industroyer/CrashOverride (2016) мог отключать выключатели через уязвимости протоколов, но если бы прошивки проверяли ЭП – установка вредоносного кода была бы невозможна.

7. ГОСТ Р 34.10-2012 (эллиптические кривые) даёт длину подписи 64 байта (для 256-битной кривой) против 256 байт у RSA-2048, что критично для контроллеров с ограниченной памятью и пропускной способностью каналов.