МИНИСТЕРСТВО НАУКИ, ВЫСШЕГО ОБРАЗОВАНИЯ И ИННОВАЦИЙ КР
КЫРГЫЗСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ им. И.АРАБАЕВА
ОСПО ИНСТИТУТА НОВЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

«УТВЕРЖДАЮ»
Директор ИНИТ
КГУ им. И. Арабаева
к.т.н., и. о. доц. Керимов У.Т.
«04» 09 2025г.

УЧЕБНО-МЕТОДИЧЕСКИЙ КОМПЛЕКС

по дисциплине: **«Информационная безопасность»**

для студентов специальности: 230109 «Программное обеспечение вычислительной техники и автоматизированных систем», 230701 «Прикладная информатика (по отраслям)», 220206 «Автоматизированные системы обработки информации и управления (по отраслям)»

форма обучения: очное/заочное

Учебно-методический комплекс составлен на основе Государственного Образовательного Стандарта среднего профессионального образования КР

Учебно-методический комплекс разработала: магистр-преподаватель отделения СПО ИНИТ КГУ имени И. Арабаева Нурлан кызы Айжамал

Бишкек 2025г.

ОГЛАВЛЕНИЕ

МИНИСТЕРСТВО НАУКИ, ВЫСШЕГО ОБРАЗОВАНИЯ И ИННОВАЦИЙ КР
КЫРГЫЗСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ им. И.АРАБАЕВА
ОСПО ИНСТИТУТА НОВЫХ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

«УТВЕРЖДАЮ»
Директор ИНИТ
КГУ им. И. Арабаева
к.т.н., и. о. доц. У.Т. Керимов
«04» 09 2025г.

РАБОЧАЯ ПРОГРАММА

по дисциплине: **«Информационная безопасность»**
для студентов специальности: 230109 «Программное обеспечение вычислительной техники и автоматизированных систем», 230701 «Прикладная информатика (по отраслям)», 220206 «Автоматизированные системы обработки информации и управления (по отраслям)».

форма обучения: очное
институт: ИНИТ
отделение: ОСПО ИНИТ
курс: 3
семестр: 5/6
аттестация (семестр): 5
экзамен (семестр): 6
всего часов по учебному плану: 60/60
из них:
-лекции:22/22
-практические: 14/14
-самостоятельная работа: 24/24

Рабочая программа составлена в соответствии с требованиями Государственного Образовательного Стандарта среднего профессионального образования КР
Рабочую программу разработала: магистр-преподаватель отделения СПО ИНИТ КГУ имени И. Арабаева Нурлан кызы Айжамал

Рассмотрена и утверждена на заседании
ОСПО ИНИТ КГУ им. И. Арабаева
Протокол № 1
от «02» 09 2025г.

Зав. отделением: Н.С. Сейтказиева

Одобрено учебно-методическим советом
ИНИТ КГУ им. И. Арабаева
Протокол № 1
от «04» 09 2025г.

Председатель УМС ИНИТ:

Бишкек 2025г.

3

1. Цели и задачи изучения дисциплины, ее значение в учебном процессе

1.1. Цели дисциплины

- Понимание студентами роли и перспектив развития информационных процессов и информатизации общества;
- Ознакомление студентов и студентов с тенденцией развития информационной безопасности, с моделями возможных угроз, терминологией и основными понятиями теории безопасности информации, а также с нормативными документами России по данному вопросу и правилами получения соответствующих лицензий.

1.2. Задачи изучения дисциплины

- Проектирование политики информационной безопасности в профессиональной компьютеризированной среде;
- Приобретение практических навыков работы с современными функционально-ориентированными программными средствами защиты информации и использования сетевых ресурсов.

В результате освоения дисциплины студент должен:

Знать

- Основы информационной безопасности
- роли и перспективы развития информационных процессов и информатизации общества;
- тенденции развития информационной безопасности;
- модели возможных угроз;
- терминологию и основные понятия теории безопасности информации;
- нормативные документы Кыргызстана по данному вопросу и правила получения соответствующих лицензий.

Уметь

- проектировать политику информационной безопасности в профессиональной компьютеризированной среде;
- самостоятельно устанавливать антивирусные программы для защиты информации;

Владеть:

- работать с современными функционально-ориентированными программными средствами защиты информации и использования сетевых

Изучение дисциплины формирует знания и навыки, необходимые специалистам по защите информации и администраторам локальных сетей.

2. Компетенции по Госстандарту.

Выпускник в соответствии с целями основной профессиональной образовательной программы и задачами профессиональной деятельности, указанными в пунктах 11 и 15 настоящего Государственного образовательного стандарта, должен обладать следующими компетенциями:

а) общими (ОК):

ОК-1.	Уметь организовывать собственную деятельность, выбирать методы и способы выполнения профессиональных задач, оценивать их эффективность и качество.
ОК-2.	Решать проблемы, принимать решения в стандартных и нестандартных ситуациях, проявлять инициативу и ответственность.
ОК-3	Осуществлять поиск, интерпретацию и использование информации, необходимой для эффективного выполнения профессиональных задач, профессионального и личностного развития.
ОК-4	Использовать информационно – коммуникационные технологии в профессиональной деятельности.
ОК-5	Уметь работать в команде, эффективно общаться с коллегами, руководством, клиентами.
ОК-6	Брать ответственность за работу членов команды (подчиненных), за результат выполнения заданий.
ОК-7	Управлять собственным личностным и профессиональным развитием, адаптироваться к изменениям условий труда и технологий в профессиональной
ОК-8	Быть готовым к организационно – управленческой работе с малыми коллективами.
ОК-9	Способен приобретать новые знания, с большой степенью самостоятельности, с использованием современных образовательных и информационных технологий.
ОК-10	Способен на научной основе оценить свой труд, оценивать с большой степенью самостоятельности, результаты своей деятельности.

220206 – “Автоматизированные системы обработки информации и управление (по отраслям)”

б) профессиональными, соответствующими основным видам профессиональной деятельности (ПК):

производственно-технологическая деятельность:

ПК-4	способен использовать методы конструирования программного обеспечения;
ПК-7	способен применять инструментальные средства к проектированию, моделированию программных продуктов;

230701 – «Прикладная информатика (по отраслям)»

ПК-6	осуществлять сбор и анализ информации для определения потребностей клиента;
ПК-12	осуществлять продвижение и презентацию программного обеспечения отраслевой направленности;

3. Межпредметные связи. Перечень дисциплин и их разделов, усвоение которых необходимо при изучении данной дисциплины

Пререквизиты:

№	Названия дисциплин
1	Компьютерные сети
2	ОС и среды
3	Технические средства информатизации

Постреквизиты:

№	Названия дисциплин
1	Программное обеспечение компьютерных сетей
2	База данных

Структура и трудоемкость дисциплины

Вид работы, семестр	Трудоемкость, час	
	очное обучение	заочное обучение
№ семестров	5/6	
кредит		
Общая трудоемкость	60/60	
Аудиторная работа	36/36	
Лекции	22/22	
Практические занятия/семинары	14/14	
Лабораторные работы		
Самостоятельная работа	24/24	
Курсовые работы или проекты *(при наличии)*		
Рефераты *(при наличии)*		
Внеаудиторные самостоятельные работы *(расчетно-графические задания, типовые расчеты, и т.д.)*		
Самоподготовка *(самостоятельное изучение теоретического материала, подготовка к практическим занятиям, текущему контролю и т.д.)*		
Виды текущего контроля *(перечислить)*		
Вид итогового контроля	Аттестация-5 сем экзамен – 6 сем.	

4. Структура дисциплины с разбивкой по видам занятий, часам и модулям

№	Тематика лекционных занятий	Лекции	Примечание
5-семестр			
1	**Лекция 1**. Маалыматты коргоо. Маалыматтык коопсуздукка киришүү/ Защита информации. Введение в информационную безопасность	2	
2	**Лекция 2** Маалымат коопсуздугуна коркунучтар (МК): Түрлөрү жана моделдери/ Угрозы в информационной безопасности (ИБ): Типы и модели	2	
3	**Лекция 3** Маалыматты коргоо ыкмалары: Паролдор жана ПИН коддор/ Методы защиты информации: Пароли и Пин-коды.	2	
4	**Лекция 4** Идентификация, аутентификация жана авторизация: санариптик дүйнөдө коопсуздуктун үч негизи/ Идентификация, аутентификация и авторизация: Три столпа безопасности в цифровом мир	2	
5	**Лекция 5** Вирустар жана зыяндуу программалар/ Вирусы и вредоносные программы	2	
Итого по модулю №1		10	
6	**Лекция 6** Антивирустук программалардын түрлөрү жана орнотуу/ Типы и установка антивирусных программ	2	
7	**Лекция 7** Фишинг жана анын түрлөрү/ Фишинг и его типы.	2	
8	**Лекция 8** Киберкоопсуздуктук негиздери/ Основы кибербезопасности.	2	
9	**Лекция 9** Wireshark: Негиздери жана программаны орнотуу/ Wireshark: Основы и установка программы.	2	
10	**Лекция 10** Маалыматты коргоонун криптографиялык ыкмалары/ Криптографические методы защиты информации	2	
11	**Лекция 11** Активдүү аудит. Бүтүндүктү көзөмөлдөө/ Активный аудит. Контроль целостности	2	
Итого по модулю №2		**12**	
Итого за 5-семестр		**22**	
6-семестр			

7	**Лекция 1.** Понятия криптография, криптология и криптоанализ.	2	
1	**Лекция 2.** Хэш-функция Криптографические протоколы	2	
2	**Лекция 3.** Электронная цифровая подпись (ЭЦП)	2	
3	**Лекция 4.** Методы и приемы обеспечения информационной безопасности	2	
4	**Лекция 5.** Фишинг. Виды фишинга	2	
5	**Лекция 6.** Технология Блокчейн и ее применение.	2	
6	**Лекция 7.** Основы кибербезопасности Психологические аспекты информационной безопасности	2	
8	**Лекция 8.** Безопасность в сети интернет (Угрозы)	2	
Итого по модулю №1		16	
9	**Лекция 9.** Безопасность в сети интернет (Защита)	2	
10	**Лекция 10.** Вредоносные программы	2	
11	**Лекция 11.** Будущее информационной безопасности	2	
Итого по модулю №2		**6**	
Итого за 6-семестр		**22**	

№	**Тематика практических занятий**	**Кол-во часов**	**Примечание**
1	Защита информации, антивирусная защита	2	
2	Использование методов замены для шифрования данных	2	
3	Асимметричные алгоритмы	2	
4	Использование методов перестановки для шифрования данных.	2	
5	Методы криптоанализа классических шифров	2	
6	Защита информации с помощью пароля	2	
7	Криптографические методы защиты информации	2	
Итого 5 семестр		**14**	
1	Простое шифрование и дешифрование - шифр цезаря	2	
2	Симметричные криптосистемы	2	
3	Создание и настройка виртуальной машины virtualbox на пк с операционной системой ubuntu	2	
4	Анализ и защита виртуальной среды: установка и тестирование антивируса в Windows 10 под VirtualBox		
5	Защита данных, управление доступом и анализ безопасности в windows 10	2	

6	Хеширование	2	
7	Аутентификация пользователей web-систем средствами технологии php	2	
Итого 6 семестр		**14**	

5. Темы заданий СРС по дисциплине

Формы обучения кол-во часов	Задания для СРС
5 семестр	
СРС (2 ч.)	Защита информации и информационная безопасность в структуре профессиональной подготовки специалиста библиотечно-информационной деятельности
СРС (2 ч.)	Основные направления государственной политики в сфере информационной безопасности
СРС (2ч.)	Основные угрозы компьютерной безопасности при работе в сети Интернет
СРС (2ч.)	Основные параметры и черты информационной компьютерной преступности
СРС (2ч.)	Организационное обеспечение информационной безопасности
СРС (2ч.)	Проблемы защиты интеллектуальной собственности средствами патентного и авторского права
СРС (2ч.)	Информационная безопасность личности и общества: современные проблемы цивилизации
СРС (2ч.)	Криптографические методы защиты информации
СРС (2ч.)	Понятие шифра
СРС (2ч.)	Принципы защиты информации
СРС (2ч.)	Условия, сопутствующие утечки информации
СРС (2ч.)	Состав угроз информационной безопасности
24ч	ИТОГО
6 семестр	
СРС (2 ч.)	Защита информации и информационная безопасность в структуре профессиональной подготовки специалиста библиотечно-информационной деятельности
СРС (2 ч.)	Основные направления государственной политики в сфере информационной безопасности
СРС (2ч.)	Основные угрозы компьютерной безопасности при работе в сети Интернет
СРС (2ч.)	Основные параметры и черты информационной компьютерной преступности
СРС (2ч.)	Организационное обеспечение информационной безопасности
СРС (2ч.)	Проблемы защиты интеллектуальной собственности средствами патентного и авторского права
СРС (2ч.)	Информационная безопасность личности и общества: современные проблемы цивилизации
СРС (2ч.)	Криптографические методы защиты информации
СРС (2ч.)	Понятие шифра
СРС (2ч.)	Принципы защиты информации
СРС (2ч.)	Условия, сопутствующие утечки информации
СРС (2ч.)	Состав угроз информационной безопасности
24ч	ИТОГО

6. Распределение баллов по модулям и видам учебных занятий

№	Этапы проверки	Вид средства проверки	Баллы
1	1 модуль	Проверка практических заданий. Устный, тестирование. Посещение занятий.	100
2	2 модуль	Проверка практических заданий. Тестирование. Посещение занятий.	100
3	Итоговый контроль: • Практическое занятие; • СРС.	Контрольные и графические работы, рефераты, презентации, СРС, практические задания. Тестирование. Посещение занятий.	100
	Итого средний балл		**100**

Итоговое распределение баллов по модулям

		Удовлетворительно	Хорошо	Отлично
Модуль 1 – 100 б.		60-79	80-89	90-100
Модуль 2 – 100 б.		60-79	80-89	90-100
Практическое занятие – 50 б.	Итоговый контроль	60-79	80-89	90-100

7. Список основной и дополнительной литературы

№	Библиографическое описание издания (автор, наименование, вид, место и год издания, кол.стр.)	Виды занятия в котором используется	Эл.книги
Основная литература			
1	Прохорова О. В. Информационная безопасность и защита информации : учебник для СПО / О. В. Прохорова. — 2е изд., стер. — СанктПетербург : Лань, 2021. — 124 с.	Лекция	
2	Безопасность веб-приложений на Python / пер. с англ. С. С. Скобелева, А. Н. Киселева. – М.: ДМК Пресс, 2023. – 334 с.	Лекция Практика	
3	П. Торстейнсон, Г. А. Ганеш КРИПТОГРАФИЯ И БЕЗОПАСНОСТЬ В ТЕХНОЛОГИИ .NET Перевод с английского В. Д. Хорева под редакцией С. М. Молявко 4е издание, электронное Москва Лаборатория знаний 2020-482 стр	Практика	Let's Start!
4	Солодов А.В, Мунистер В.Д. «Экономика знаний. Блокчейн и умные контракты» 2021 г. -128с	Практика	
Дополнительная литература			
1	"Криптография и безопасность" (А.Ю. Зубов)— Русскоязычная книга по алгоритмам шифрования. 🔗 PDF на rulit.me		
2	Hacking: The Art of Exploitation" (Jon Erickson)— Классика по этичному хакингу и эксплуатации уязвимостей. 🔗 PDF на archive.org.		
1	**Сайты для скачивания и чтения онлайн книг:**		

1. **Google Books**
2. **Internet Archive**
3. **OpenLibra**
4. **Project Gutenberg**
5. **PDF Drive**
6. **BookFi**

Сайты для бесплатных курсов:

1. **Coursera**
2. **edX**
3. **Udemy**
4. **Khan Academy**
5. **Cybrary**
6. **MIT OpenCourseWare**

8. Вопросы (тесты) к модулям

№		Перечень вопросов (тестов)
1	Модуль 1	
2	Модуль 2	
	Ключ (Правильные ответы)	

8.1. Примерные вопросы к экзамену по дисциплине

	Темы
1	Основные понятия информационной безопасности.
2	Информационные технологии и необходимость ИБ.
3	Система защиты информации и ее структуры.
4	Профессиональные тайны, их виды. Объекты коммерческой тайны на предприятии.
5	Персональные данные и их защита.
6	Информационные угрозы, их виды и причины возникновения.
7	Информационные угрозы для государства.
8	Информационные угрозы для компании.
9	Информационные угрозы для личности (физического лица).
10	Действия и события, нарушающие информационную безопасность.
11	Личностно-профессиональные характеристики и действия сотрудников, способствующих реализации информационных угроз.
12	Способы воздействия информационных угроз на объекты.
13	Внешние и внутренние субъекты информационных угроз.
14	Компьютерные преступления и их классификация.
15	Исторические аспекты компьютерных преступлений и современность.
16	Субъекты и причины совершения компьютерных преступлений.
17	Вредоносные программы, их виды.
18	История компьютерных вирусов и современность.
19	Деятельность международных организаций в сфере информационной безопасности.
20	Задачи ИБ в программе «цифровая экономика».
21	Политика безопасности и ее принципы.
22	Фрагментарный и системный подход к защите информации.
23	Методы и средства защиты информации.
24	Организационное обеспечение ИБ.
25	Организация конфиденциального делопроизводства.
26	Организационно-экономическое обеспечение ИБ.
27	Инженерно-техническое обеспечение компьютерной безопасности.
28	Организационно-правовой статус службы безопасности.
29	Защита информации в Интернете.
30	Электронная почта и ее защита.
31	Защита от компьютерных вирусов.
32	Популярные антивирусные программы и их классификация.
33	Этапы и освоение защиты информации экономических объектов.
34	Криптографические методы защиты информации.
35	Оценка эффективности инвестиций в информационную безопасность.
36	Фирмы, оценивающие работу персонала в компании.
37	Менеджмент и аудит ИБ на уровне предприятия.
38	Аудит ИБ автоматизированных банковских систем.
39	Аудит ИБ электронной коммерции.
40	Информационная безопасность предпринимательской деятельности.

9. Методическая разработка аудиторных форм работы

9.1. Курс лекционных занятий

Лекция 1. Хэш-функции и ЭЦП

Хэш-функция — это математическая функция, которая преобразует произвольный вход (например, данные любого размера) в строку фиксированной длины, называемую хэш-значением или дайджестом. В криптографии хэш-функции используются для различных целей, таких как обеспечение целостности данных и их аутентификации. Важно, чтобы хэш-функция обладала рядом свойств:

1. **Детерминированность**: Для одного и того же входа хэш-функция всегда должна давать одинаковый результат.
2. **Односторонность**: Из хэш-значения невозможно получить исходные данные.
3. **Быстродействие**: Хэш-функция должна быстро вычисляться.
4. **Устойчивость к коллизиям**: Невозможно найти два разных входа, которые дают одинаковый хэш.
5. **Устойчивость к предобразованию**: Не должно быть возможности предсказать, какой хэш будет у конкретных данных, на основе других значений хэшей.

Примеры криптографических хэш-функций: SHA-256, MD5, SHA-3.

Криптографические протоколы — это наборы правил и процедур, которые обеспечивают безопасное взаимодействие между сторонами, защищая данные от несанкционированного доступа, изменений и подделки. Эти протоколы используются для различных целей: для аутентификации пользователей, шифрования данных, обмена ключами и обеспечения конфиденциальности.

Некоторые важные криптографические протоколы:

1. **SSL/TLS** — протоколы для безопасного обмена данными по сети (например, для HTTPS). Они используют шифрование для защиты данных и обеспечивают аутентификацию серверов.
2. **Diffie-Hellman** — протокол обмена криптографическими ключами, который позволяет двум сторонам безопасно обменяться секретными ключами через небезопасный канал.
3. **RSA** — криптографический алгоритм с открытым ключом, используемый для шифрования и цифровых подписей.
4. **IPsec** — протокол для обеспечения безопасности на уровне сетевого интерфейса, используется для защиты данных в виртуальных частных сетях (VPN).
5. **Kerberos** — протокол аутентификации, который используется для подтверждения личности пользователей в компьютерных сетях.

Лекция 2. Электронная цифровая подпись (ЭЦП)

Электронная цифровая подпись (ЭЦП) — это криптографический механизм, используемый для обеспечения подлинности и целостности электронных документов и данных. ЭЦП работает как аналог традиционной подписи, но в цифровом виде и с использованием криптографических технологий. ЭЦП позволяет удостовериться в том, что документ был подписан конкретным лицом, а также что его содержимое не было изменено после подписания.

Основные принципы работы ЭЦП:

1. **Генерация ключей:** Для создания ЭЦП используется пара криптографических ключей:
    - **Приватный (секретный) ключ** — используется для подписания документа.
    - **Публичный ключ** — используется для проверки подписи. Этот ключ может быть доступен всем.
2. **Подписание документа:** Чтобы подписать документ, его хэш (с помощью хэш-функции) вычисляется и затем этот хэш шифруется с использованием приватного ключа подписанта. Полученная зашифрованная строка и будет являться ЭЦП документа.
3. **Проверка подписи:** При проверке подписи:
    - Получается хэш подписанного документа.
    - ЭЦП расшифровывается с помощью публичного ключа подписанта.
    - Если расшифрованный хэш совпадает с хэшем документа, то подпись считается действительной. Это подтверждает, что документ не был изменен и что подпись действительно принадлежит указанному подписанту.

Свойства ЭЦП:

1. **Подтверждение подлинности:** Только тот, кто обладает приватным ключом, может подписать документ, и любой другой пользователь может проверить подлинность подписи с помощью публичного ключа.
2. **Целостность:** Если содержимое подписанного документа изменится, проверка подписи не пройдет, что подтверждает, что данные были изменены.
3. **Неотказуемость:** Подписав документ с помощью ЭЦП, подписант не может впоследствии отказаться от своей подписи, так как она ассоциируется с уникальной парой ключей и может быть проверена.

Применение ЭЦП:

- **Электронные контракты**: ЭЦП используется для подписания юридически значимых документов в электронном виде, что упрощает и ускоряет процессы.
- **Государственные услуги**: В различных странах ЭЦП используется для удостоверения личных данных при взаимодействии с государственными органами (например, при подаче налоговых деклараций, регистрации юридических лиц и т. д.).
- **Электронные платежи и транзакции**: ЭЦП обеспечивает безопасность транзакций в электронной коммерции и онлайн-банкинге.
- **Почтовые и нотариальные услуги**: ЭЦП может использоваться для подтверждения подлинности отправленных электронных писем или для нотариального заверения электронных документов.

Пример использования:

Предположим, вы подписываете договор с контрагентом. Когда вы подписываете его с помощью своей ЭЦП, это гарантирует, что:

1. Договор был подписан именно вами (так как только у вас есть ваш приватный ключ).
2. Документ не был изменен после подписания (проверяется с помощью хэш-функции и подписи).
3. Контрагент может быть уверен в подлинности подписи, проверив ее с помощью вашего публичного ключа.

Лекция 3. Методы и приемы обеспечения информационной безопасности

Методы и приемы обеспечения информационной безопасности (ИБ) включают в себя комплекс мер, направленных на защиту информации от различных угроз, таких как утечка данных, несанкционированный доступ, уничтожение или повреждение информации. Эти методы могут быть как техническими, так и организационными, а также правовыми.

Основные методы и приемы обеспечения информационной безопасности:

1. **Шифрование**: Использование криптографических методов для защиты данных от несанкционированного доступа. Это включает в себя как шифрование данных при их передаче (например, через интернет), так и при хранении.
2. **Аутентификация и авторизация**: Методы, которые позволяют проверять личность пользователя (например, через пароли, биометрические данные, токены) и контролировать его доступ к ресурсам на основе установленных прав.
3. **Мониторинг и аудит безопасности**: Системы для отслеживания активности в информационных системах, которые позволяют выявить подозрительные действия и нарушителей безопасности.
4. **Физическая безопасность**: Защита физических устройств (серверов, ПК, мобильных устройств) от физического воздействия (например, кражи или повреждения).
5. **Резервное копирование и восстановление**: Создание копий данных, чтобы в случае потери или повреждения информации можно было восстановить её из резервных источников.
6. **Обучение сотрудников**: Проведение регулярных тренингов и обучения для повышения осведомленности сотрудников о рисках и методах защиты информации.
7. **Защита от вредоносных программ**: Установка антивирусных программ, файрволов и систем предотвращения вторжений (IPS) для защиты от вирусов, троянов и других видов вредоносного ПО.

Правовые аспекты информационной безопасности

Вопросы правового регулирования информационной безопасности охватывают несколько ключевых областей, таких как:

1. **Правовая ответственность**: Нарушения в области ИБ могут привести к уголовной или административной ответственности. Например, в случае несанкционированного доступа к данным, распространения вредоносных программ или утечки личных данных.
2. **Защита персональных данных**: В большинстве стран существуют законы, защищающие права граждан на конфиденциальность и защиту их персональных данных. Компании и государственные органы обязаны защищать эти данные, а также соблюдать требования о уведомлении субъектов данных о сборе и обработке их информации.
3. **Киберпреступления**: Все больше государств разрабатывают законодательство, направленное на борьбу с киберпреступностью, включая кибершпионаж, кибератаки, фишинг и другие угрозы.
4. **Интернациональное сотрудничество**: Международное сотрудничество в области ИБ основывается на правовых соглашениях и договорах, которые обеспечивают защиту данных и взаимодействие между государствами.

Кыргызское законодательство в области информационной безопасности

Кыргызстан имеет несколько законов и нормативных актов, направленных на обеспечение информационной безопасности:

1. **Закон Кыргызской Республики «Об информации» (2008)**: Этот закон регулирует вопросы, связанные с доступом к информации, её защитой и использованием, а также установление ответственности за нарушение прав на информацию.
2. **Закон Кыргызской Республики «О защите персональных данных» (2017)**: Этот закон регулирует сбор, обработку и защиту персональных данных граждан, а также устанавливает права субъектов персональных данных и обязанности операторов.
3. **Закон о кибербезопасности (в разработке)**: В Кыргызстане активно разрабатывается законодательство в области кибербезопасности, которое будет охватывать защиту критической инфраструктуры, борьбу с киберпреступностью и защиту данных в интернете.
4. **Конституция Кыргызской Республики** также регулирует вопросы информационной безопасности, обеспечивая защиту гражданских прав и свобод, включая право на информацию и её защиту.
5. **Нормативные акты государственных органов**: Важными элементами регулирования ИБ являются акты, принимаемые государственными органами, например, **Государственной службой защиты информации** и **Государственным комитетом связи и информационных технологий**.

Международные стандарты и соглашения в области информационной безопасности

1. **ISO/IEC 27001** — международный стандарт для управления информационной безопасностью. Он описывает требования к созданию, внедрению, поддержке и улучшению системы управления информационной безопасностью (СУИБ).
2. **ISO/IEC 27002** — стандарт, который предоставляет практические рекомендации для применения наилучших практик в области управления информационной безопасностью.
3. **GDPR (General Data Protection Regulation)** — Регламент Европейского Союза, направленный на защиту персональных данных. Он регулирует сбор, обработку и хранение данных граждан ЕС и устанавливает строгие требования к их защите.
4. **Конвенция Совета Европы о киберпреступности (Будапештская конвенция)** — первый международный юридически обязательный акт, направленный на борьбу с киберпреступлениями, включающий вопросы преступлений, связанных с компьютерными системами и данными.
5. **NIST Cybersecurity Framework** — Рамочная модель по обеспечению кибербезопасности, разработанная Национальным институтом стандартов и технологий США. Этот стандарт используется во многих странах для управления рисками в области ИТ-безопасности.
6. **Киотский протокол** и **Парижская декларация** также касаются вопросов защиты информации в контексте международных соглашений по защите окружающей среды и других областей, требующих безопасности данных.

Лекция 4. Фишинг. Виды фишинга

Фишинг — это метод киберпреступности, при котором злоумышленники пытаются обманом получить конфиденциальную информацию (например, логины, пароли, данные банковских карт, персональные данные) у пользователей. Обычно это достигается с помощью поддельных веб-

сайтов или сообщений, которые выглядят как официальные и доверенные источники. Фишинг является одной из самых популярных форм социального инженерства.

Основные виды фишинга:

1. **Классический фишинг (Email фишинг)**:
   - Это один из самых распространенных типов фишинга. Злоумышленники отправляют пользователю поддельное письмо, которое выглядит как сообщение от известной компании (например, банка, онлайн-магазина или социальной сети). В письме обычно содержится ложный запрос, например, просьба подтвердить личные данные, обновить пароль или подтвердить транзакцию. Часто в письме будет присутствовать ссылка на фальшивую веб-страницу, которая выглядит как настоящая, но является подделкой.
   - **Пример**: Письмо от "Ваш банк" с сообщением, что ваш аккаунт был заблокирован, и вам нужно перейти по ссылке для его восстановления.
2. **Смс-фишинг (SMiShing)**:
   - В этом случае фишинг осуществляется через SMS-сообщения. Злоумышленники отправляют текстовое сообщение с ложным уведомлением, которое может содержать ссылку на фальшивую страницу или просьбу отправить личные данные. SMiShing используется для кражи паролей, банковских данных или даже для заражения устройств вредоносными программами.
   - **Пример**: Смс-сообщение с уведомлением о выигрыше, в котором предлагается перейти по ссылке для получения приза или вознаграждения.
3. **Вишинг (Voice phishing)**:
   - Вишинг — это фишинг через телефонные звонки. Злоумышленники могут позвонить и представиться сотрудниками банка, службы безопасности или другого учреждения, чтобы выманить конфиденциальную информацию. Во время звонка они могут попросить пользователя сообщить пароли, номера карт или другую личную информацию.
   - **Пример**: Звонок с "представителя банка", который сообщает, что ваш счет был скомпрометирован, и для его восстановления требуется предоставить код безопасности.
4. **Фишинг через социальные сети (Social Media Phishing)**:
   - В этом случае злоумышленники используют социальные сети (например, Facebook, Instagram, LinkedIn) для связи с жертвами. Они могут создать фальшивый профиль, чтобы выманить личную информацию или запрашивать доступ к аккаунту через поддельные страницы или сообщения.
   - **Пример**: Поддельные объявления о предложениях вакансий, лотереях или акциях, которые требуют ввести личные данные или отправить деньги.
5. **Сетевой фишинг (Man-in-the-Middle Phishing)**:
   - Этот тип фишинга использует уязвимости в сетевых соединениях. Злоумышленники могут вмешиваться в коммуникацию между пользователем и сервером (например, на открытых Wi-Fi-сетях), перехватывая пароли или другую личную информацию. Это может происходить без ведома жертвы, пока она не заметит проблему.
   - **Пример**: Человек подключается к общедоступной сети Wi-Fi и вводит личные данные на поддельной странице, созданной злоумышленниками.
6. **Фарминг (Pharming)**:
   - Фарминг — это более сложный вид фишинга, при котором злоумышленники изменяют настройки DNS-серверов, чтобы перенаправить пользователей на фальшивые веб-сайты. Эти сайты выглядят как настоящие, но на самом деле служат для кражи личных данных пользователей.

- **Пример**: При попытке перейти на сайт вашего банка, вы автоматически попадаете на поддельную страницу, которая соблазняет вас ввести свои данные.

7. **Печать фишинга (Quizzes or Surveys Phishing)**:
   - Этот вид фишинга включает создание поддельных опросов или викторин, которые запрашивают личную информацию в обмен на якобы "подарки" или "призы". Вопросы могут касаться как информации о пользователе, так и его аккаунтов в разных сервисах.
   - **Пример**: Викторина в социальных сетях, где нужно ввести личные данные, чтобы получить бесплатный доступ к сервисам или выигрыши.

Как защититься от фишинга:

1. **Проверка URL-адреса**: При переходе по ссылкам всегда проверяйте правильность URL. Например, сайт банка должен начинаться с `https://www.bank.com`, а не с подозрительных доменов.
2. **Не доверять сомнительным письмам**: Если вы получили неожиданные письма или сообщения с просьбой предоставить личные данные или перейти по ссылке, всегда перепроверяйте информацию, связавшись с организацией напрямую.
3. **Использование двухфакторной аутентификации (2FA)**: Это добавляет дополнительный слой защиты, требуя второго подтверждения при входе в аккаунт.
4. **Осторожность при передаче личных данных**: Никогда не отправляйте свои данные через открытые каналы связи или по телефону, если не уверены в надежности источника.
5. **Антивирусные программы и фаерволы**: Использование надежных антивирусных решений и фаерволов помогает защититься от некоторых видов фишинговых атак.

Фишинг — это серьёзная угроза, но с помощью осмотрительности, технологий и грамотного подхода к безопасности можно значительно уменьшить риски.

Лекция 5. Технология Блокчейн и ее применение.

Технология блокчейн — это распределенная база данных, которая позволяет хранить данные в виде цепочки блоков, где каждый блок содержит информацию о предыдущем, создавая тем самым непрерывную, неизменяемую и прозрачную цепь данных. Основной особенностью блокчейна является то, что он позволяет создать безопасную и децентрализованную систему хранения и передачи информации без необходимости в централизованном управляющем органе, например, банке или другом посреднике.

Основные характеристики технологии блокчейн:

1. **Децентрализация**: Блокчейн не имеет центрального управляющего органа. Все данные хранятся на множестве узлов (компьютеров) в сети, что делает систему более устойчивой к сбоям и атакам.
2. **Неизменность**: После того как данные записаны в блокчейн, их нельзя изменить. Чтобы изменить информацию в блоке, нужно изменить все последующие блоки, что требует колоссальных вычислительных ресурсов, что делает фальсификацию данных практически невозможной.
3. **Прозрачность**: Все участники сети могут увидеть все транзакции, происходящие в системе, что делает блокчейн прозрачным и открытым для проверки.

4. **Безопасность**: Каждая транзакция в блокчейне подтверждается с использованием криптографии. Чтобы подтвердить или изменить запись, требуется решение математических задач, что делает систему защищенной от фальсификаций.

Применение технологии блокчейн

Технология блокчейн имеет множество приложений, которые могут значительно изменить различные отрасли и процессы. Вот несколько примеров использования блокчейна:

1. **Криптовалюты**:
   - Самое известное приложение блокчейн-технологии — это криптовалюты, такие как **Bitcoin**, **Ethereum**, **Ripple** и другие. Блокчейн используется для обеспечения децентрализованных платежей и финансовых транзакций, позволяя отправлять и получать деньги без посредников (банков), снижая комиссии и повышая скорость операций.
2. **Смарт-контракты**:
   - **Смарт-контракты** — это программируемые контракты, которые автоматически исполняются при соблюдении заранее установленных условий. Блокчейн позволяет хранить и исполнять эти контракты на децентрализованных платформах, таких как **Ethereum**. Например, смарт-контракт может автоматически перевести средства с одного аккаунта на другой, как только выполнятся определенные условия (например, доставлена товарная партия).
3. **Цифровая идентификация**:
   - Блокчейн может быть использован для создания безопасной и защищенной цифровой идентичности. В странах, где существует система цифровых паспортов, можно использовать блокчейн для аутентификации граждан в различных государственных и частных сервисах, улучшая безопасность и снижая возможность мошенничества.
4. **Управление цепочками поставок**:
   - Блокчейн может быть применен для отслеживания товаров в цепочке поставок. Каждая транзакция или передвижение товара записывается в блокчейн, что позволяет обеспечить прозрачность, улучшить контроль за продуктами и сократить возможности для мошенничества и подделки. Это особенно важно для таких отраслей, как пищевая промышленность, фармацевтика и производство.
5. **Голосование**:
   - Использование блокчейна для голосования позволяет создать безопасную и прозрачную систему для проведения выборов. Все голосования могут быть записаны в блокчейн, что делает процесс менее подверженным фальсификациям, а результаты — легко проверяемыми.
6. **Медицинские записи**:
   - Блокчейн может использоваться для создания безопасных и доступных медицинских записей. Врач или пациент могут получить доступ к медицинским данным через блокчейн, что обеспечивает целостность и защиту личной информации от несанкционированного доступа.
7. **Интеллектуальная собственность и авторские права**:
   - Блокчейн может помочь в управлении авторскими правами на цифровые произведения (музыка, изображения, текст и т. д.). Каждое произведение может быть зарегистрировано на блокчейне, что позволяет автоматизировать выплату авторских вознаграждений, отслеживание использования контента и защиту от нарушения прав.

8. **Децентрализованные приложения (DApps)**:
    - На основе блокчейна можно разрабатывать децентрализованные приложения, которые работают без необходимости в центральных серверах и посредниках. Примером таких приложений является **Decentralized Finance (DeFi)** — финансовые сервисы, работающие без участия банков и финансовых учреждений.
9. **Токенизация активов**:
    - Блокчейн позволяет создавать токены, которые могут представлять собой любую ценность — от физических объектов до цифровых активов. Например, недвижимость может быть токенизирована, и доли в этой недвижимости можно будет покупать и продавать на блокчейн-платформах.
10. **Прогнозирование и страхование**:
    - Блокчейн может использоваться для создания прозрачных и справедливых систем прогнозирования и страхования, в которых смарт-контракты автоматически выполняются на основе заранее прописанных условий. Например, в случае задержки рейса смарт-контракт может автоматически выплатить компенсацию.

Преимущества и недостатки блокчейн-технологии

Преимущества:

- **Безопасность**: Высокий уровень криптографической защиты.
- **Прозрачность**: Все транзакции можно проследить.
- **Снижение издержек**: Уменьшение необходимости в посредниках (банки, нотариусы, бюрократические процессы).
- **Доступность**: Возможность использования для людей, не имеющих доступа к традиционным финансовым системам (например, в странах с развивающейся экономикой).

Недостатки:

- **Энергозатраты**: Некоторые консенсусные алгоритмы (например, Proof of Work) требуют значительных вычислительных мощностей, что ведет к высоким энергетическим затратам.
- **Масштабируемость**: Текущие блокчейн-системы (например, Bitcoin) имеют ограниченную пропускную способность и могут не подходить для высокоскоростных транзакций в реальном времени.
- **Юридические вопросы**: Блокчейн не всегда соответствует законодательству разных стран, особенно в области финансового регулирования, и вызывает вопросы по поводу налогообложения и защиты данных.

Лекция 6. Основы кибербезопасности

Основы кибербезопасности — это совокупность практик, технологий и процедур, направленных на защиту информационных систем, данных и пользователей от различных киберугроз и атак. В условиях глобализации и быстрого роста цифровых технологий кибербезопасность становится важнейшей частью обеспечения безопасности и функционирования любых организаций и личных пользователей.

Основные цели кибербезопасности:

1. **Защита конфиденциальности**: Обеспечение того, чтобы только уполномоченные лица имели доступ к конфиденциальной информации.
2. **Обеспечение целостности данных**: Защита данных от несанкционированных изменений, повреждений или уничтожения.
3. **Обеспечение доступности**: Гарантирование того, что информация и системы будут доступны для пользователей в нужное время.
4. **Аутентификация и авторизация**: Удостоверение личности пользователей и контроль их доступа к ресурсам системы.
5. **Защита от киберугроз**: Противодействие различным киберпреступлениям, включая вирусы, хакерские атаки, утечку данных и другие угрозы.

Основные компоненты кибербезопасности:

1. **Управление рисками**:
   - Оценка и минимизация рисков, связанных с угрозами информационной безопасности. Организации должны регулярно проводить анализ уязвимостей и принимать меры для снижения рисков.
2. **Шифрование**:
   - Использование криптографических методов для защиты данных, как при их хранении, так и при передаче. Это один из важнейших инструментов для защиты конфиденциальной информации и предотвращения её утечек.
3. **Физическая безопасность**:
   - Защита физического доступа к информационным системам, серверам, базам данных и другим компонентам инфраструктуры. Это включает в себя контроль за доступом в серверные помещения, видеонаблюдение, системы контроля доступа и другие меры.
4. **Брандмауэры (фаерволы)**:
   - Защита сети от несанкционированного доступа и атак путем фильтрации входящего и исходящего трафика, блокировки вредоносных попыток подключения.
5. **Антивирусные программы и системы защиты от вредоносного ПО**:
   - Использование антивирусных решений для защиты устройств от вирусов, троянов, шпионских программ, программ-вымогателей и других типов вредоносных программ.
6. **Системы предотвращения вторжений (IDS/IPS)**:
   - Системы для мониторинга и анализа сетевого трафика с целью обнаружения попыток атак и вторжений. Они могут предупреждать об угрозах или автоматически блокировать подозрительную активность.
7. **Резервное копирование данных**:
   - Регулярное создание резервных копий данных и систем для восстановления информации в случае утраты, повреждения или атаки (например, атаки программ-вымогателей).
8. **Обучение пользователей**:
   - Одним из важнейших аспектов кибербезопасности является повышение осведомленности пользователей об угрозах, таких как фишинг, социальная инженерия, слабые пароли и другие. Регулярное обучение сотрудников снижает вероятность ошибок, приводящих к компрометации системы.

Основные угрозы кибербезопасности:

1. **Вредоносные программы (Malware)**:
   - Вирусы, трояны, черви, программы-вымогатели, шпионские программы — это различные виды вредоносных программ, которые могут повреждать или похищать данные, захватывать управление устройствами, шифровать файлы и требовать выкуп.
2. **Фишинг**:
   - Попытка получить конфиденциальную информацию, такую как логины, пароли или данные банковских карт, с помощью поддельных сообщений или веб-сайтов, которые маскируются под официальные. Пример — электронные письма от «банка», в которых предлагается перейти по ссылке и ввести данные.
3. **Атаки типа «отказ в обслуживании» (DDoS)**:
   - Массированная атака, направленная на перегрузку ресурсов системы или сети, что делает её недоступной для пользователей. Это может быть сделано путем отправки огромного количества запросов или пакетов данных.
4. **Социальная инженерия**:
   - Техники манипуляции людьми для получения доступа к конфиденциальной информации или системам. Примеры: фишинг, вишинг (телефонные атаки), поддельные опросы и т.д.
5. **Уязвимости в ПО**:
   - Программные ошибки или недостатки в коде, которые могут быть использованы хакерами для получения несанкционированного доступа или выполнения вредоносных действий. Регулярные обновления и патчи — необходимая мера для защиты от таких угроз.

Важные аспекты кибербезопасности для организаций:

1. **Политики и стандарты безопасности**:
   - Разработка внутренних политик и стандартов безопасности для определения, кто и как может получить доступ к системам и данным, а также какие меры предпринимать для защиты.
2. **Многослойная защита (Defense in Depth)**:
   - Подход, заключающийся в использовании нескольких уровней защиты. Например, комбинирование шифрования, аутентификации, защиты от вирусов и мониторинга трафика.
3. **Контроль доступа**:
   - Определение, кто имеет право доступа к определенным данным или системам. Это может включать использование паролей, биометрических данных, карт доступа, а также технологий двухфакторной аутентификации.
4. **Регулярный аудит безопасности**:
   - Оценка состояния информационной безопасности в организации через тесты на проникновение, анализ уязвимостей, проверку политики безопасности и внутренние проверки.

Советы по улучшению кибербезопасности для пользователей:

1. **Использование надежных паролей**: Создавайте сложные пароли и меняйте их регулярно. Используйте менеджеры паролей для безопасного хранения паролей.

2. **Двухфакторная аутентификация (2FA)**: Включите двухфакторную аутентификацию на всех сервисах, где это возможно, чтобы добавить дополнительный уровень безопасности.
3. **Будьте осторожны с электронной почтой**: Не открывайте подозрительные вложения и ссылки, даже если они пришли от знакомых. Поддерживайте бдительность по отношению к фишинговым письмам.
4. **Регулярно обновляйте ПО**: Обновления часто содержат исправления для уязвимостей, которые могут быть использованы злоумышленниками для проникновения в систему.
5. **Защита от вредоносных программ**: Используйте антивирусные программы, регулярно обновляйте их базы данных и выполняйте проверки на наличие угроз.
6. **Резервное копирование данных**: Создавайте резервные копии важных файлов и данных, чтобы в случае утраты или атаки их можно было восстановить.

Лекция 7. Понятия криптография, криптология и криптоанализ.

Криптография, **криптология** и **криптоанализ** — это взаимосвязанные дисциплины, связанные с защитой информации и её анализом с использованием математических и алгоритмических методов.

1. Криптография

Криптография — это наука о методах защиты информации с использованием математических принципов и алгоритмов, которые позволяют шифровать данные, делать их нечитаемыми для посторонних и обеспечивать их целостность. Криптография охватывает широкий спектр техник и методов, предназначенных для защиты данных от несанкционированного доступа и их подделки.

Основные задачи криптографии:

- **Шифрование**: Преобразование открытого текста (пользовательских данных) в зашифрованный, который невозможно прочитать без ключа.
- **Дешифрование**: Процесс восстановления исходной информации из зашифрованного текста с использованием соответствующего ключа.
- **Цифровая подпись**: Механизм, который позволяет проверить подлинность данных и удостовериться, что они не были изменены.
- **Аутентификация**: Процесс проверки подлинности личности пользователя, устройства или системы.
- **Целостность данных**: Методы проверки того, что данные не были изменены в процессе передачи или хранения.

2. Криптология

Криптология — это более широкая область, которая включает в себя **криптографию** и **криптоанализ**. Криптология охватывает изучение всех аспектов защиты и взлома информации. Это научная дисциплина, которая занимается как разработкой методов защиты информации (криптография), так и методами её взлома (криптоанализ).

Криптология включает в себя:

- Разработку криптографических алгоритмов, которые обеспечивают безопасность.
- Исследование и анализ существующих криптографических методов и попытки их взлома.

- Оценку стойкости криптографических систем и алгоритмов, проверка их устойчивости к атакующим действиям.

3. Криптоанализ

Криптоанализ — это наука и практика взлома или анализа криптографических систем и алгоритмов с целью найти уязвимости, слабые места или способы дешифровки данных без знания ключа. Криптоанализ помогает определить, насколько надёжна криптографическая система и насколько хорошо она защищает данные от несанкционированного доступа.

Основные цели криптоанализа:

- **Нахождение уязвимостей**: Криптоаналитики изучают криптографические алгоритмы на предмет ошибок или слабых мест, которые могут быть использованы для взлома.
- **Взлом алгоритмов**: Применение методов для восстановления исходных данных (например, ключей или сообщений) без знания секретной информации.
- **Оценка стойкости алгоритмов**: Криптоанализ помогает оценить степень безопасности системы, выявляя её уязвимости и тестируя методы атаки.

Методы криптоанализа:

- **Брутфорс (перебор)**: Метод, при котором проверяются все возможные варианты ключей до нахождения правильного.
- **Анализ частотности**: Используется для дешифровки шифров, основанных на частотном распределении символов, например, в случае с классическим шифром Цезаря.
- **Криптоаналитические атаки**: Методики, такие как атаки на основе известных открытых текстов, атаки с использованием выбранного текста, атаки на основе времени и другие.

Пример взаимосвязи:

- **Криптография** разрабатывает новые способы защиты информации (например, шифрование данных с использованием алгоритма RSA).
- **Криптология** включает как криптографию (создание систем защиты), так и криптоанализ (поиск уязвимостей в этих системах).
- **Криптоанализ** пытается взломать шифры и криптографические системы, чтобы понять, насколько они безопасны и могут ли они быть обойдены.

Лекция 8. Безопасность в сети интернет (Угразы)

Интернет – это объединенные между собой компьютерные сети, глобальная мировая система передачи информации с помощью информационно-вычислительных ресурсов.

Самые опасные угрозы сети Интернет

Компью́терный ви́рус — разновидность компьютерных программ или вредоносный код, отличительной особенностью которых является способность к размножению (саморепликация).

Классификация

В настоящее время не существует единой системы классификации и именования вирусов. Принято разделять вирусы на следующие группы.

❖ ПО ПОРАЖАЕМЫМ ОБЪЕКТАМ

- Файловые вирусы

Это вирусы-паразиты, которые при распространении своих копий обязательно изменяют содержимое исполняемых файлов, при этом файлы, атакованные вирусом, в большинстве случаев полностью или частично теряют работоспособность)

- Загрузочные вирусы

Это компьютерные вирусы, записывающиеся в первый сектор гибкого или жесткого диска и выполняющиеся при загрузке компьютера.

- Скриптовые вирусы

Требуют наличия одного из скриптовых языков (Javascript, VBScript) для самостоятельного проникновения в неинфицированные скрипты.

- Макровирусы

Это разновидность компьютерных вирусов разработанных на макроязыках, встроенных в такие прикладные пакеты ПО, как Microsoft Office.

- Вирусы, поражающие исходный код

Вирусы данного типа поражают или исходный код программы, либо её компоненты (OBJ-, LIB-, DCU- файлы) а так же VCL и ActiveX компоненты.

- Вирусы, поражающие исходный код

Вирусы данного типа поражают или исходный код программы, либо её компоненты (OBJ-, LIB-, DCU- файлы) а так же VCL и ActiveX компоненты.

❖ ПО ПОРАЖАЕМЫМ ОПЕРАЦИОННЫМ СИСТЕМАМ И ПЛАТФОРМАМ

- DOS
- Microsoft Windows
- Unix
- Linux

❖ ПО ТЕХНОЛОГИЯМ, ИСПОЛЬЗУЕМЫМ ВИРУСОМ

- Полиморфные вирусы

Вирус, который при заражении новых файлов и системных областей диска шифрует собственный код.

- Стелс-вирусы

Вирус, полностью или частично скрывающий свое присутствие в системе, путем перехвата обращений к операционной системе, осуществляющих чтение, запись, чтение дополнительной информации о зараженных объектах (загрузочных секторах, элементах файловой системы, памяти и т.д.)

- Руткит

Программа или набор программ для скрытия следов присутствия злоумышленника или вредоносной программы в системе

❖ ПО ЯЗЫКУ, НА КОТОРОМ НАПИСАН ВИРУС

- ассемблер
- высокоуровневый язык программирования
- скриптовый язык
- и др.

❖ ПО ДОПОЛНИТЕЛЬНОЙ ВРЕДОНОСНОЙ ФУНКЦИОНАЛЬНОСТИ

- Бэкдоры

Программы, которые устанавливает взломщик на взломанном им компьютере после получения первоначального доступа с целью повторного получения доступа к системе.

- Кейлоггеры

Модули для перехвата нажатий клавиш на компьютере пользователя, включаемые в состав программ-вирусов.

- Шпионы

Spyware — программное обеспечение, осуществляющее деятельность по сбору информации о конфигурации компьютера, деятельности пользователя и любой другой конфиденциальной информации без согласия самого пользователя.

- Ботнеты

Это компьютерная сеть, состоящая из некоторого количества хостов, с запущенными ботами — автономным программным обеспечением.

Обычно используются для нелегальной или неодобряемой деятельности — рассылки спама, перебора паролей на удалённой системе, атак на отказ в обслуживании.

Лекция 9. Безопасность в сети интернет (Защита)

Борьба с сетевыми угрозами

Установите комплексную систему защиты!

Установка обычного антивируса – вчерашний день. Сегодня актуальны так называемые «комплексные системы защиты», включающие в себя антивирус, файрволл, антиспам – фильтр и еще пару – тройку модулей для полной защиты вашего компьютера.

Новые вирусы появляются ежедневно, поэтому не забывайте регулярно обновлять базы сигнатур, лучше всего настроить программу на автоматическое обновление.

Будьте осторожны с электронной почтой!

Не стоит передавать какую-либо важную информацию через электронную почту.

Установите запрет открытия вложений электронной почты, поскольку многие вирусы содержатся во вложениях и начинают распространяться сразу после открытия вложения.

Программы Microsoft Outlooku Windows Mail помогают блокировать потенциально опасные вложения.

Пользуйтесь браузерами Mozilla Firefox, Google Chrome и Apple Safari!

Большинство червей и вредоносных скриптов ориентированы под Internet Explorer и Opera.

IE до сих пор удерживает первую строчку в рейтинге популярности, но лишь потому, что он встроен в Windows.

Уровень безопасности сильно хромает как у одного, так и у второго браузера, поэтому лучше им и не пользоваться вовсе.

Обновляйте операционную систему Windows!

Постоянно обновляйте операционную систему Windows.

Корпорация Microsoft периодически выпускает специальные обновления безопасности, которые могут помочь защитить компьютер.

Эти обновления могут предотвратить вирусные и другие атаки на компьютер, закрывая потенциально опасные точки входа.

Не отправляйте SMS-сообщения!

Сейчас очень популярны сайты, предлагающие доступ к чужим SMS и распечаткам звонков, также очень часто при скачивании файлов вам предлагают ввести свой номер, или внезапно появляется блокирующее окно, которое якобы можно убрать с помощью отправки SMS.

При отправке SMS, в лучшем случае, можно лишиться денег на счету телефона – если нужно будет отправить сообщение на короткий номер для оплаты, в худшем – на компьютере появится ужасный вирус.

Поэтому никогда не отправляйте SMS-сообщения и не вводите свой номер телефона на сомнительных сайтах при регистрации.

Пользуйтесь лицензионным программным обеспечением!

Если вы скачиваете пиратские версии программ или свеженький взломщик программы, запускаете его и сознательно игнорируете предупреждение антивируса, будьте готовы к тому, что можете поселить вирус на свой компьютер.

Причем, чем программа популярнее, тем выше такая вероятность.

Лицензионные программы избавят Вас от подобной угрозы!

Используйте брандмауэр!

Используйте брандмауэр Windows или другой брандмауэр, оповещающий о наличии подозрительной активности при попытке вируса или червя подключиться к компьютеру.

Он также позволяет запретить вирусам, червям и хакерам загружать потенциально опасные программы на компьютер.

Используйте сложные пароли!

Как утверждает статистика, 80% всех паролей — это простые слова: имена, марки телефона или машины, имя кошки или собаки, а также пароли вроде 123. Такие пароли сильно облегчают работу взломщикам.

В идеале пароли должны состоять минимум из семи, а лучше двенадцати символов. Время на подбор пароля из пяти символов — 2-4 часа, но чтобы взломать семисимвольный пароль, потребуется 2-4 года.

Лучше использовать пароли, комбинирующие буквы разных регистров, цифры и разные значки.

Делайте резервные копии!

При малейшей угрозе ценная информация с вашего компьютера может быть удалена, а что ещё хуже – похищена.

Возьмите за правило обязательное создание резервных копий важных данных на внешнем устройстве –флеш-карте, оптическом диске, переносном жестком диске.

Функция «Родительский контроль» обезопасит детей!

Для детской психики Интернет – это постоянная угроза получения психологической травмы и риск оказаться жертвой преступников.

Лекция 10. Управление инцидентами информационной безопасности: Жизненный цикл инцидента. Методы реагирования на инциденты

Управление инцидентами информационной безопасности (ИБ) — это процесс выявления, оценки, реагирования и анализа инцидентов, которые могут нарушить безопасность информационных систем, данных или сети. Инциденты могут быть вызваны как внешними угрозами, так и внутренними ошибками, и они могут иметь различные последствия для организации. Важно не только обнаружить инцидент, но и оперативно отреагировать на него, чтобы минимизировать ущерб и восстановить нормальную работу системы.

Жизненный цикл инцидента информационной безопасности

Жизненный цикл инцидента информационной безопасности состоит из нескольких этапов, каждый из которых играет ключевую роль в эффективном управлении инцидентами.

1. **Обнаружение инцидента**
   - На этом этапе важно выявить возможное нарушение безопасности. Обнаружение может происходить с помощью системы мониторинга (IDS/IPS), анализа логов, отчетности пользователей или автоматических систем.
   - **Признаки инцидента**: Нестандартная активность в сети, попытки взлома, утечка данных, несанкционированный доступ и другие.
2. **Оценка инцидента**
   - После того как инцидент обнаружен, необходимо оценить его серьезность и последствия. Это включает в себя определение типа инцидента, его масштаба и воздействия на систему или организацию.
   - **Оценка рисков**: Определение того, как инцидент может повлиять на конфиденциальность, целостность и доступность данных, а также на репутацию организации.
3. **Сдерживание инцидента**
   - На этом этапе предпринимаются меры для того, чтобы ограничить распространение инцидента и минимизировать его воздействие. Это может включать изоляцию зараженных систем, блокировку определенных пользователей или процессов.
   - **Пример**: Блокировка сетевого трафика с подозрительного IP-адреса или отключение зараженной машины от сети.
4. **Устранение инцидента**
   - После сдерживания инцидента необходимо устранить его причины. Это может включать в себя удаление вредоносного кода, исправление уязвимостей в системе, восстановление поврежденных данных или восстановление рабочих систем.
   - **Пример**: Удаление вирусов или троянов, установка патчей, удаление учетных записей, использованных злоумышленниками.
5. **Восстановление после инцидента**
   - После устранения последствий инцидента организация восстанавливает нормальное функционирование всех систем и сервисов. Это может включать восстановление данных из резервных копий, переподключение пользователей или серверов, а также проведение тестирования для подтверждения, что система работает безопасно.
   - **Пример**: Восстановление работы приложения после атаки или восстановление доступа к корпоративной сети.

6. **Анализ инцидента (Пост-операционный анализ)**
    - На этом этапе проводится анализ инцидента для того, чтобы понять его причины и сделать выводы для улучшения безопасности в будущем. Это включает в себя изучение того, как инцидент был обнаружен, какие действия были предприняты, что было сделано правильно, а что можно улучшить.
    - **Пример**: Обсуждение причин инцидента, совершенствование политики безопасности, обновление планов реагирования на инциденты.
7. **Документация и отчетность**
    - Вся информация о инциденте, включая его описание, предпринятые меры и результаты анализа, должна быть задокументирована для последующих отчетов. Эта документация помогает в обучении сотрудников, а также может быть использована в юридических целях, если инцидент имеет серьезные последствия.
    - **Пример**: Создание отчета для руководства, органов регулирования или внешних проверяющих.

Методы реагирования на инциденты

Методы реагирования на инциденты могут варьироваться в зависимости от типа инцидента, его серьезности и ресурсов организации. Однако в любом случае реакция должна быть быстрой и эффективной, чтобы минимизировать ущерб и быстро восстановить нормальное функционирование систем.

1. **План реагирования на инциденты**
    - Для каждого типа инцидента должен существовать заранее подготовленный план реагирования. Этот план описывает процедуры, которые должны быть выполнены в случае различных инцидентов (например, взлом системы, утечка данных, атаки DDoS).
    - **Пример**: План реагирования для инцидента с утечкой данных включает процедуры для оповещения пользователей, замены компрометированных паролей, уведомления органов надзора и т. д.
2. **Использование автоматизированных систем**
    - Автоматизированные системы безопасности, такие как системы IDS/IPS, могут быстро реагировать на определенные типы атак и угроз. Например, если система IDS обнаруживает подозрительный трафик, она может автоматически блокировать источник трафика или прекратить определенные соединения.
    - **Пример**: Автоматическая блокировка IP-адресов, с которых исходят DDoS-атаки.
3. **Использование виртуальных или физический сегментов сети**
    - В случае серьезных атак важно сегментировать сеть, чтобы предотвратить распространение инцидента на другие части организации. Это может включать создание отдельных виртуальных сетей (VLAN), изоляцию зараженных сегментов сети или блокировку некоторых портов.
    - **Пример**: В случае атаки с использованием вируса, изоляция зараженной машины от корпоративной сети.
4. **Уведомление заинтересованных сторон**
    - Важно вовремя оповещать всех заинтересованных сторон о текущем инциденте. Это может включать информирование внутренних и внешних пользователей, руководство, юридический отдел, аудиторов, а также, если необходимо, органы надзора и правоохранительные органы.
    - **Пример**: Уведомление клиентов о возможной утечке персональных данных или контакт с регуляторными органами в случае серьезной утечки данных.
5. **Тренировки и учения**

- Организация должна регулярно проводить тренировки и учения для своей команды реагирования на инциденты. Это позволяет улучшить подготовленность к реальным инцидентам и выявить слабые места в процессах.
- **Пример**: Проведение учений по реакции на кибератаки, симуляция утечек данных, тестирование планов реагирования.

6. **Реакция на инциденты по типу угроз**
   - В зависимости от типа угрозы могут быть использованы различные методы реагирования. Например, в случае DDoS-атаки важно быстро внедрить механизмы фильтрации трафика, в то время как при утечке данных требуется оперативное оповещение и мониторинг компрометированных данных.
   - **Пример**: При атаке с использованием уязвимостей необходимо срочно применить патчи или блокировать уязвимые сервисы.

Лекция 11. Будущее информационной безопасности

Будущее информационной безопасности будет определяться рядом технологий, изменений в законодательстве, новых угроз и методов защиты. В условиях быстрого развития технологий и увеличения числа киберугроз, информационная безопасность будет сталкиваться с новыми вызовами и возможностями. Вот несколько ключевых тенденций, которые, скорее всего, будут определять будущее информационной безопасности:

1. Рост использования искусственного интеллекта и машинного обучения

Искусственный интеллект (ИИ) и машинное обучение (МО) будут играть важную роль в информационной безопасности, как с точки зрения атак, так и с точки зрения защиты. Системы на базе ИИ смогут:

- **Обнаруживать аномалии**: ИИ и МО смогут анализировать большие объемы данных и обнаруживать аномальные паттерны, которые могут свидетельствовать о вторжении.
- **Анализировать поведение**: Машинное обучение позволит отслеживать поведение пользователей и устройств, чтобы выявлять потенциальные угрозы до того, как они станут настоящей проблемой.
- **Автоматизировать реагирование**: ИИ будет способен автоматически реагировать на угрозы, блокируя атаки или изолируя зараженные системы в реальном времени.

Однако, с другой стороны, злоумышленники также могут использовать ИИ для создания более совершенных атак, таких как фишинг с использованием глубоких фейков (deepfake) или более сложных методов обхода систем защиты.

2. Увеличение угроз от Интернета вещей (IoT)

С ростом числа подключенных устройств (от умных домов до производственных систем), угрозы, связанные с **Интернетом вещей (IoT)**, становятся все более актуальными. Многие IoT-устройства не имеют встроенных механизмов безопасности, что делает их уязвимыми для атак. Киберпреступники могут использовать уязвимости в этих устройствах для атак на более крупные инфраструктуры.

Будущее будет включать в себя усиленную безопасность IoT-устройств, с акцентом на:

- **Интеграцию более жестких стандартов безопасности**.

- **Шифрование данных**, передаваемых между устройствами.
- **Обновления и патчи** для IoT-устройств, так как многие из них не получают регулярных обновлений безопасности.

3. Защита данных и конфиденциальности

В условиях все большего сбора данных о пользователях и организаций, защита персональной и корпоративной информации будет оставаться важнейшей задачей информационной безопасности. Это включает в себя:

- **Регулирование конфиденциальности данных**: В будущем можно ожидать более жесткие требования по защите данных (аналогично GDPR в Европе). Это будет требовать от компаний использования более эффективных технологий защиты данных и соблюдения новых стандартов.
- **Шифрование**: Защита данных будет еще больше полагаться на шифрование, как для защиты при передаче, так и для защиты в состоянии покоя.
- **Технологии для защиты конфиденциальности (например, Zero-Knowledge Proofs)**: Будут развиваться методы, которые позволяют делиться данными, не раскрывая саму информацию, что будет особенно актуально в условиях защиты личной информации.

4. Киберзащита в облаке и гибридных инфраструктурах

С увеличением использования облачных сервисов, организации будут сталкиваться с новыми вызовами безопасности, связанными с хранением данных и выполнением операций в облаке:

- **Облачные угрозы**: В будущем в облачных сервисах будет развиваться больше методов защиты от атак, таких как утечки данных, несанкционированный доступ, атаки на виртуализированные системы.
- **Сложность защиты гибридных инфраструктур**: Многие организации будут использовать гибридные инфраструктуры (облачные и локальные ресурсы), что потребует новых методов защиты для защиты данных и сервисов, работающих как в облаке, так и в частных дата-центрах.

5. Прогнозирование угроз и проактивная безопасность

В будущем информационная безопасность будет все больше стремиться к **проактивной защите**, что подразумевает предсказание угроз и принятие мер до того, как инцидент произойдет:

- **Прогнозирование с использованием больших данных**: Анализ данных, собранных из множества источников (системы мониторинга, данные об угрозах, уязвимостях и другие), позволит прогнозировать возможные атаки.
- **Threat Hunting (активное обнаружение угроз)**: Специалисты по безопасности будут использовать интеллектуальные системы для поиска и нейтрализации угроз до того, как они нанесут ущерб.

6. Развитие блокчейн-технологий

Блокчейн уже активно используется в криптовалютах, но в будущем его применение в области безопасности будет расширяться:

- **Безопасность транзакций**: Блокчейн может быть использован для защиты транзакций и данных, благодаря его неизменяемости и криптографической защите.
- **Цифровые удостоверения и аутентификация**: Блокчейн может сыграть ключевую роль в области **идентификации и аутентификации**, обеспечивая защиту личных данных и предотвращая фальсификацию данных.

7. Социальная инженерия и фишинг

Несмотря на развитие технологий, **социальная инженерия** и фишинг останутся важнейшими инструментами киберпреступников. В будущем они станут еще более изощренными благодаря использованию:

- **Глубоких фейков (Deepfakes)**: Злоумышленники могут использовать фальшивые аудио- и видеозаписи для манипулирования людьми и организации фишинговых атак.
- **Психологической манипуляции**: В будущем атаки будут становиться более персонализированными и точечными, что потребует усиленного внимания к обучению сотрудников и распространению информации о новых угрозах.

8. Кибервойна и геополитические угрозы

Кибератаки будут становиться не только инструментом для киберпреступников, но и **средством ведения геополитической борьбы**. Государственные и террористические группы будут использовать кибератаки для достижения политических и экономических целей:

- **Атаки на критически важную инфраструктуру**: Энергетические сети, транспортные системы, финансовые учреждения — все эти области будут нацелены в рамках кибервойны.
- **Международные соглашения и сотрудничество**: Ожидается, что страны будут работать вместе для создания международных стандартов безопасности и борьбы с киберугрозами.

9. Применение квантовых технологий

В будущем, с развитием **квантовых вычислений**, появится необходимость в новых методах защиты данных:

- **Квантовое шифрование**: Квантовые технологии обеспечат совершенно новые методы защиты данных, которые будут практически невозможно взломать с использованием современных методов.
- **Устойчивость к квантовым атакам**: В то же время появятся угрозы со стороны квантовых вычислений, которые смогут обходить традиционные методы шифрования.

9.2. Содержание практических занятий

ПРАКТИЧЕСКАЯ РАБОТА №1

ПРОСТОЕ ШИФРОВАНИЕ И ДЕШИФРОВАНИЕ - ШИФР ЦЕЗАРЯ

До появления компьютеров криптография состояла из символьных алгоритмов. Криптографические алгоритмы либо заменяли одни символы другими, либо переставляли символы. Лучшие алгоритмы делали и то и другое, причем многократно.

Подстановочные шифры

Подстановочным (substitution cipher) называется шифр, который каждый символ открытого текста заменяет другим символом в шифротексте. Получатель выполняет обратную подстановку в шифротексте, восстанавливая открытый текст. В классической криптографии существуют четыре типа подстановочных шифров.

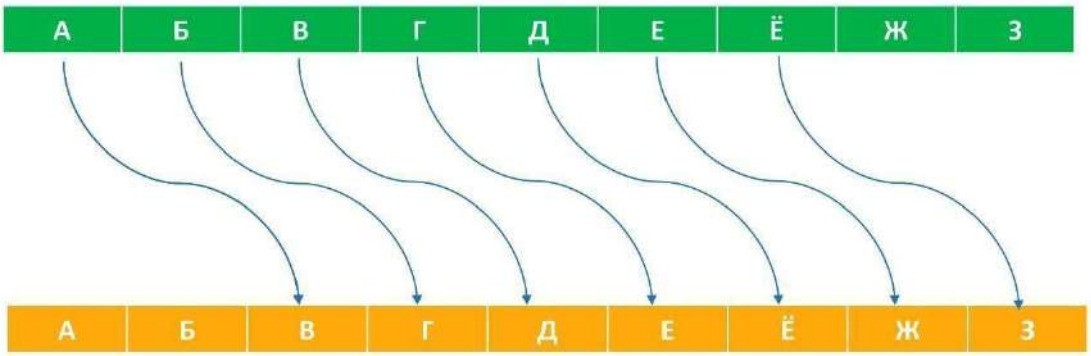
Простой подстановочный шифр (simple substitution cipher), или моноал- фавитный шифр (monoalphabetic cipher), - это шифр, который заменяет каждый символ открытого текста соответствующим символом шифротекста. Примером простых подстановочных шифров являются криптограммы в газетах.

Омофонический подстановочный шифр (homophonic substitution cipher) похож на простую подстановочную криптосистему, за исключением того, что один символ открытого текста заменяется несколькими символами шифротекста. Например, букве А может соответствовать набор чисел 5, 13, 25 или 56, букве В — 7, 19, 31 или 42 и т.д.

Полиграммный подстановочный шифр (polygram substitution cipher) — это шифр, который заменяет одни блоки символов другими. Например, символам ABA могут соответствовать символы RTQ, символам ABB — символы SLL и т.д.

Полиалфавитный подстановочный шифр (polyalphabetic substitution cipher) состоит из нескольких простых подстановочных шифров. Например, можно использовать пять разных простых подстановочных фильтров так, что каждый символ открытого текста заменяется с использованием одного конкретного шифра.

Шифр Цезаря, также известный как шифр сдвига, код Цезаря или сдвиг Цезаря — один из самых простых и наиболее широко известных методов шифрования. Шифр Цезаря — это вид шифра подстановки, в котором каждый символ в открытом тексте заменяется символом, находящимся на некотором постоянном числе позиций левее или правее него в алфавите. Например, в шифре со сдвигом вправо на 1, А была бы заменена на Б, Б станет Г, и так далее:



Шифр Цезаря представляет собой простой подстановочный фильтр. На самом деле этот алгоритм еще проще, чем подстановочный, потому что алфавит шифротекста представляет собой результат смещения алфавита открытого текста, а не его случайную перестановку.

Лабораторная работа выполняется в среде Visual Studio 2019 с использование языка программирования Python. Начало работы определено в методических указаниях «*Среда Visual Studio 2019 и использование языка программирования Python для проектов дисциплины «Информационная безопасность телекоммуникационных систем*».

Формирование шифротекста

В начале программы определяем алфавиты русский и английский.

```
alfavit_EU =  'ABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOPQRSTUVWXYZ'
alfavit_RU = 'АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯАБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ'
Вручную будет определяться шаг сдвига в шифротексте и сообщение для шифровки, переменные
smeshenie и message, а также переменная для шифротекста itog
smeshenie = int(input('Шаг шифровки: '))
message = input("Сообщение для шифровки: ").upper() itog = ''
```

В программе можно использовать либо русский текст или английский. Вручную вводится тип языка.

```
lang = input('Выберите язык RU/EU: ')     #Добавляем возможность выбора языка
```

Алгоритм формирования шифротекста и его печать будет следующий:

```
if lang == 'RU':
for i in message:
mesto = alfavit_RU.find(i)  # Алгоритм для шифрования сообщения на русском
new_mesto = mesto + smeshenie if i in alfavit_RU:
itog += alfavit_RU[new_mesto] else:
itog += i
else:
for i in message:
mesto = alfavit_EU.find(i) # Алгоритм для шифрования сообщения на английском
new_mesto = mesto + smeshenie if i in alfavit_EU:
itog += alfavit_EU[new_mesto] else:
itog += i

print (itog)
```

Дешифровка шифротекста (сообщения)

Алгоритм дешифровки шифротекста вляется обратным шифровке. Изменения произойдут только в части величины шаг, знак должен указываться « - ».

```
alfavit_EU =  'ABCDEFGHIJKLMNOPQRSTUVWXYZABCDEFGHIJKLMNOPQRSTUVWXYZ'

alfavit_RU = 'АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯАБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ'

smeshenie = int(input('Шаг дешифровки (не забудьте про знак): ')) message = input("Сообщение для Дешифровки: ").upper()

itog = ''

lang = input('Выберите язык RU/EU: ') if lang == 'RU':

for i in message:

mesto = alfavit_RU.find(i) new_mesto = mesto + smeshenie if i in alfavit_RU:

itog += alfavit_RU[new_mesto] else:

itog += i

else:

for i in message:

mesto = alfavit_EU.find(i) new_mesto = mesto + smeshenie if i in alfavit_EU:

itog += alfavit_EU[new_mesto] else:

itog += i


print (itog)
```

ЗАДАНИЯ РАБОТЫ

Создать проект в среде Visual Studio 2019 с использование языка программирования Python.Сформироватьнеобходимое окружения языка Python из библиотек, необходимых для выполнения лабораторной работы.

1. Создать два файла-программы в языке python для шифровки и дешифровки.
2. Сформировать тексты программ, в соответствие с методическими указаниями, для шифровки и дешифровки.
3. Подготовить 4 примера – 2 на русском языке и 2 на английском для подготовки шифротекста. Примеры должны содержать правильные тексты (символы алфавита) и ошибочные.
4. Сформировать шифротексты.
5. Выполнить дешифровку шифротекстов.
6. Оформить отчет по работе с указанием описания алгоритма Цезаря, описания программ шифровки и дешифровки, описание примеров и указания недостатков и достоинств алгоритма Цезаря.

ПРАКТИЧЕСКАЯ РАБОТА №2
СИММЕТРИЧНЫЕ КРИПТОСИСТЕМЫ

Симметричные алгоритмы или алгоритмы шифрования с одним ключом используют для шифрования и дешифрования один и тот же ключ. В этом разделе рассматриваются простейшие алгоритмы шифрования, представляющие принципиальную основу современных компьютерных алгоритмов шифрования.

3.1.1 Шифры перестановки

В шифрах средних веков часто использовались таблицы, с помощью которых выполнялись простые процедуры шифрования, основанные на перестановке букв в сообщении. Ключами в этих алгоритмах являются размеры таблицы и порядок перестановки. Пример данного метода шифрования текста «**И БУМАЖКОЙ ПРИКРОЕМ БРЕШЬ**» показан в таблицах на рис 3.1. Сначала в таблицу записывается текст сообщения (рис 3.1, а), а потом поочередно переставляются столбцы в определенном порядке (на рис 3.1, б) в первой строке, а затем строки (на рис 3.1, в) в последнем столбце. При расшифровке порядок перестановок был обратный. Пример данного метода шифрования показан в следующих таблицах:

	1	*2*	*3*	*4*	*5*
1	**И**		**Б**	**У**	**М**
2	**А**	**Ж**	**К**	**О**	**Й**
3		**П**	**Р**	**И**	**К**
4	**Р**	**О**	**Е**	**М**	
5	**Б**	**Р**	**Е**	**Ш**	**Ь**

а)

4	*2*	*1*	*3*	*5*
У		И	Б	М
О	Ж	А	К	Й
И	П		Р	К
М	О	Р	Е	
Ш	Р	Б	Е	Ь

б)

И	**П**		**Р**	**К**	*3*
М	**О**	**Р**	**Е**		*4*
Ш	**Р**	**Б**	**Е**	**Ь**	*5*
О	**Ж**	**А**	**К**	**Й**	*2*
У		**И**	**Б**	**М**	*1*

в)

Рис. 3.1. Двойная перестановка столбцов и строк

В результате перестановки текста «**И БУМАЖКОЙ ПРИКРОЕМ БРЕШЬ**» получена шифровка «**ИП РКМОРЕ ШРБЕЬОЖАКЙУ ИБМ»**. Ключом к шифру служат номера столбцов 4 2 1 3 5 и номера строк 3 4 5 2 1 исходной таблицы.

Для удобства запоминания ключей можно использовать в их качестве слова. При этом порядок перестановки определяется нумерацией букв в слове в алфавитном порядке. В приведенном примере использованы *Ключ1* – «СПОРТ», буквы в алфавите располагаются в порядке 4 2 1 3 5 (ОПРСТ – 12345) и *Ключ2* – «СТУЖА», буквы в алфавите располагаются в порядке 3 4 5 2 1 (АЖСТУ–12345). Число вариантов двойной перестановки достаточно быстро возрастает с увеличением размера таблицы: для таблицы 3 х 3 их 36, для 4 х 4 их 576, а для 5 х 5 их 14400.

Для обеспечения дополнительной защиты можно повторно шифровать сообщение, которое уже было зашифровано. Для этого размер второй таблицы подбирают так, чтобы длины ее строк и столбцов отличались от длин строк и столбцов первой таблицы. При этом размеры второй таблицы должны быть взаимно простыми с размерами первой таблицы.

Для шифрования применялись магические квадраты – квадратные таблицы с вписанными в их клетки последовательными натуральными числами, начиная с единицы, которые дают в сумме по каждому столбцу, каждой строке и каждой диагонали одно и то же число. Свойство магического квадрата используется для повышения эффективности шифра при данном алгоритме шифрования. Для шифрования необходимо вписать исходный текст «ЧИСЛОШЕСТНАДЦАТЬ» по приведенной в магическом квадрате нумерации и затем переписать содержимое таблицы по строкам (рис 3.2). В результате получается шифротекст «ЬСИЦОНАСТШЕДЛТАЧ», сформированный благодаря перестановке букв исходного сообщения.

Исходный текст: Ч И С Л О Ш Е С Т Н А Д Ц А Т Ь

Ч	И	С	Л	О	Ш	Е	С	Т	Н	А	Д	Ц	А	Т	Ь
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

Магический квадрат

16	3	2	13
5	10	11	8
9	6	7	12
4	15	14	1

Шифрование

Ь	С	И	Ц
О	Н	А	С
Т	Ш	Е	Д
Л	Т	А	Ч

Шифротекст: Ь С И Ц О Н А С Т Ш Е Д Л Т А Ч
Рис. 3.2. Шифрование с помощью магического квадрата

Дешифровка шифротекста происходит в обратном порядке: вначале текст вписывается последовательно слева направо в квадрат, затем буквы с квадрата выбираются в порядке, определенном в магическом квадрате.

3.1.2 Шифры простой замены

Шифры простой замены использовались еще в древней Греции (V–VI до н.э.), которые и в настоящее время являются частью отдельных алгоритмов шифрования.

Система шифрования Цезаря – частный случай шифра простой замены. Метод основан на замене каждой буквы сообщения на другую букву того же алфавита, путем смещения от исходной буквы на K букв.

Известная фраза Юлия Цезаря VENI VINI VICI – пришел, увидел, победил, зашифрованная с помощью данного метода, преобразуется в SBKF SFAF SFZF (при смещении на 4 символа).

Греческим писателем Полибием за 100 лет до н.э. был изобретен так называемый *квадрат Полибия* размером 5 х 5, заполненный алфавитом в случайном порядке. Греческий алфавит имеет 24 буквы, а 25-м символом является пробел. Для шифрования на квадрате находили букву текста и записывали в шифротекст букву, расположенную ниже ее в том же столбце. Если буква оказывалась в нижней строке таблицы, то брали верхнюю букву из того же столбца.

3.1.3 Шифры сложной замены

Шифр Гронсфельда состоит в модификации шифра Цезаря числовым ключом. Для этого под буквами сообщения записывают цифры числового ключа. Если ключ короче сообщения, то его запись циклически повторяют. Шифротекст получают примерно также, как в шифре Цезаря, но отсчитывают не третью букву по алфавиту (как в шифре Цезаря), а ту, которая смещена по алфавиту на соответствующую цифру ключа.

Пусть в качестве ключа используется группа из трех цифр – 314, тогда сообщение «**ТРУДНО В УЧЕНИИ**» преобразуется в шифрограмму «**ХСШЖОТВГГЧШИРКН**» (рис. 3.3).

Сообщение **Т Р У Д Н О В У Ч Е Н И И**
Ключ **3 1 4 3 1 4 3 1 4 3 1 4 3 1 4**
Шифровка **Х С Ш Ж О Т В Г Г Ч Ш И Р К Н**

Рис. 3.3. Реализация шифра Гронсфельда

В *шифрах многоалфавитной замены* для шифрования каждого символа исходного сообщения применяется свой шифр простой замены (рис. 3.4).

	АБВГДЕЁЖЗИКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ_

А	АБВГДЕЁЖЗИКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ_
Б	_АБВГДЕЁЖЗИКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ
В	Я_АБВГДЕЁЖЗИКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮ
Г	ЮЯ_АБВГДЕЁЖЗИКЛМНОПРСТУФХЦЧШЩЪЫЬЭ
.	……
Я	ВГДЕЁЖЗИКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ_АБ
_	БВГДЕЁЖЗИКЛМНОПРСТУФХЦЧШЩЪЫЬЭЮЯ_А

Рис. 3.4. Таблица для реализации шифра многоалфавитной замены

Сообщение	Э К С П Е Р И М Е Н Т
Ключ	Б А Б А Я Г А Б А Б
Шифровка	Ь К Р П Ё Т Ё М Д Н С

Рис. 3.5. Пример реализации шифра многоалфавитной замены

Каждая строка в этой таблице соответствует одному шифру замены аналогично шифру Цезаря для алфавита, дополненного пробелом. При шифровании сообщения его выписывают в строку, а под ним ключ. Если ключ оказался короче сообщения, то его циклически повторяют. Шифротекст получают, находя символ в колонке таблицы (рис 3.4) по букве текста и строке, соответствующей букве ключа. Например, используя ключ «БАБА ЯГА», из сообщения «ЭКСПЕРИМЕНТ» получаем следующую шифровку «ЬКРПЁТЁМДНС».

Такая операция с использованием таблицы, приведенной на рис 3.4, соответствует сложению кодов ASCII символов сообщения и ключа по модулю равным числу символов.

3.1.4 Гаммирование

Процесс шифрования заключается в генерации гаммы шифра (ключа) и наложении этой гаммы на исходный открытый текст. Перед шифрованием открытые данные разбиваются на блоки $T(0)_i$ одинаковой длины (например, по 64 бита). Ключ шифра (гамма шифра) вырабатывается в виде последовательности блоков $Г(ш)_i$ аналогичной длины, шифротекст $T(ш)_i=Г(ш)_i+T(0)_i$, где «+» – побитовое сложение, $i=1,2,…,m$. Процесс дешифрования сводится к повторной генерации шифра текста и наложение этой гаммы на зашифрованные данные, т.е. $T(0)_i=Г(ш)_i+T(ш)_i$. Алгоритмы гаммирования легко реализуются на компьютере и, как правило, являются частью симметричных алгоритмов криптографии.

Задания для выполнения

3.2.1 Используя алгоритмы двойной перестановки строк и столбцов выполнить шифрование следующих фраз (ключ выбирать самостоятельно, номер варианта выбрать по номеру в списке группы):

1. И дольше века длится день.
2. Без труда не выловишь и рыбку из труда
3. Ученый без трудов — дерево без плодов.
4. Надо любить жизнь больше, чем смысл жизни.
5. Подумай, прежде чем подумать.
6. Желудок умнее мозга, потому что желудок умеет тошнить. Мозг же глотает любую дрянь
7. Только самые мудрые и самые глупые не поддаются обучению.
8. Окно в мир можно закрыть газетой.
9. Чаще всего выход там, где был вход.
10. Безграмотные вынуждены диктовать.
11. Хлеб открывает любой рот.
12. Деньги не пахнут, но улетучиваются.
13. Человек не умирает до тех пор, пока живут знавшие его.
14. Когда счастье есть, о нем не думают.
15. У земли нет края, у труса нет Родины.
16. Самое трудное для человека — быть каждый день Человеком.
17. Беззубым многое легче выговаривать.
18. Добро не лежит на дороге, его случайно не подберешь.
19. Говорят, что друзья познаются в беде, а как мне кажется, и в радости они тоже познаются
20. Важно не количество знаний, а качество их. Можно знать очень многое, не зная самого нужного.
21. Вписывайся во влиятельные круги.
22. И без людей человек не может жить и с людьми тяжко.
23. Добру человек у человека учится.

3.2.2 Используя алгоритмы двойной перестановки строк и столбцов выполнить дешифрование шифрограмм, приведенные в таблице 3.1 (номер варианта выбрать по последней цифре номера шифра). В шифротексте следует обратить внимание на наличие пробелов в тексте, длина текста по всем вариантам равняется 25 символам:

Таблица 3.1

Номер вар-та	Шифротекст	Ключ 1	Ключ 2
1	**В ОН, Т ОЭЗКНОА УОРСЗКНОА**	КРУТО	СТУЖА
2	**ЗВАОЛИ ЛАН ОДОРОНЧСАЧТЕЗ**	ВЕСНА	ОСЕНЬ
3	**ПАЙРДЕЕЖ М ЧЕДАТУМЬДУПОМ**	ОСЕНЬ	ДОСУГ
4	**ДОВХЫМА Т ЕД Г ДО ХВ ИИЩ**	ТРАВА	ДОСУГ
5	**!Т РОЙОЛЮБ БХЛЕ ТВАЕЫРОТК**	ПРАВО	ТРАВА
6	**Ь ДА ОЖЧЕДТУНДРЕ СВЕЕОП Г**	КРУТО	ПРАВО
7	**ЕН ПОЕРД ЕОБР! ЗАН А ШИИК**	СПОРТ	КРУТО
8	**Е ВГОБЫ-М БЕУЗЗ ЛЧЕГОРЬИТ**	ВЕТЕР	СПОРТ
9	**ГАЛЕР ЗВИИОМУНЗЯТ НЕ РАОП**	СТУЖА	ВЕТЕР
10	**СЯТООН ОН УЫЖННЫПЕН ЕЖННУ**	ДРЕВО	СТУЖА

3.2.3 Используя магический квадрат (таблица 3.2) расшифровывать следующие шифрограммы (шифрограммы приведены в таблице 3.3, номер варианта выбрать по числу букв в фамилии):

Таблица 3.2

11	24	7	20	3
4	12	25	8	16
17	5	13	21	9
10	18	1	14	22
23	6	19	2	15

Таблица 3.3

Вариант	Шифротекст
1	ОЛ ЕЛ ОДУЛА-СЕЛЯС МЕЛЙДГ
2	ВТТЙЕБА КЛЮ Е РЫБХТРОООЛ
3	ОЕР Д ТОАЛОЗЗЧНИОААЧСЛНВ
4	УЗУНОБЛЯСВАОИЕИМТСРЛЬДЬВО
5	ЕТЙДДУЖЬ ЧЕМДУПРМПЕМАА О
6	ЕАЕЕУДГД ПОНОЧВСДТ Ь ЕЖР
7	КРШЗ РЕИ НПЕА АНДБОИ ЕО
8	ОНЫ НУСЫЕННЖТН ПОНОУЖН ЕЯ
9	ЕМИАГАНУ ПОЛЯЗЗВ РТНОИРЕ
10	НУУ З Е!ДЛЬТ РА КБТЫБРОЕО

3.2.4 Используя шифр многоалфавитной замены шифровать фразу из п. 3.2.1 (исключив пробелы и знаки препинания), используя в качестве ключа «Ключ 1» из пункта 3.2.2. Для шифрования использовать алфавит замены из таблицы 3.5.

3.2.5 Используя шифр многоалфавитной замены дешифровать фразу, используя «Ключ» (шифрограммы и ключи приведены в таблице 3.4, номер варианта выбрать по последней цифре суммы числа букв в имени и фамилии).

Таблица 3.4

Номер вар-та	Шифротекст	Ключ
1	РПЮЫВОНЕЩОИТЯФАМХМЯЬЕЕШТТРО	ВЕСНА
2	ПЦМРМОЭУЯЙЙЦЗИЙБЧЙТЙЙХНЧОЬУЕЯШ	ОСЕНЬ
3	ЩЩЦФСЦШБОЕДУГЮБЕЬЪГСЦ	ДОСУГ
4	ГЭЫЙАФШСБТАВПРЛАЦЕПИСБПБЩЧУО	ТРАВА
5	РХЗЙБРЛМОЫЭУОЗЩФУЧЗРКУОДОЯШВВАЛ	ПРАВО
6	ХШЙЧЪПААНЧЩРЮТЕШБЮТПХПШДЭПВЮР	КРУТО
7	ЩЬАХЭЬФШВЕСЪКЭТРВХЮГГЛЖШВЩБЯП	СПОРТ
8	ДФЪЦЛДЕЫЦПДУФРШБЧЧРМПАЧПАХИЪ	ВЕТЕР
9	УБЫЧЫУТЬЧЯУАХСИРДШСЬЮНШРРДХЫ	СТУЖА
10	МЬЕБАСШПКТЙВЗПЬЗГЦРРФХСЗЫЙЬ	ДРЕВО

Дешифрованный текст привести с пробелами.

Таблица 3.5

	А	**Б**	**В**	**Г**	**Д**	**Е**	**Ж**	**З**	**И**	**Й**	**К**	**Л**	**М**	**Н**	**О**	**П**	**Р**	**С**	**Т**	**У**	**Ф**	**Х**	**Ц**	**Ч**	**Ш**	**Щ**	**Ъ**	**Ы**	**Ь**	**Э**	**Ю**	**Я**
А	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я
Б	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А
В	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б
Г	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В
Д	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г
Е	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д
Ж	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е
З	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж
И	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З
Й	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И
К	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й
Л	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К
М	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л
Н	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М
О	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н
П	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О
Р	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П
С	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р
Т	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С
У	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т
Ф	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У
Х	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф
Ц	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х
Ч	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц
Ш	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч
Щ	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш
Ъ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ
Ы	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ
Ь	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы
Э	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь
Ю	Ю	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э
Я	Я	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю
И	И	Й	К	Л	М	Н	О	П	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я	А	Б	В	Г	Д	Е	Ж	З

ПРАКТИЧЕСКАЯ РАБОТА №3

СОЗДАНИЕ И НАСТРОЙКА ВИРТУАЛЬНОЙ МАШИНЫ VIRTUALBOX НА ПК С ОПЕРАЦИОННОЙ СИСТЕМОЙ UBUNTU

Цель работы:

1. Установка платформы виртуализации VirtualBox.
2. Создание и настройка виртуальной машины VirtualBox с ОС Ubuntu.
3. Практическое знакомство с ОС Ubuntu-desktop x32.
4. Компиляция программ на C/C++ в терминале Ubuntu-desktop x32.

Введение

С помощью бесплатной программы VirtualBox можно создать на своем компьютере виртуальную машину с другой гостевой операционной системой. Платформа виртуализации VirtualBox создает виртуальные машины, в которые можно будет установить разные операционные системы: Windows, Linux, Mac OS X и т. д.

VirtualBox – платформа виртуализации, имитирующее работу ПК. Позволяет устанавливать и запускать операционные системы как обыкновенные приложения. Создает на ПК изолированное окружение, состоящее из: жесткого диска, видеокарты, памяти, контроллеров устройств. Может кому-то потребуется включить виртуализацию. Дело в том, что по умолчанию в настройках BIOS большинства материнских плат виртуализация отключена. Ее необходимо включить, зайдя в BIOS в соответствующий раздел, который называется у каждого производителя по-своему, например, «Virtualization Technology», изменив значение опции с «Disabled» на «Enabled».

Популярные способы применения:

1. Знакомство с другими ОС: Linux, FreeBSD, MacOS, любая из версий Windows, Android. Система работает изолированно. Можно экспериментировать, не боясь, что нарушится работа реальной системы.
2. Запуск программных продуктов, несовместимых с основной ОС.
3. Использование старых приложения.
4. Тестирование потенциально опасных приложений.

Ubuntu – дистрибутив Linux, основанный на ядре Linux. Основным разработчиком и спонсором является компания Canonical. В настоящее время проект активно развивается и поддерживается свободным сообществом. Ubuntu – это операционная система, которая идеально подходит для использования на персональных компьютерах, ноутбуках и серверах. Она содержит все необходимые программы, которые нужны всем: программу просмотра Интернет, офисный пакет для работы с текстами, электронными таблицами и презентациями, программы для общения в Интернет и много других.

Ссылки на скачивание VirtualBox и Ubuntu: VirtualBox-6.0.24-139119-

Win.exe https://file.tpu.ru/index.php/s/v5PnrEHucLBuOIf

Ubuntu-14.04.6-desktop-i386.iso

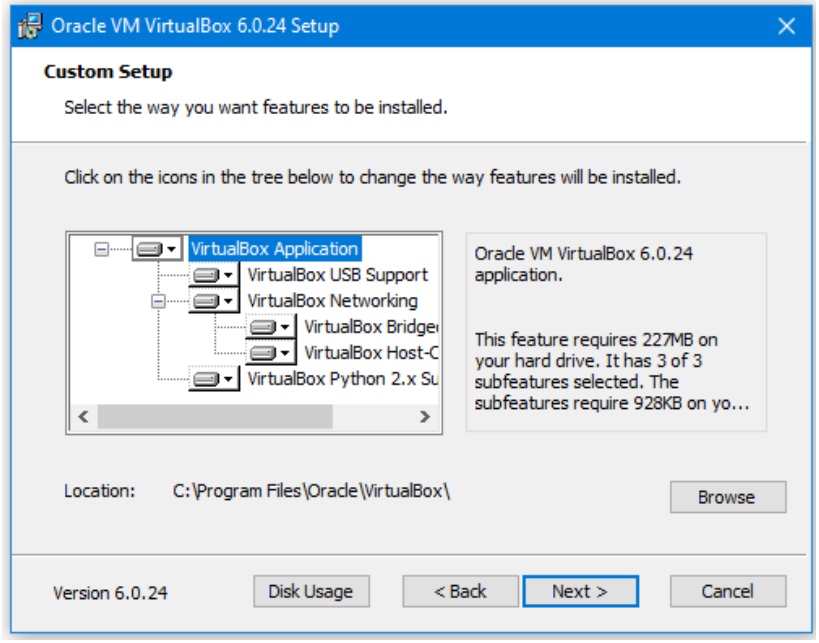
https://file.tpu.ru/index.php/s/CQ7G6WPyhhi4cSF

Установка платформы виртуализации VirtualBox

Дистрибутив представлен как один инсталляционный файл расширения «exe».

Нажимаем по нему два раза мышкой. Откроется окно помощника. Нажимаем «Next».

Нажимаем «Next».
Выбираем место для инсталляции. Будет предложено установить все компоненты. Не рекомендуются без необходимости отключать их. Они нужны даже при минимальном использовании. Нажимаем «Next».

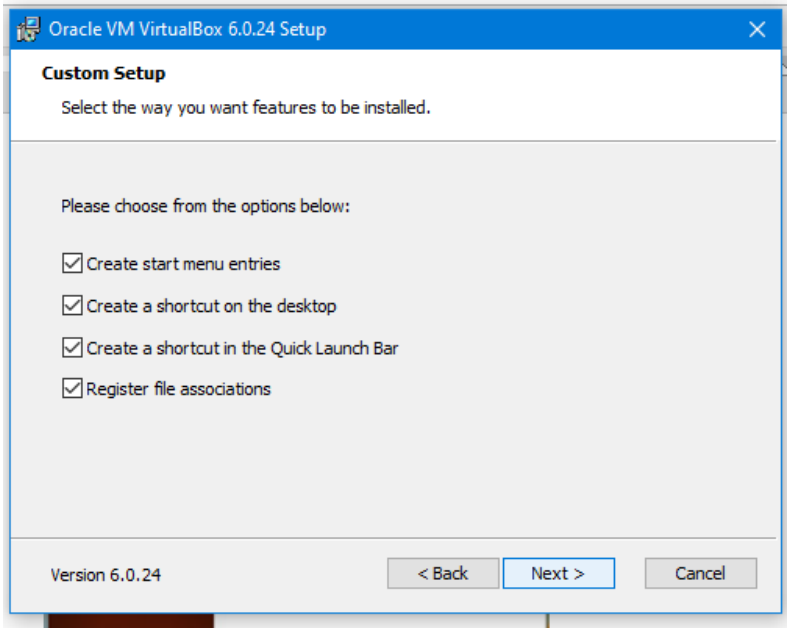


В новом окне расположены такие настройки запуска:

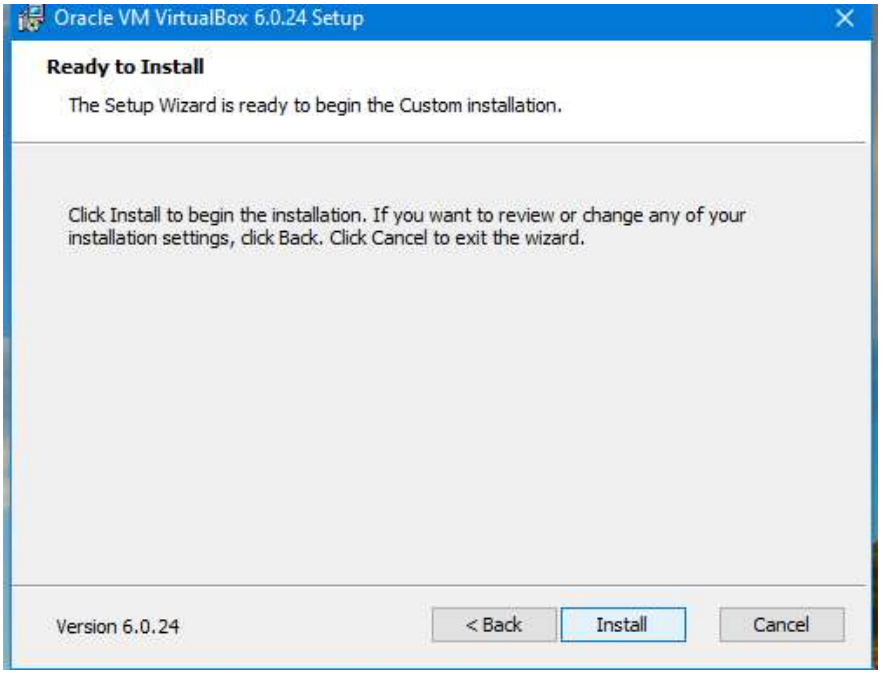
1. Создание пунктов меню "Пуск";

2. Создание ярлыка на рабочем столе;
3. Ярлык на панели быстрого запуска;
4. Зарегистрировать расширения файлов программы в ОС.

Нажимаем «Next».

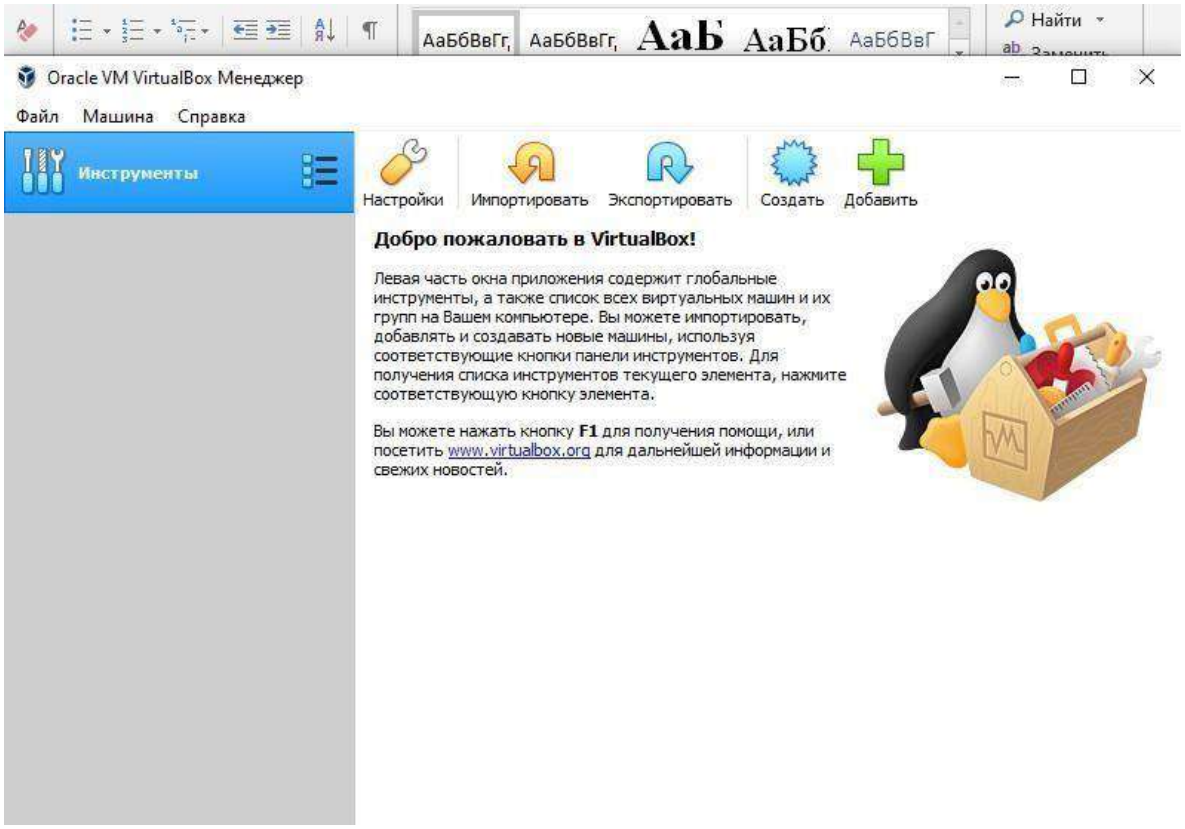


Будет предложено запустить процесс загрузки программы. Нажимаем «Install.



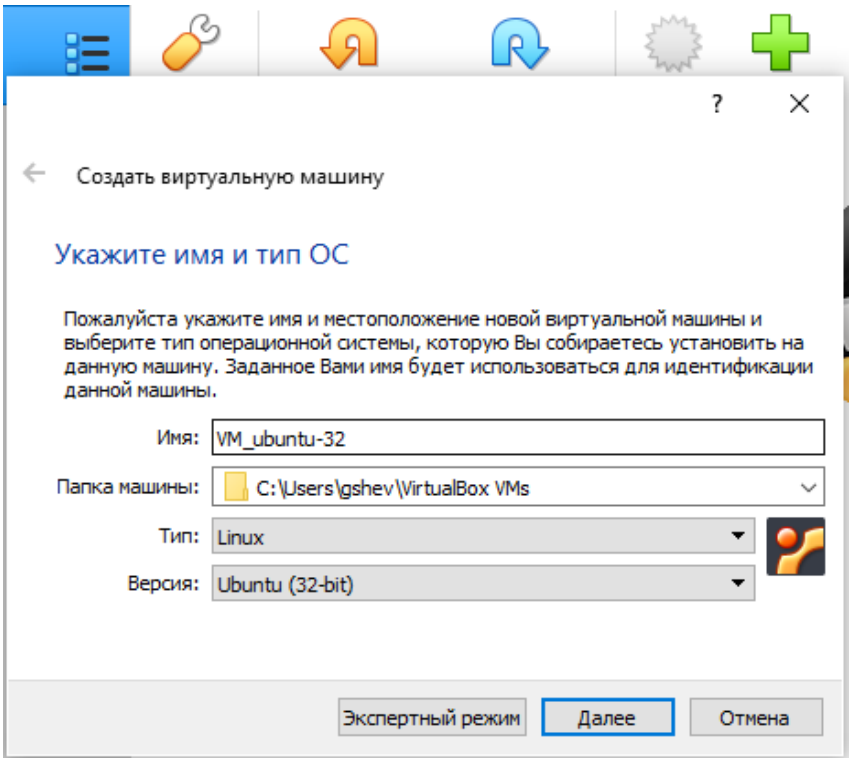
Далее нажимаем на «Finish».

Платформа виртуализации установлена.

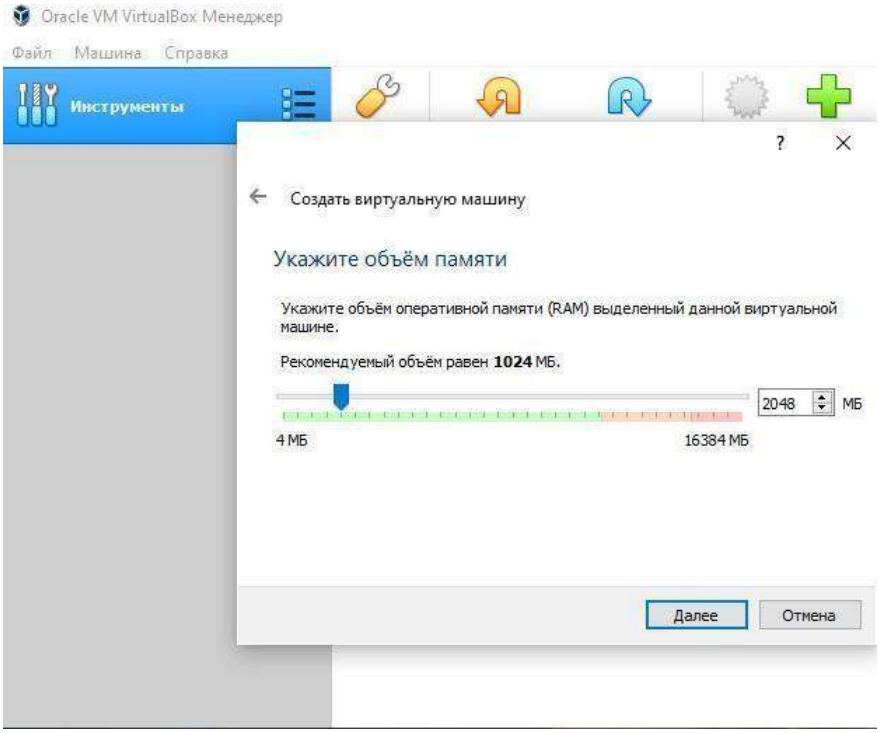


Создание и настройка виртуальной машины VirtualBox с ОС Ubuntu

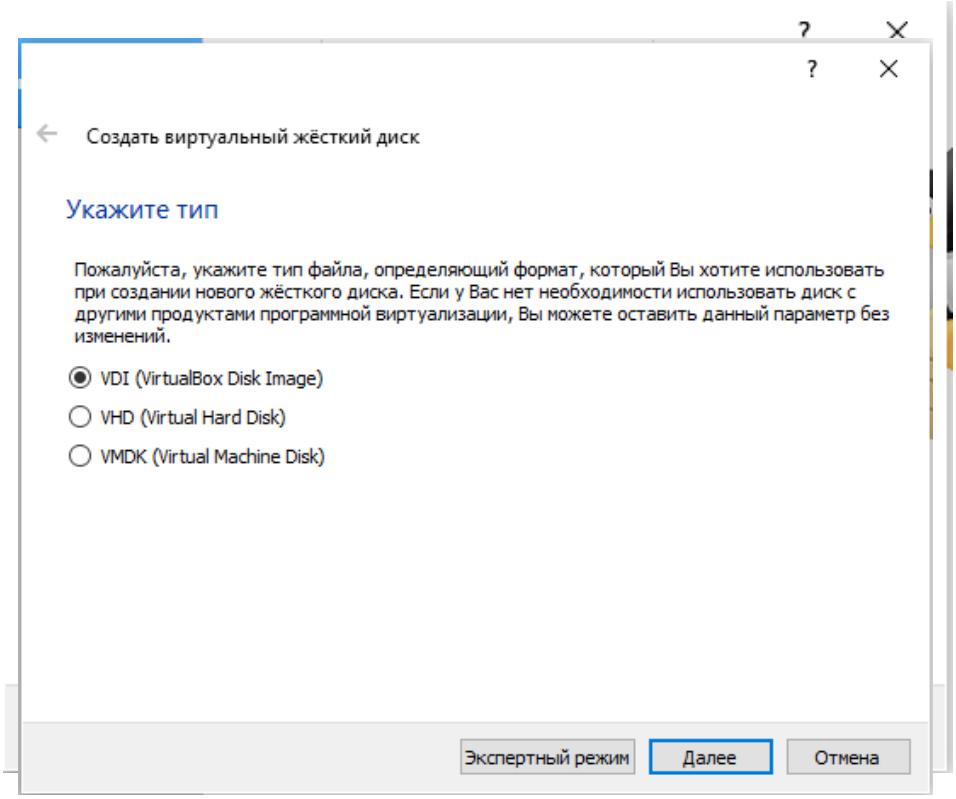
В окне программы нажимаем «Создать».



Пропишите имя, тип и версию. Например, VM_Ubuntu-32. По нему вы будете идентифицировать систему. Поэтому создавайте его информативным. Будем устанавливать 32-битную версию Ubuntu desktop. Следовательно, версия: Ubuntu (32-бит).

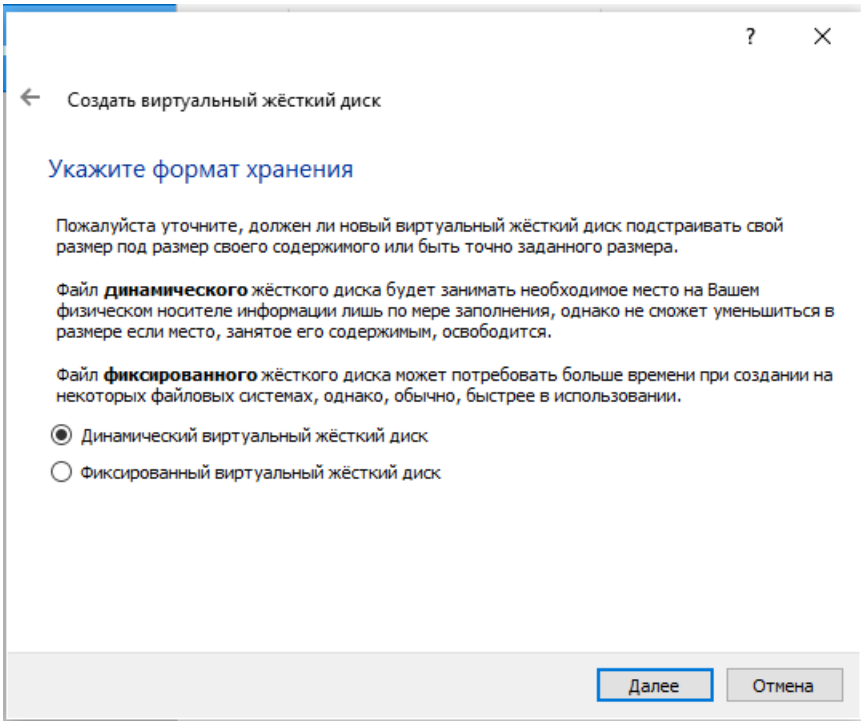


Нажимаем «Далее». Пропишите объем ОЗУ, который вы отдадите виртуальной машине. Например, хотя бы 1 ГБ. Укажите объем, перемещая специальный ползунок, или пропишите вручную. После выбора нажимаем «Далее».



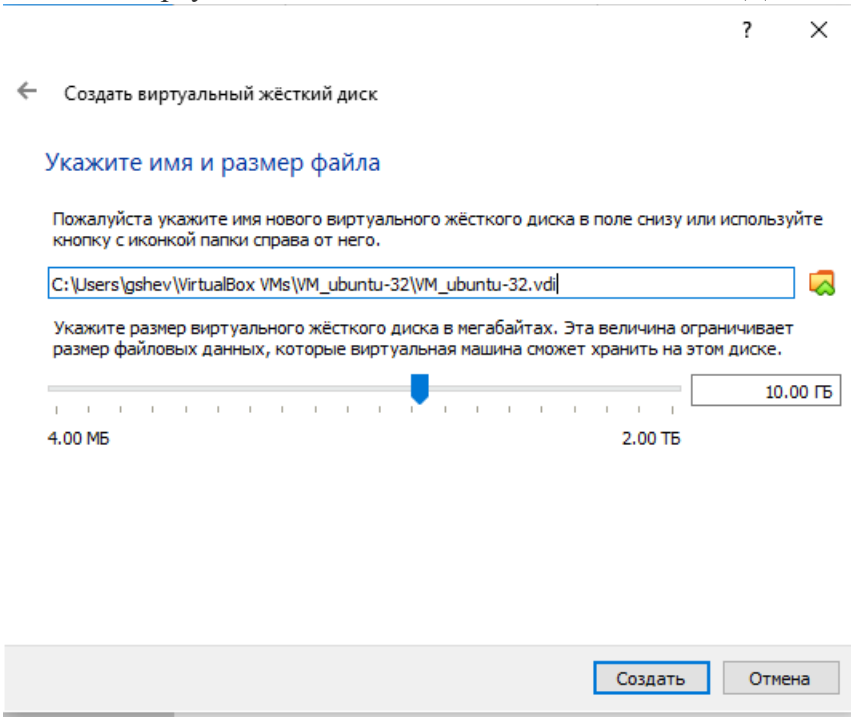
Создаем жесткий диск. Если рассматривать его физически, это файл, хранящийся на одном из разделов на вашем HDD. Программа предлагает три варианта. Выбираем 2-й вариант: создать новый виртуальный жесткий диск.

Нажимаем «Создать». Определяемся с типом виртуального HDD. Выбираем тип VDI, т.е. первый вариант. Затем выбираем формат хранения: динамический виртуальный жесткий диск.



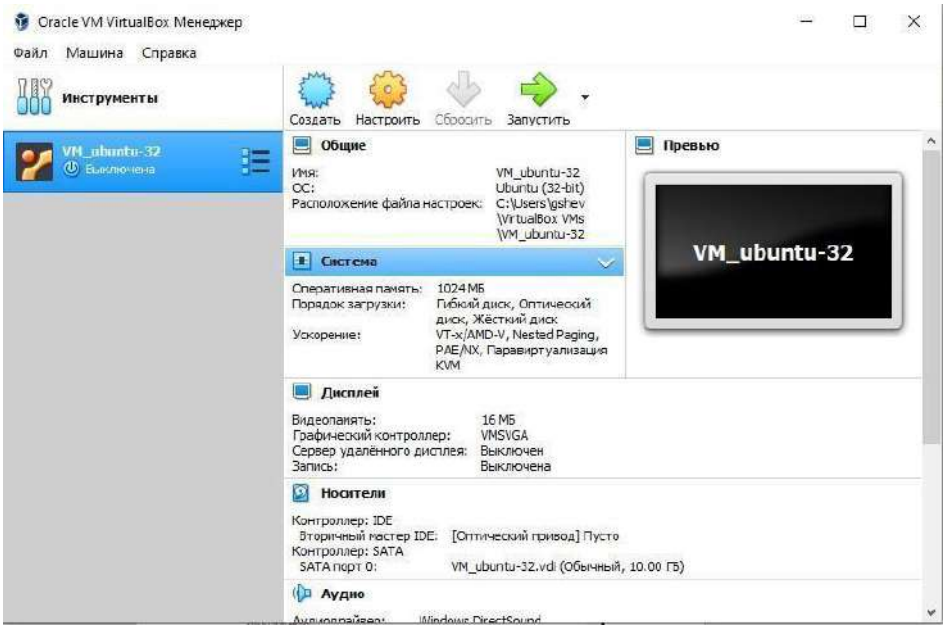
Укажите имя и объем виртуального HDD, например, VM_ubuntu-32. При необходимости

укажите другое место хранения диска. Для этого надо кликнуть кнопку справа от поля ввода. На шкале укажите объем виртуального жесткого диска в мегабайтах. Для этого перетащите



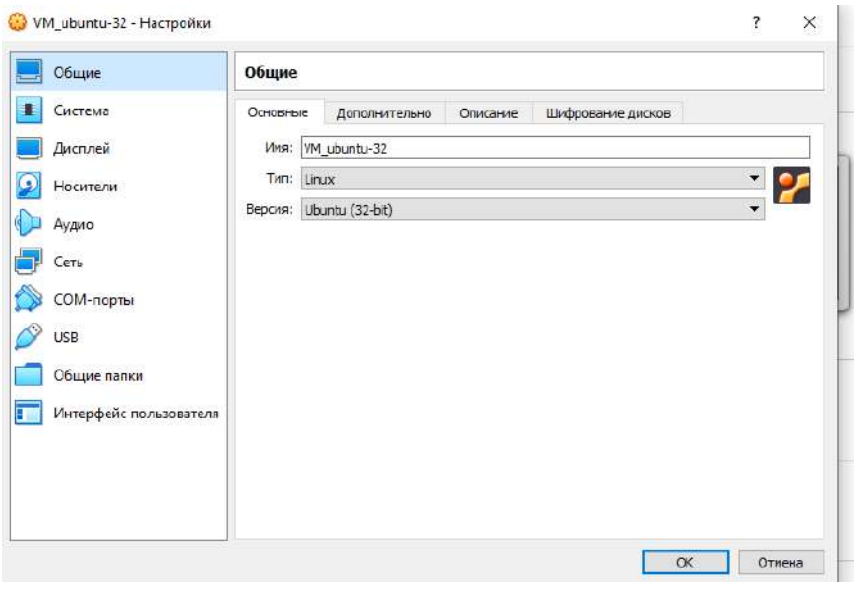
ползунок на нужное место, соответствующее вашим потребностям, например, 10 ГБ. После выбора настроек, нажмите на кнопку «Создать».

Виртуальная машина создана. После этого откроется главное окно «Oracle VM VirtualBox Менеджер» с вновь созданной виртуальной машиной. В правой части окна вы можете ознакомиться с некоторыми параметрами виртуальной машины.

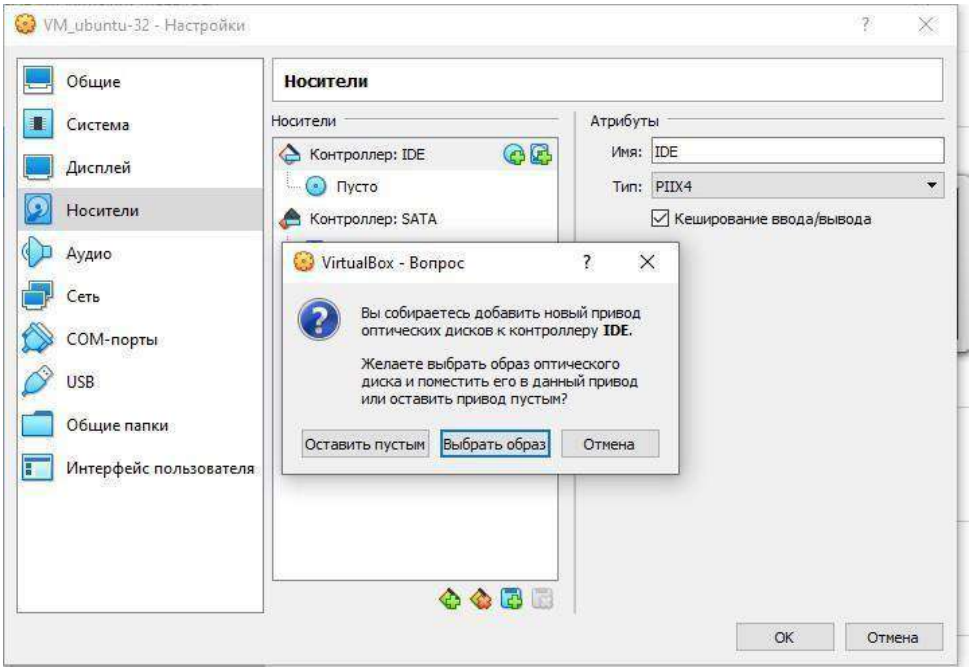


Теперь необходимо будет сделать дополнительные настройки перед установкой операционной системы Ubuntu на виртуальную машину.
В главном окне VirtualBox нажмите на кнопку «Настроить» для входа в настройки этой виртуальной машины:

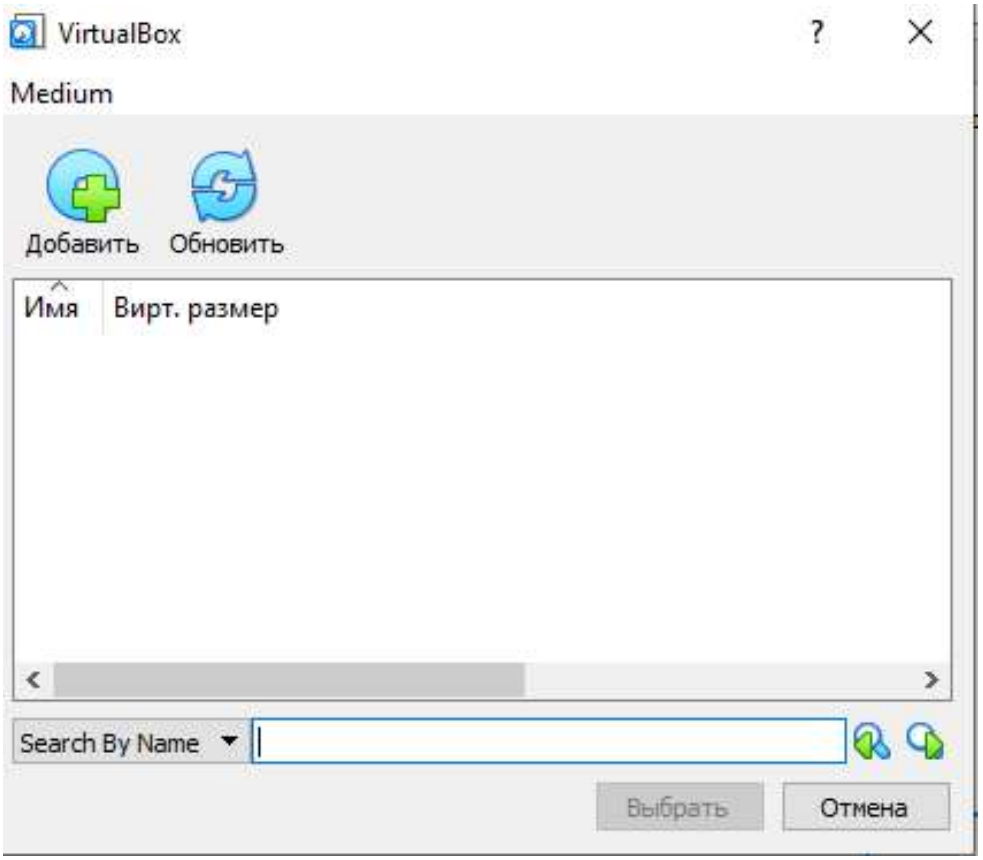
1. Для установки операционной системы на виртуальной машине потребуется загрузка с установочного диска. В среде VirtualBox имеется возможность выполнения загрузки с использованием виртуального привода, создаваемого на основе образа загрузочного диска. При первом запуске виртуальной машины, когда еще нет установленной гостевой операционной системы, VirtualBox предложит выбрать устройство загрузки.



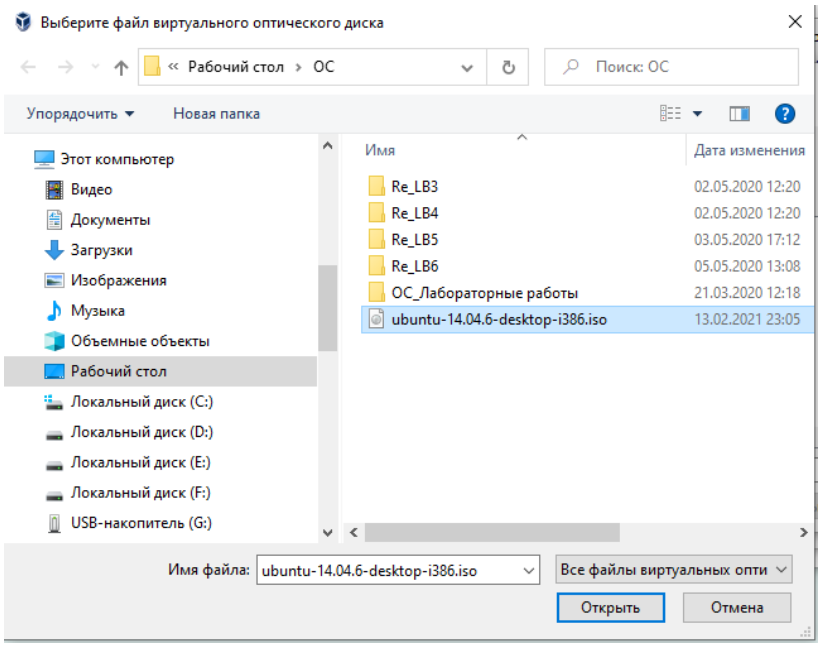
Выбрать пункт «Носители»



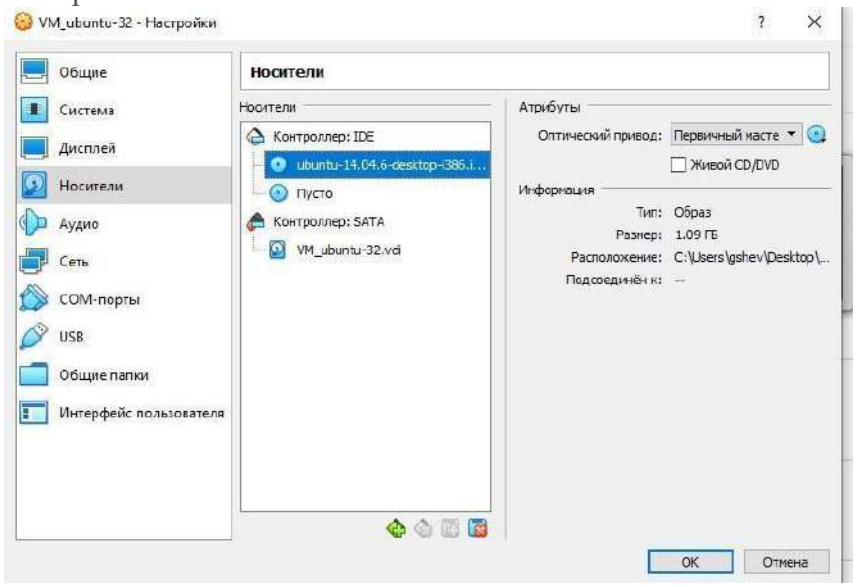
Добавить привод оптического диска (нажать на 1-ю кнопку справа от контроллера: IDE).
Затем «Выбрать образ»



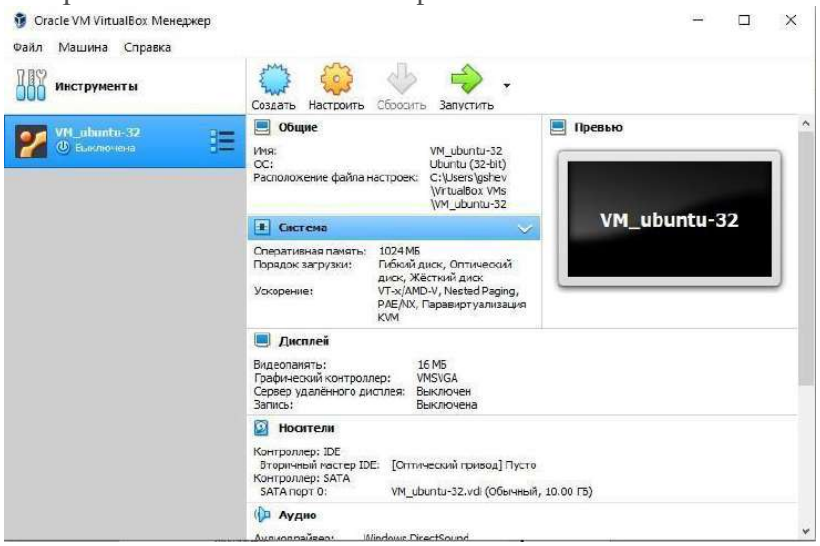
Выбрать кнопку «Добавить».
Указать путь на своем компьютере к образу Ubuntu.



«Открыть»



Выбрать ubuntu 14.04.6-desktop-i386… OK.



«Запустить» Oracle VM VirtualBox

Выберите язык из списка в левой части окна. Нажмите «Установить Ubuntu». Начнется

загрузка ОС.

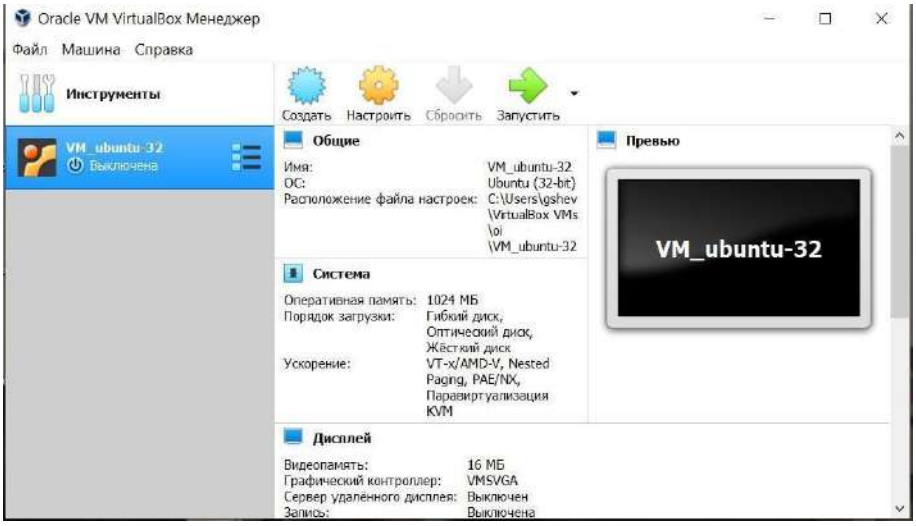
Процесс установки гостевой ОС ничем не отличается от установки на реальной машине. Можно выбрать язык для устанавливаемой системы (обычно Русский), часовой пояс, раскладку клавиатуры и т. п. Большинство параметров можно оставить по умолчанию, в том числе и Тип установки.

Установщик Linux предостерегает вас от ошибочных действий. Ознакомьтесь с представленной вам информацией и смело жмите «Установить сейчас». процессе установки необходимо задать имя компьютера, пользователя, пароль и режим входа в систему (лучше выбрать «Входить в систему автоматически»).

Дальнейшая установка Ubuntu выполняется без какого-либо вмешательства пользователя и завершается предложениям перезагрузить компьютер. По сравнению с установкой системы на реальном компьютерном оборудовании, установка на виртуальной машине выполняется медленнее, что вполне ожидаемо. Степень снижения производительности в основном, зависит от быстродействия оборудования реального компьютера.
При первой загрузке вновь установленной операционной системы, диспетчер VirtualBox автоматически отключит виртуальный привод на основе образа диска с дистрибутивом Ubuntu, загрузка будет выполнена с виртуального жесткого диска и по ее завершению, на экране отобразится приглашение ко входу в систему.

2. Настроим VirtualBox так, что в ней можно будет работать с флешкой.

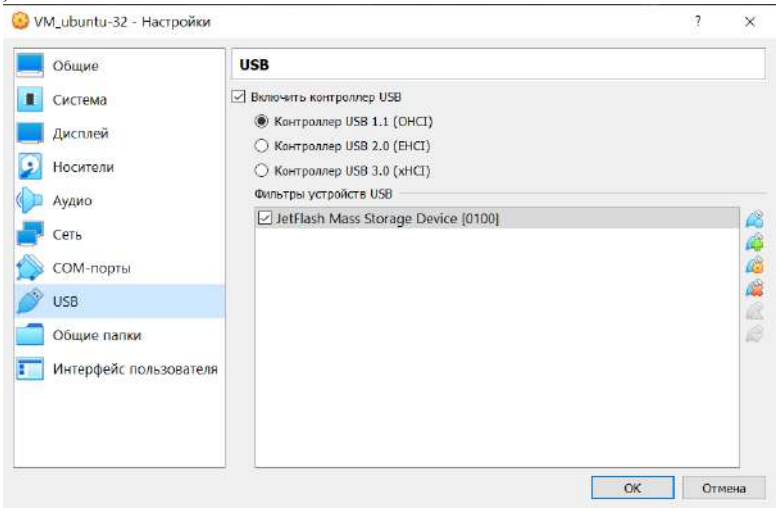
Подключаем флешку (надо знать ее имя) к компьютеру и запускаем VirtualBox, затем жмём на кнопку «Настроить».



Выбираем Категорию USB.

Отмечаем пункты Включить контроллер USB Включить
контроллер USB 1.1 (OHCI)
В окне Фильтры устройств USB жмём на + (справа второй сверху) и выбираем нашу флешку (в

данном случае JetFlash…), отмечаем её.



ПРАКТИЧЕСКОЕ ЗАНЯТИЕ № 4.

Анализ и защита виртуальной среды: установка и тестирование антивируса в Windows 10 под VirtualBox

Цель занятия: Исследовать эффективность защиты виртуальный среды на базе Windows 10 в VirtualBox

Сначала установим Windows на VirtualBox .Это простой процесс. Вот пошаговая инструкция:

1. Подготовка

- Скачайте образ Windows (ISO):
https://www.microsoft.com/software-download (для Windows 10/11).
- Или использовать собственный ISO-файл.

2. Создание виртуальной машины (ВМ)

1. Запустите VirtualBox → нажмите "Создать".
2. Укажите параметры:
   - Имя: Например, "Windows 10"
   - Тип: Microsoft Windows
   - Версия: Выберите нужную (Windows 10/11, 32/64-bit)
3. Выделите оперативной памяти (RAM):
   - Рекомендуется 2048 МБ (2 ГБ) или больше для Windows 10/11.
4. Создайте виртуальный жесткий диск:
   - Выберите "Создать новый виртуальный жесткий диск" → VDI → Динамический (рекомендуется).
   - Размер: Не менее 32 ГБ для Windows 10/11.

3. Настройка виртуальной машины

1. Выберите созданную ВМ → "Настроить".
2. Дисплей → Видеопамять:
   - Увеличьте до 128 МБ (для лучшей производительности).
3. Носители → Контроллер: IDE:
   - Нажмите на "Пусто" → Значок диска → Выберите образ ISO.
4. Общие папки (опционально):
   - Чтобы обмениваться файлами с хост-системой, в "Общие папки" укажите нужную папку.

Если после создания виртуальной машины (ВМ) с Windows 10 в VirtualBox вы видите в разделе "Носители" только:
- Контроллер SATA с диском `Windows 10.vdi`
- и пункт "Пусто" под ним,
это нормально — но если вы хотите установить Windows 10, нужно прикрепить ISO-образ системы к виртуальному приводу. Вот как это сделать:

---
1. Подключение ISO-образа Windows 10
1. Выберите вашу ВМ в списке → "Настроить".
2. Перейдите в раздел "Носители".
3. В "Контроллере: SATA" вы увидите:
   - `Windows 10.vdi` (это ваш виртуальный жесткий диск).
   - "Пусто" (это виртуальный дисковод).

4. Кликните на "Пусто" → справа нажмите на значок диска (💿) → "Выбрать образ диска".
5. Найдите ваш ISO-образ Windows 10 (скачанный ранее) и откройте его.
6. Теперь вместо "Пусто" будет название вашего ISO-файла.
7. Нажмите "OK", чтобы сохранить настройки.

4. Запуск и установка Windows
1. Запустите ВМ → начнется загрузка с ISO.
2. Следуйте инструкциям установщика Windows:
   - Выберите язык → "Установить".
   - Введите ключ (или пропустите, если нет).
   - Выберите "Выборочная установка".
   - Отформатируйте виртуальный диск (если нужно) → "Далее".
3. Дождитесь завершения установки.

5. Установка дополнений гостевой ОС
После установки Windows:
1. В меню VirtualBox выберите "Устройства" → "Подключить образ диска Дополнений гостевой ОС".
2. Откройте "Этот компьютер" в гостевой Windows → запустите установку с виртуального диска.
3. Перезагрузите ВМ. 🚀
Теперь можете установить **антивирусную программу** на гостевую операционную систему (Windows) в VirtualBox. Это даже рекомендуется, если вы:
- Тестируете подозрительные файлы/программы (чтобы не заразить основную систему).
- Проверяете работу антивируса в изолированной среде.
- Имитируете реальную систему для обучения или экспериментов.

[WINDOWS 10]
🔽 Выбираем правильный вариант для VirtualBox
1. Запустите скачанный инструмент (MediaCreationTool.exe).
2. На вопрос *"Что вы хотите сделать?"* выберите:
   → "Создать установочный носитель (USB-устройство флэш-памяти, DVD-диск или ISO-файл)"
   (этот вариант нужен для VirtualBox!)

🔽 Создаём ISO-образ
3. В следующем окне:

- Снимите галочку *"Использовать рекомендуемые параметры"* (если хотите выбрать конкретную версию Windows)
- Выберите:
  • Язык: Русский
  • Выпуск: Windows 10
  • Архитектура: 64-разрядная (x64)
4. Нажмите "Далее"

5. Выберите "ISO-файл" (VirtualBox работает именно с ISO!)
6. Укажите, куда сохранить (например, в "Загрузки" или на Рабочий стол)
7. Дождитесь завершения загрузки (размер ~5 ГБ)

🔽 Что дальше?
Теперь у вас есть ISO-файл Windows 10. Вернитесь в VirtualBox и:
1. В настройках виртуальной машины зайдите в "Носители"
2. Нажмите на "Пусто" → значок диска → Выберите ваш ISO
3. Запустите ВМ - начнётся установка Windows!

⚠ Важно!
- Если выбирать "Обновить этот компьютер сейчас" - система установится на ваш реальный ПК, а не в VirtualBox!
- Для виртуальной машины всегда выбираем ISO-файл, а не USB/DVD.

Нужна помощь с настройкой VirtualBox после этого? ☺

Время загрузки ISO-образа Windows 10 через Media Creation Tool зависит от скорости вашего интернета. Вот ориентировочные сроки:

 Примерное время загрузки:
Скорость интернета	Приблизительное время
10 Мбит/с (~1.2 МБ/с)	~1.5-2 часа
50 Мбит/с (~6 МБ/с)	~20-30 минут
100 Мбит/с (~12 МБ/с)	~10-15 минут
300 Мбит/с (~37 МБ/с)	~3-5 минут

🔍 Что влияет на скорость:
1. Загрузка серверов Microsoft (вечером может быть медленнее из-за нагрузки)
2. Ваше интернет-соединение (проводное обычно стабильнее Wi-Fi)
3. Антивирусы/фаерволы (могут немного замедлять процесс)

🛠 Как ускорить процесс:
- Закройте торренты и видеостриминги
- Отключите VPN (если не требуется)
- Подключите компьютер по кабелю (Ethernet)

✅ Как проверить прогресс:
1. В окне Media Creation Tool будет полоса прогресса и проценты
2. В проводнике можно посмотреть размер частично скачанного файла (полный ISO весит ~5.5 ГБ)

Если загрузка зависла на долгое время (более 30 минут без движения прогресса) - лучше перезапустите инструмент

Как установить антивирус на Windows в VirtualBox?

1. Скачайте антивирус:

- Бесплатные:

Avast Free,
AVG,
Kaspersky Free,
Bitdefender Free,
Microsoft Defender (встроен в Windows 10/11).

- Платные (если есть лицензия):

Kaspersky,
ESET,
Norton, Malwarebytes и др.

2. Способы загрузки антивируса в гостевую ОС:

- Через интернет (если есть сеть в ВМ):
  - Просто откройте браузер в гостевой Windows и скачайте антивирус с официального сайта.
- Через общую папку (если нет интернета):
  1. Настройте "Общие папки" в VirtualBox (в свойствах ВМ).
  2. Скопируйте установочный файл антивируса на хост-системе в эту папку.
  3. В гостевой Windows откройте общую папку и запустите установку.
- Через ISO-образ:
  - Можно создать ISO с антивирусом и подключить его как виртуальный диск в VirtualBox.

3. Установите и настройте антивирус:

- Запустите установщик → следуйте инструкциям.
- После установки обновите базы (если есть интернет).
- Настройте сканирование по расписанию (если нужно).

Нужен ли антивирус в виртуальной машине?

- Да, если вы:
  - Работаете с опасным ПО или посещаете подозрительные сайты.
  - Хотите протестировать вирусы в безопасной среде.
- Нет, если вы:
  - Используете ВМ только для тестов и после каждого сеанса возвращаетесь к чистому снэпшоту.
  - Не выходите в интернет и не загружаете файлы извне.

Советы по безопасности

✓ Делайте снэпшоты перед рискованными действиями (чтобы быстро откатиться).

✓ Отключайте общие папки, если тестируете вирусы (чтобы не заразить хост).

✓ Используйте "Режим песочницы" в некоторых антивирусах (например, Kaspersky, Avast).

ТЕМА: ЗАЩИТА ДАННЫХ, УПРАВЛЕНИЕ ДОСТУПОМ И АНАЛИЗ БЕЗОПАСНОСТИ В WINDOWS 10

Цель лабораторной работы:

1. Ознакомление студентов с основными механизмами безопасности в Windows 10.
2. Обучение основам управления паролями, шифрования данных и обнаружения вредоносных программ.
3. Практическое применение инструментов для анализа безопасности в ОС.

Оборудование:

- Компьютеры с операционной системой Windows 10.
- Доступ в Интернет для загрузки инструментов и утилит.

Ход работы:

Часть 1. Настройка политики безопасности паролей

Цель: Научить студентов настраивать политику паролей в Windows 10 для улучшения безопасности.

Шаг 1: Открытие локальной политики безопасности

1. Нажмите **Win + R**, введите **secpol.msc** и нажмите Enter.
2. Откроется окно «Локальная политика безопасности».
3. Перейдите в раздел **Политики учётных записей → Политика паролей**.

Шаг 2: Настройка требований к паролям

1. Убедитесь, что следующие параметры настроены:
    - **Минимальная длина пароля**: установите значение, например, 8 символов.
    - **Сложность пароля**: включите требование для пароля содержать хотя бы одну цифру, заглавную и строчную буквы.
    - **Максимальный срок действия пароля**: установите на 30 дней.

Шаг 3: Применение настроек

1. Перезагрузите систему или выполните команду **gpupdate /force** в командной строке для применения изменений.

Шаг 4: Проверка паролей

1. Попробуйте создать учетную запись с паролем, не соответствующим установленным требованиям, и убедитесь, что система отклоняет такие пароли.

Часть 2. Шифрование данных с помощью BitLocker

Цель: Ознакомить студентов с шифрованием дисков в Windows 10 с использованием BitLocker.

Шаг 1: Включение BitLocker

1. Откройте **Панель управления → Система и безопасность → Шифрование дисков BitLocker**.

2. Найдите диск, который хотите зашифровать (например, диск C:) и нажмите «Включить BitLocker».
3. Выберите способ разблокировки: через пароль или USB-накопитель.
4. Следуйте инструкциям на экране, чтобы завершить процесс шифрования.

Шаг 2: Тестирование шифрования

1. После завершения шифрования попытайтесь получить доступ к данным на диске. Данные должны быть доступны только после ввода пароля или подключения USB-накопителя с ключом.

Шаг 2: Тестирование шифрования с помощью **BitLocker** в Windows 10 включает проверку того, что данные на зашифрованном диске могут быть доступны только после ввода пароля или подключения USB-накопителя с ключом. Вот подробное руководство:

1. Включение шифрования BitLocker (если еще не выполнено)

Если вы ещё не активировали BitLocker, следуйте этим шагам, чтобы зашифровать диск:

1. Нажмите **Win + X** и выберите **Панель управления**.
2. Перейдите в **Система и безопасность → Шифрование дисков BitLocker**.
3. Выберите диск, который хотите зашифровать (например, диск C:), и нажмите **Включить BitLocker**.
4. Выберите метод разблокировки:
    - **Через пароль**: задайте пароль, который потребуется для доступа к данным.
    - **Через USB-накопитель**: выберите USB-накопитель для использования в качестве ключа для разблокировки.
5. Следуйте инструкциям на экране, чтобы завершить процесс шифрования. Это может занять некоторое время в зависимости от объема данных на диске.

После завершения процесса шифрования данные на диске будут защищены и доступны только через выбранный способ разблокировки.

2. Тестирование шифрования

После того как BitLocker завершит процесс шифрования, выполните следующие действия для тестирования, чтобы убедиться, что доступ к данным возможен только после ввода пароля или подключения USB-ключа.

1. Перезагрузите компьютер или выйдите из текущей учетной записи

- Если вы установили **пароль** для разблокировки, при следующем запуске или после выхода из системы вам будет предложено ввести этот пароль.
- Если вы использовали **USB-накопитель** как ключ для разблокировки, подключите его к компьютеру. Без него система не загрузится.

2. Попробуйте получить доступ к данным без ввода пароля или подключения USB-ключа

- **Без пароля**: Перезагрузите систему и попытайтесь войти на зашифрованный диск. Если пароль не был введен, система не позволит вам получить доступ к данным.

- **Без USB-накопителя**: Если BitLocker был настроен на использование USB-накопителя для разблокировки, без подключения этого устройства компьютер не загрузится, и вы не сможете получить доступ к данным на диске.

3. Введите правильный пароль или подключите USB-накопитель

- Если вы выбрали метод с **паролем**, при загрузке системы будет предложено ввести пароль для доступа к зашифрованному диску. Введите его, и система загрузится с доступом к данным.
- Если вы выбрали метод с **USB-накопителем**, подключите накопитель в порт, и система автоматически разблокирует диск, позволяя вам получить доступ к данным.

4. Попробуйте получить доступ к файлам

После того как вы правильно разблокируете диск (введите пароль или подключите USB-ключ), данные на нем станут доступны как обычно. Вы сможете открыть файлы, работать с ними, копировать или перемещать их.

3. Попробуйте доступ к диску, если забыт пароль или нет USB-ключа

Если вы забудете пароль или потеряете USB-накопитель, диск останется зашифрованным, и доступ к данным будет невозможен без восстановления ключа восстановления. Поэтому крайне важно **сохранить ключ восстановления** в безопасном месте, когда вы настраиваете BitLocker.

Для восстановления ключа:

- В процессе настройки BitLocker вам будет предложено сохранить ключ восстановления. Если вы его сохранили, используйте его для восстановления доступа к данным. Вы также можете сохранить ключ в вашем аккаунте Microsoft или на USB-накопителе.

Таким образом, тестирование шифрования BitLocker позволяет убедиться, что данные на зашифрованном диске защищены и доступны только при наличии правильного пароля или USB-ключа.

Шаг 3: Отключение BitLocker (по желанию)

1. Для отключения BitLocker вернитесь в окно «Шифрование дисков BitLocker» и выберите «Отключить BitLocker».

Часть 3. Использование встроенного антивируса Windows Defender для сканирования на вредоносные программы

Цель: Научить студентов использовать антивирусные средства для поиска и устранения угроз.

Шаг 1: Открытие Windows Defender

1. Перейдите в **Параметры → Обновление и безопасность → Безопасность Windows**.
2. Нажмите на **Защита от вирусов и угроз**.

Шаг 2: Запуск полного сканирования системы

1. В разделе **Параметры сканирования** выберите **Полное сканирование**.
2. Нажмите **Сканировать сейчас** и дождитесь завершения процесса.

Шаг 3: Обзор отчета сканирования

1. После завершения сканирования посмотрите результаты. Если были обнаружены угрозы, попытайтесь устранить их, выбрав действие (например, «Удалить» или «Поместить в карантин»).

Шаг 4: Симуляция обнаружения вируса

1. Для учебных целей можно использовать безопасные тестовые файлы с вирусами, такие как **EICAR test file**, чтобы убедиться в правильности работы антивируса.

Часть 4. Мониторинг безопасности с помощью журналов событий

Цель: Научить студентов работать с журналами событий для анализа активности в системе.

Шаг 1: Открытие журнала событий

1. Нажмите **Win + R**, введите **eventvwr.msc** и нажмите Enter.
2. Откроется окно «Просмотр событий».

Шаг 2: Просмотр системных журналов

1. Перейдите в **Журналы Windows → Система** или **Применение**, чтобы просматривать события системы.
2. Посмотрите, есть ли какие-либо записи, связанные с ошибками безопасности (например, неудачные попытки входа в систему).

Шаг 3: Настройка уведомлений

1. Для мониторинга можно настроить уведомления для определенных событий, например, неудачных попыток входа.
2. В разделе «Управление журналами событий» выберите «Создать задачу» и настройте уведомления для определенных типов событий.

Часть 5. Резервное копирование и восстановление данных

Цель: Ознакомить студентов с процессом создания резервных копий и восстановления данных.

Шаг 1: Создание резервной копии

1. Перейдите в **Панель управления → Резервное копирование и восстановление (Windows 7)**.
2. Выберите **Настроить резервное копирование** и следуйте инструкциям для создания резервной копии данных.

Шаг 2: Восстановление данных

1. Для восстановления данных используйте меню восстановления из резервной копии и выберите нужные файлы или полный образ системы.

Чтобы создать учетную запись в Windows 10 с паролем, не соответствующим установленным требованиям безопасности, выполните следующие шаги:

1. Откройте окно Управление компьютером

1. Нажмите **Win + X** и выберите **Управление компьютером** из контекстного меню.
2. В открывшемся окне выберите **Локальные пользователи и группы → Пользователи**.

2. Создайте новую учетную запись

1. В правой части окна кликните правой кнопкой мыши по пустому месту и выберите **Новый пользователь**.
2. Введите имя пользователя и пароль. Для примера используйте пароли, которые **не соответствуют требованиям безопасности** (например, слишком короткие пароли или без использования цифр и заглавных букв).
    - Например, используйте такие пароли, как:
        - `12345` (слишком короткий)
        - `password` (нет цифр или заглавных букв)
3. Нажмите **Создать**.

3. Убедитесь, что система отклоняет неподобающие пароли

Теперь, когда учетная запись создана, Windows должна отклонить пароли, которые не соответствуют установленным правилам безопасности.

Примечание:

- Если вы установили требования для пароля, как указано в предыдущем шаге (например, минимальная длина 8 символов, обязательное использование цифр и заглавных букв), при попытке ввести некорректный пароль система должна отобразить ошибку, например: "Пароль не удовлетворяет требованиям политики безопасности."

Таким образом, Windows 10 будет предотвращать создание учетных записей с паролями, не соответствующими установленным в локальной политике безопасности.

Чтобы проверить, была ли успешно создана учетная запись в Windows 10, вы можете воспользоваться несколькими методами. Вот пошаговое руководство:

Метод 1: Проверка через Управление компьютером

1. Нажмите **Win + X** и выберите **Управление компьютером**.
2. В открывшемся окне в левой части выберите **Локальные пользователи и группы → Пользователи**.
3. В правой части окна отобразится список всех пользователей на вашем компьютере. Найдите учетную запись, которую вы создали.

Если ваша учетная запись присутствует в списке, значит, она была успешно создана.

Метод 2: Проверка через Панель управления

1. Откройте **Панель управления** (для этого нажмите **Win + R**, введите **control** и нажмите Enter).
2. Перейдите в раздел **Учетные записи пользователей**.
3. Здесь вы увидите список всех учетных записей, зарегистрированных на вашем компьютере.

Метод 3: Проверка через Командную строку

1. Нажмите **Win + R**, введите **cmd** и нажмите Enter для открытия командной строки.
2. В командной строке введите следующую команду и нажмите Enter:

   ```
   net user
   ```

3. Команда выведет список всех учетных записей, созданных на компьютере. Найдите учетную запись, которую вы создали.

Если она отображается в списке, значит, учетная запись была успешно добавлена.

Метод 4: Проверка через Настройки Windows (для стандартных учетных записей Microsoft)

1. Нажмите **Win + I** для открытия настроек.
2. Перейдите в раздел **Учетные записи → Семья и другие пользователи**.
3. В этом разделе будет отображен список всех учетных записей, включая локальные и учетные записи Microsoft.

Если учетная запись отображается здесь, значит, она была успешно создана.

Метод 5: Вход в систему

1. Перезагрузите компьютер или выйдите из текущей учетной записи.
2. На экране входа в систему проверьте, доступна ли учетная запись, которую вы создали. Если она присутствует, значит, она была успешно добавлена.

Используя один из этих методов, вы сможете проверить, была ли учетная запись успешно создана в вашей системе Windows 10.

Чтобы удалить учетную запись в Windows 10, выполните несколько простых шагов. Вот подробное руководство:

Метод 1: Удаление учетной записи через Панель управления

1. **Откройте Панель управления**:
   - Нажмите **Win + R**, введите **control** и нажмите Enter.
2. Перейдите в раздел **Учетные записи пользователей**:
   - Выберите **Удалить учетную запись**.
3. Выберите учетную запись, которую вы хотите удалить:
   - В списке учетных записей найдите ту, которую хотите удалить, и нажмите на нее.

4. **Удалите учетную запись**:
    - Нажмите **Удалить учетную запись**.
    - Вам будет предложено выбрать, сохранить ли файлы пользователя или удалить их. Выберите нужный вариант:
        - **Сохранить файлы** — файлы пользователя будут перемещены в папку **C:\Users<имя пользователя>**.
        - **Удалить файлы** — все файлы пользователя будут удалены, включая документы и настройки.
5. Подтвердите удаление:
    - После этого учетная запись будет удалена, и вы вернетесь в список всех пользователей.

Метод 2: Удаление учетной записи через Управление компьютером

1. **Откройте Управление компьютером**:
    - Нажмите **Win + X** и выберите **Управление компьютером**.
    - В левой части окна выберите **Локальные пользователи и группы → Пользователи**.
2. Найдите учетную запись, которую хотите удалить:
    - В правой части окна выберите учетную запись, которую хотите удалить.
3. **Удалите учетную запись**:
    - Щелкните правой кнопкой мыши на нужной учетной записи и выберите **Удалить**.
4. Подтвердите удаление:
    - Подтвердите, что хотите удалить учетную запись. Если это учетная запись без пароля, вам будет предложено подтвердить удаление.

Метод 3: Удаление учетной записи через Настройки Windows

1. Нажмите **Win + I**, чтобы открыть **Настройки**.
2. Перейдите в раздел **Учетные записи**:
    - Выберите **Семья и другие пользователи**.
3. Выберите учетную запись для удаления:
    - В разделе **Другие пользователи** выберите учетную запись, которую хотите удалить.
4. **Удалите учетную запись**:
    - Нажмите на учетную запись и выберите **Удалить**.
5. Подтвердите удаление учетной записи:
    - Вы получите предупреждение о том, что все данные пользователя будут удалены. Подтвердите удаление, выбрав **Удалить учетную запись и данные**.

Метод 4: Удаление учетной записи с помощью Командной строки

1. Откройте **Командную строку** с правами администратора:
    - Нажмите **Win + X** и выберите **Командная строка (администратор)** или **Windows PowerShell (администратор)**.
2. Введите команду для удаления учетной записи:
    - Чтобы удалить учетную запись, введите следующую команду, заменив `<имя пользователя>` на имя учетной записи:

```
net user <имя пользователя> /delete
```

3. Нажмите **Enter**. Учетная запись будет удалена.

После выполнения этих шагов выбранная учетная запись будет удалена из вашей системы Windows 10.

Заключение:

По завершении лабораторной работы студенты:

- Ознакомятся с основными методами защиты данных в Windows 10, такими как управление паролями, шифрование и использование антивируса.
- Получат практические навыки работы с инструментами мониторинга безопасности и резервного копирования.
- Освоят использование встроенных инструментов для анализа и устранения угроз безопасности.

Такая лабораторная работа поможет студентам понять, как применять реальные средства защиты и анализировать безопасность в операционных системах.

ПРАКТИЧЕСКАЯ РАБОТА № 5.
ХЕШИРОВАНИЕ

Цель работы:

Изучить основные принципы хеширования, познакомиться с алгоритмами хеширования, реализовать простейшие функции хеширования на Python и проанализировать их свойства.

Теоретическая часть:

Хеширование — это процесс преобразования входных данных произвольной длины в фиксированную строку (хеш) с использованием специальной функции (хеш-функции). Хеширование широко используется в информационной безопасности для проверки целостности данных, хранения паролей и других задач.

Основные свойства хеш-функций:

1. **Детерминированность**: Для одних и тех же входных данных хеш-функция всегда возвращает одинаковый результат.
2. **Быстрота вычисления**: Хеш-функция должна быстро вычисляться для любых входных данных.
3. **Стойкость к коллизиям**: Вероятность того, что два разных входных данных дадут одинаковый хеш, должна быть минимальной.
4. **Необратимость**: По хешу должно быть невозможно восстановить исходные данные.

Практическая часть:

Задание 1: Реализация простейшей хеш-функции

Описание задачи:

В этом задании вам нужно реализовать простейшую хеш-функцию, которая работает следующим образом:

1. Для каждого символа входной строки вычисляется его ASCII-код (числовое значение символа).
2. Все ASCII-коды суммируются.
3. Результат суммирования берется по модулю 100, чтобы получить хеш-значение в диапазоне от 0 до 99.

Пример:

Возьмем строку "hello":

- ASCII-коды символов:
  - h = 104
  - e = 101
  - l = 108
  - l = 108
  - o = 111
- Сумма ASCII-кодов: 104 + 101 + 108 + 108 + 111 = 532
- Хеш-значение: 532 % 100 = 32 (остаток от деления

на 100) Таким образом, для строки "hello" хеш-значение

будет равно **32**. **Код:**

```
def simple_hash(input_string):
    hash_value = 0
    for char in input_string:
        hash_value += ord(char)  # ord() возвращает ASCII-код символа
    return hash_value % 100
# Пример использования
input_data = "hello"
print(f"Хеш для '{input_data}': {simple_hash(input_data)}")
```

Вывод:

```
Хеш для 'hello':
32
```

Анализ:
Эта хеш-функция очень простая и не обладает высокой стойкостью к коллизиям. Например, строки "hello" и "ohell" дадут одинаковый хеш, так как сумма ASCII- кодов будет одинаковой.

Задание 2: Использование встроенной функции

hash() Описание задачи:

В Python есть встроенная функция hash (), которая возвращает хеш-значение для объекта. Хеш-значение — это целое число, которое используется для быстрого

сравнения ключей в словарях.

Для строки "hello":

Пример:

```
input_data = "hello"
print (f"Хеш для '{input_data}': {hash(input_data)}")
```

Вывод:

Хеш для 'hello': 4474357321870335833

Анализ:

- Встроенная функция hash () возвращает большое число, которое уникально для каждого объекта (в пределах текущего запуска программы).
- В отличие от нашей простой хеш-функции, встроенная функция hash() более сложная и лучше справляется с коллизиями.

Задание 3: Использование библиотеки hashlib Описание задачи:

Библиотека hashlib предоставляет криптографические хеш-функции, такие как MD5, SHA- 1, SHA-256 и другие. Эти функции используются для создания уникальных хешей, которые трудно подделать.

Пример:

1. **MD5**:
    - MD5 — это популярная хеш-функция, которая возвращает 128-битный хеш (32 символа в шестнадцатеричном формате).
    - Пример для строки "hello":

MD5 хеш для 'hello': 5d41402abc4b2a76b9719d911017c592

```
def md5_hash(input_string):
    return hashlib.md5(input_string.encode()).hexdigest()
```

Вывод:SHA-256:

- SHA-256 — это более безопасная хеш-функция, которая возвращает 256- битный хеш (64 символа в шестнадцатеричном формате).
- Пример для строки "hello":

```
def sha256_hash(input_string):
    return hashlib.sha256(input_string.encode()).hexdigest()


print(f"SHA-256 хеш для '{input_data}': {sha256_hash(input_data)}")
```

Вывод:

```
SHA-256 хеш для 'hello': 2cf24dba5fb0a30e26e83b2ac5b9e29e1b161e5c1fa7425e73043362938b9824
```

Анализ:

- MD5 и SHA-256 возвращают хеши фиксированной длины, которые уникальны для каждой входной строки.
- SHA-256 считается более безопасным, чем MD5, так как он менее подвержен коллизиям.

Задание 4: Анализ коллизий

ASCII-коды всех символов в строке. Давайте разберем, как это работает:

```
def simple_hash(input_string):
    """Простая хеш-функция для строк."""
    hash_value = 0
    for char in input_string:
        hash_value += ord(char)
    return hash_value
```

- hash_value инициализируется нулем.
- Для каждого символа в строке input_string функция ord(char) возвращает ASCII-код символа, который затем добавляется к hash_value.
- В конце функция возвращает итоговое значение

hash_value. Теперь рассмотрим пример использования:

python

```
string1 = "abc"
string2 = "cba"
                print(f"Хеш для '{string1}': {simple_hash(string1)}")
                print(f"Хеш для '{string2}': {simple_hash(string2)}")
```

Для строки "abc":

- ord('a') = 97
- ord('b') = 98
- ord('c') = 99

- Сумма: 97 + 98 + 99 = 294

Для строки "cba":

- ord('c') = 99
- ord('b') = 98
- ord('a') = 97
- Сумма: 99 + 98 + 97 = 294

Таким образом, вывод будет:

```
Хеш для 'abc': 294

Хеш для 'cba': 294
```

Замечания:

1. **Коллизии**: Эта хеш-функция может легко приводить к коллизиям, так как разные строки могут иметь одинаковую сумму ASCII-кодов. Например, строки "abc" и "cba" дают одинаковый хеш.
2. **Простота**: Эта функция очень проста и не подходит для серьезных применений, таких как криптография или хеш-таблицы, где требуется минимизировать коллизии и обеспечить равномерное распределение хеш-значений.

Если вам нужна более надежная хеш-функция, рассмотрите использование встроенных функций, таких как hash() в Python или библиотек, таких как hashlib

Задание 5: Практическое применение

хеширования Описание задачи:

Хеширование часто используется для проверки целостности данных. В этом задании вы реализуете функцию, которая проверяет, были ли данные изменены, сравнивая их хеш с оригинальным хешем. В Python для работы с хеш-функциями, такими как MD5, можно использовать модуль hashlib. Давайте исправим ваш код, добавив реализацию функции md5_hash и доработав его.

```
import hashlib
def md5_hash(data):
    """Вычисляет MD5-хеш для строки."""
    # Преобразуем строку в байты, так как hashlib работает с байтами
    byte_data = data.encode('utf-8')
    # Создаем MD5-хеш
    hash_object = hashlib.md5(byte_data)
    # Возвращаем хеш в виде шестнадцатеричной строки
    return hash_object.hexdigest()
def check_integrity(data, original_hash, hash_function):
    """Проверяет целостность данных, сравнивая их хеш с оригинальным хешем."""
    return hash_function(data) == original_hash
# Пример использования
original_data = "hello"
original_md5_hash = md5_hash(original_data)
# Проверка для неизмененных данных
print(check_integrity("hello", original_md5_hash, md5_hash)) # True
# Проверка для измененных данных
print(check_integrity("hell0", original_md5_hash, md5_hash))  # False
```

Объяснение:

Функция md5_hash:

Принимает строку data.

Преобразует строку в байты с помощью encode('utf-8'), так как модуль hashlib работает с байтами.

Вычисляет MD5-хеш с помощью hashlib.md5().

Возвращает хеш в виде шестнадцатеричной строки с помощью метода hexdigest().

Функция check_integrity:

Принимает данные (data), оригинальный хеш (original_hash) и функцию хеширования (hash_function).

Вычисляет хеш для данных с помощью hash_function и сравнивает его с original_hash.

Возвращает True, если хеши совпадают (данные не изменены), и False, если хеши различаются (данные изменены).

Пример использования:

Сначала вычисляется оригинальный хеш для строки "hello".

Затем проверяется целостность данных для строк "hello" (неизмененные данные) и "hell0" (измененные данные).

Вывод:

```
True
False
```

Анализ:

- Если данные не изменены, их хеш совпадает с оригинальным хешем.
- Если данные изменены, хеш будет другим, и функция вернет False.

Итог:

- В задании 1 вы реализовали простую хеш-функцию и увидели её ограничения.
- В задании 2 вы познакомились с встроенной функцией hash().
- В задании 3 вы изучили криптографические хеш-функции (MD5 и SHA-256).
- В задании 4 вы проанализировали коллизии.
- В задании 5 вы применили хеширование для проверки целостности данных.

ПРАКТИЧЕСКАЯ РАБОТА № 6.
АУТЕНТИФИКАЦИЯ ПОЛЬЗОВАТЕЛЕЙ WEB-СИСТЕМ СРЕДСТВАМИ ТЕХНОЛОГИИ PHP

1. *Цель работы:*

изучение принципов аутентификации пользователей в Web-системах на примере PHP-сеансов.

2. Основные теоретические сведения

2.1. Принципы аутентификации

Аутентификация — это попытка доказать, что кто-то на самом деле является тем, за кого себя выдает. Аутентификация обычно используется для разрешения или запрещения доступа к определенной информации. Существует множество методов обеспечения аутентификации, но, как и в случае с общей защитой, чем больше метод защищен, тем сложнее его использовать.

Методы аутентификации включают пароли, цифровые подписи, биометрические показатели, такие как отпечатки пальцев, а также методы с использованием специального оборудования, например, интеллектуальные карточки (смарт-карты). В Web повсеместно используются только два метода — пароли и цифровые сертификаты.

Пароли легко реализовать, просто использовать, и они не требуют наличия специального оборудования. Пароли обеспечивают некоторый уровень аутентификации, но не подходят в качестве единственного метода аутентификации для систем с сильной системой защиты.

Концепция паролей достаточно простая. Ваш пароль знают только вы и система. Если посетитель утверждает, что он — это вы и знает ваш пароль, то система верит, что посетитель — это вы. До тех пор пока никто не знает пароль, система остается защищенной. Сами по себе пароли обладают некоторыми потенциальными недостатками и не могут использоваться для надежной аутентификации.

Пароли могут перехватываться электронным путем. Используя программы перехвата клавиатурного ввода на терминалах или анализаторы сетевых протоколов (sniffers), взломщики могут перехватывать интересные пары имя пользователя-пароль. Шифрование сетевого потока данных позволяет снизить риск перехвата паролей.

Механизмы аутентификации встроены в наиболее популярные Web-браузеры и серверы. Аутентификация пользователей - это достаточно распространенная задача и существуют возможности аутентификации, встроенные в HTTP-протокол (Hypertext Transfer Protocol - протокол передачи гипертекстов). Сценарии и Web-серверы могут запрашивать аутентификацию у Web-браузера. После этого Web-браузер должен вывести на экран диалоговое окно или что-то подобное и запросить у пользователя необходимую информацию. Хотя Web-сервер запрашивает новые детали аутентификации в каждом запросе пользователя, Web-браузеру нет необходимости запрашивать эту информацию для каждой страницы. В общем случае браузер хранит детали аутентификации, пока открыто окно браузера, и автоматически отправляет их без вмешательства со стороны пользователя.

Описанная возможность HTTP-протокола называется базовой аутентификацией. Базовую аутентификацию можно включить средствами Web-сервера.

2.2. Выполнение аутентификации пользователей средствами управления сеансом в PHP

PHP (PHP Hypertext Preprocessor) - это серверный язык создания сценариев, разработанный специально для Web. В HTML-страницу можно внедрить код PHP, который будет выполняться при каждом ее посещении. Код PHP интерпретируется Web-сервером и генерирует HTML или иной вывод, наблюдаемый посетителем страницы. Синтаксис PHP основывается на других языках программирования, в первую очередь на C и Perl.

HTTP-протокол иногда называют «протоколом без состояния». Это означает, что данный протокол не имеет встроенного способа поддержки состояния между двумя транзакциями. Когда пользователь запрашивает друг за другом две страницы, HTTP не обеспечивает возможности уведомить, что оба запроса исходят от одного и того же пользователя. Это усложняет перенос между страницами введенных пользователем данных, таких как данные аутентификации.

Таким образом, идея управления сеансами заключается в обеспечении отслеживания пользователя в течение одного сеанса связи с Web-сайтом.

Реализация управления простым сеансом

Основными этапами использования сеанса являются следующие:

Запуск сеанса

Регистрация переменных сеанса

Использование переменных сеанса

Отмена регистрации переменных и закрытие сеанса

Заметим, что все перечисленные этапы не обязательно могут содержаться в одном сценарии, и некоторые из них могут находиться в нескольких сценариях. Рассмотрим каждый из этих этапов последовательно.

2.2.1. Запуск сеанса

Прежде чем можно будет воспользоваться функциональными возможностями сеанса, следует запустить сам сеанс. Существует несколько способов сделать это.

Но самый простой заключается в том, что сценарий начинается с вызова функции:

```
session_start () ;
```

Эта функция проверяет, существует ли идентификатор текущего сеанса. Если нет, она его создает. Если же идентификатор текущего сеанса уже существует, она загружает зарегистрированные переменные сеанса.

2.2.2. Регистрация переменных сеанса

Для того чтобы получить возможность отслеживать переменные от одного сценария к другому, их необходимо зарегистрировать. Это делается путем вызова функции session_register (). Например, для регистрации переменной $myvar применяется следующий код:

```
$myvar=5   ; session register("myvar")  ;
```

Данный оператор регистрирует имя переменной и отслеживает ее значение. Отслеживание переменной будет осуществляться, пока не завершится сеанс, либо пока вручную не отменится ее регистрация.

За один прием можно зарегистрировать более одной переменной, передав разделенный запятыми список имен переменных:

Session register ("myvar1","myvar2");

2.2.3. Использование переменных сеанса

Чтобы сделать переменную сеанса доступной для использования, сначала необходимо запустить сеанс.

После этого появляется доступ к этой переменной. Если включена опция register_globals, то доступ к этой переменной можно получить через сокращенную форму ее имени, например, $myvar. Если же упомянутая опция не включена, получить доступ к переменной можно через ассоциативный массив $HTTP_SESSION_VARS, например ,$HTTP_SESSIQN_VARS["myvar”].

Проверить, является ли переменная зарегистрированной переменной сеанса, можно обратившись к функции session_is_registered(). Вызов функции выполняется следующим образом:

$result=session_is_registered("myvar");

Эта функция проверит, является ли $myvar зарегистрированной переменной сеанса, и вернет true или false.

Можно поступить и по-другому — проверить массив

$HTTP SESSION_VARS на предмет наличия в нем переменной.

2.2.4. Отмена регистрации переменных и завершение сеанса

После окончания работы с переменной сеанса ее регистрацию можно отменить, воспользовавшись функцией session unregister():

session unregister ("myvar") ;

Подобно функции регистрации, эта функция требует указания имени переменной, регистрацию которой необходимо отменить, в виде строки, не включающей символ $. Данная функция за один раз может отменить регистрацию только одной переменной сеанса. Однако, для отмены регистрации всех переменных текущего сеанса можно обратиться к session_unset().

По завершении сеанса сначала потребуется отменить регистрацию всех переменных, а затем вызвать

session_destroy () ;

для обнуления идентификатора сеанса.

Пример сеанса

Вообще говоря, наилучшие функциональные возможности механизма управления сеансами обеспечиваются за счет аутентификации при помощи базы данных MySQL, но в данной работе мы этот способ рассматривать не будем.

Наш пример включает три простых сценария. Первый, index.php, обеспечивает форму для входной регистрации и аутентификации пользователей Web-сайта. Второй, secret.php, представляет информацию только для тех пользователей, которые успешно прошли входную регистрацию. Третий, logout.php, реализует выход пользователя из системы.

Данная страница предоставляет пользователю возможность войти в систему.

Посмотрим на код приложения. Большая часть кода сосредоточена в сценарии index.php. Давайте изучим его более подробно.

```
<?php
session_start ();
if (isset ($userid, $password))
if ($userid && $password)
{
if ($userid=="test" && $password=="test")
{
$valid_user = $userid; session_register("valid_user") ;
        }
    }
     ?>
```

```
<html>
<body>
<?php
if (session_is_registered ("valid_user"))
{
echo "Вы зарегистрированы в системе как" . $valid_user . "<br>";
echo "<a href=\"logout .php\">Log out</a><br>";
}
else
{
if (isset ($userid))
{
// если пользователь пытался зарегистрироваться,
//но возникла ошибка
echo "Невозможно войти";
} else
{
// если пользователь либо не пытался зарегистрироваться,
// либо покинул сайт
echo "Вы не зарегистрированы в системе. <br>";
      }
//provide form to log in
echo "<form method=post action=\"index.php\">";
echo "<table>";
echo "<tr><td>Логин:</td>";
echo "<td><input type=text name=userid></td></tr>";                    echo  “<tr><td>Пароль:</td>”;
echo  "<td><input type=password name=password></td></tr>";
echo  "<tr><td colspan=2 align=center>";
echo  "<input type=submit value=\"Войти\"></td></tr>";
echo  "</table></form>";
        }

?>
<br>
<a href="secret .php">Секретная страница</a>
</body>
</html>
```

Данный сценарий отличается сложной логикой, но иначе нельзя: ведь он осуществляет представление формы для входной регистрации и ее обработку.

Работа этого сценария сосредоточена вокруг переменной сеанса $valid_user. Основная идея здесь заключается в следующем: если кто-либо успешно прошел процедуру входной регистрации, мы регистрируем переменную сеанса с именем $valid_user, которая содержит идентификатор пользователя.

Первым делом в сценарии выполняется вызов session start(). Эта функция загружает переменную сеанса $valid_user, если последняя была зарегистрирована.

При первом проходе по сценарию ни один из условных операторов if не сработает и неудачливому пользователю к концу сценария останется лишь внимательно прочесть сообщение о том, что он не прошел процедуру входной регистрации. После этого мы предоставляем ему форму, при помощи которой он сможет это сделать:

```
echo "<form method=post action=\" index. php\">";
echo "<table>";
echo   "<tr><td>Логин:</td>";
```

```
echo "<td><input type=text name=userid></td></tr>";
echo "<tr><td>Пароль:</td>";
echo "<td><input type=password name=password></td></tr>";
echo "<tr><td colspan=2 align=center>";
echo "<input type=submit value=\"Войти\"></td></tr>";
echo "</table></form>";
```

Когда пользователь нажмет кнопку отправки (Войти), сценарий вызывается заново и вновь все начинается с начала. На этот раз в нашем распоряжении будут имя пользователя и пароль, позволяющие его аутентифицировать (они хранятся в Suserid и Spassword). Если эти переменные установлены, переходим к блоку аутентификации:

```
if   ($userid &&  $password)
{
if   ($userid=="test"    && $password=="test")
{
$valid_user = $userid; session_register ("valid_user") ;
}
```

Если выполнилось соответствие заданным логину и паролю, соответствие этой паре, мы регистрируем переменную $valid_user, которая содержит идентификатор для конкретного пользователя. Таким образом, мы знаем, кто вошел в систему и, соответственно, будем его отслеживать.

Поскольку уже известно, кто сейчас посещает сайт, то повторно предоставлять ему форму входной регистрации нет необходимости. Вместо этого мы сообщаем пользователю, что мы знаем, кто он такой, и даем ему возможность выхода из системы:

```
if (session_is_registered("valid_user"))
{
echo "Вы зарегистрированы в системе как . $valid_user . "<br>";
echo "<a href=\"logout .php\">Log out</a> <br>";
```

Если же при попытке произвести входную регистрацию пользователя мы по какой-то причине терпим неудачу, то у нас имеется идентификатор пользователя, но нет переменной $valid_user, и ничего не остается, кроме как выдать сообщение об ошибке:

```
if   (isset($userid)}
{
// если пользователь пытался зарегистрироваться, но
возникла ошибка
echo"Невозможно войти"; }
```

Поскольку $valid_user является зарегистрированной переменной сеанса, ее нельзя перезаписать путем передачи другого значения через URL, например так:

```
members        only  .  php?valid       user=testuser
```

С основным сценарием, похоже, все понятно. А теперь посмотрим на долгожданную секретную страницу.

```
<?php
session_start();
echo "<h2> Эта страница только для зарегистрированных пользователей</h2>";
// проверить переменные сеанса
if (session_is_registered("valid_user"))
{ echo "<p> Вы зарегистрированы в системе как: $valid_user.</p>";
echo "<p><h3>Это секретная информация. Она доступна только зарегистрированным пользователям. </h3></p>";
}
else
{ echo "<p>Вы не зарегистрированы в системе.</p>";
echo "<p>Страница доступна только зарегистрированным пользователям.</p>";
```

```
}
echo "<a href=\"index.php\">На главную страницу</a>";
?>
</body> </html>
```

Приведенный выше код очень прост. Все, что он делает — это запуск сеанса и проверка того, содержит ли текущий сеанс зарегистрированного пользователя, с использованием функции session_registered_user(). Если пользователь прошел процедуру входной регистрации, мы отображаем содержимое сайта для зарегистрированных пользователей, в противном случае мы сообщаем ему, что у него нет соответствующих полномочий.

И в завершение рассмотрим сценарий logout.php, который завершает регистрацию пользователя в системе.

```
<?
session_start ();
$old_user = $valid_user;
// сохранить для проверки регистрировался ли пользователь
$result = session_unregister ("valid_user"); session_destroy();
?>
<html> <body>
<ЫЗ>Выход из системы</h1>
<?
if (!empty ($old_user))
{ if ($result)
{ //если пользователь был зарегистрирован, не покинул систему echo "Вы вышли из системы. <br>";
}
else
{ // если пользователь был зарегистрирован
// и не может покинуть систему
echo "Невозможно выйти. <br>";
}
}
else
{ // если пользователь не был зарегистрирован,
// но как-то попал на эту страницу
echo "Вы не входили в систему, поэтому не можете выйти. <br>";
}
?>
<a href="index.php">На главную страницу</a>
</body>
</html>
```

Мы запускаем сеанс, запоминаем старое имя пользователя, отменяем регистрацию переменной $valid_user и завершаем сеанс. После этого мы выдаем пользователю одно из следующих сообщений: он вышел из системы, не может выйти из системы, поскольку первоначально даже не зарегистрировался.

Приведенный выше набор простых сценариев служит основой для многих проектов.

3. Порядок выполнения работы

1. Изучить принципы аутентификации пользователей в Web- системах.
2. Реализовать систему аутентификации с помощью PHP - сеансов.

4. Содержание отчета

1) титульный лист;
2) формулировку цели работы;
3) описание результатов выполнения;
4) выводы, согласованные с целью работы.

ПРАКТИЧЕСКАЯ РАБОТА № 7
КРИПТОГРАФИЧЕСКИЕ МЕТОДЫ ЗАЩИТЫ ИНФОРМАЦИИ

1. Цель работы

Ознакомиться с основными методами криптографической защиты информации. Получить практические навыки создания ПО по криптографическим преобразованиям информации.

2. Общие теоретические сведения

По мере развития и усложнения средств, методов и форм автоматизации процессов обработки информации повышается зависимость общества от степени безопасности используемых им информационных технологий, которая определяется степенью защищенности и устойчивости как компьютерных систем в целом, так и отдельных программ.

Для обеспечения защиты информации в настоящее время не существует какого-то одного технического приема или средства, однако общим в решении многих проблем безопасности является использование криптографии и криптоподобньгх преобразований информации.

Криптография - обеспечивает сокрытие смысла сообщения с помощью шифрования и открытия его расшифрованием, которые выполняются по специальным алгоритмам с помощью ключей.

Ключ - конкретное секретное состояние некоторых параметров алгоритма криптографического преобразования данных, обеспечивающее выбор только одного варианта из всех возможных для данного алгоритма

Криптоанализ - занимается вскрытием шифра без знания ключа (проверка устойчивости шифра).

Кодирование - (не относится к криптографии) -- система условных обозначений, применяемых при передаче информации. Применяется для увеличения качества передачи информации, сжатия информации и для уменьшения стоимости хранения и передачи.

Криптографические преобразования имеют цель обеспечить недоступность информации для лиц. не имеющих ключа, и поддержание с требуемой надежностью обнаружения несанкционированных искажений.

В криптографии используются следующие основные алгоритмы шифрования:

алгоритм замены (подстановки) - символы шифруемого текста заменяются символами того же или другого алфавита в соответствии с заранее обусловленной схемой замены;

алгоритм перестановки - символы шифруемого текста переставляются по определенному правилу в пределах некоторого блока этого текста;

гаммирование - символы шифруемого текста складываются с символами некоторой случайной последовательности;

аналитическое преобразование - преобразование шифруемого текста по некоторому аналитическому правилу (формуле).

Процессы шифрования и расшифрования осуществляются в рамках некоторой криптосистемы Для симметричной криптосистемы характерно применение одного и того же ключа как при шифровании, так и при расшифровании сообщений. В асимметричных криптосистемах для зашифрования данных используется один (общедоступный) ключ, а для расшифрования - другой (секретный) ключ.

2.1. Симметричные криптосистемы 2.1.1. Шифры перестановки

В шифрах средних веков часто использовались таблицы, с помощью которых выполнялись простые процедуры шифрования, основанные на перестановке букв в сообщении. Ключом в данном случае является размеры таблицы. Например, сообщение "Неясное становится еще более непонятным" записывается в таблицу из 5 строк и 7 столбцов по столбцам.

Н	О	Н	С	Б	Н	Я
Е	Е	О	Я	О	Е	Т
Я	С	В	Е	Л	П	Н
С	Т	И	Щ	Е	О	Ы
Н	А	Т	Е	Е	Н	М

Для получения шифрованного сообщения текст считывается по строкам и группируется по 5 букв:
НОНСБ НЯЕЕО ЯОЕТЯ СВЕЛП НСТИЩ ЕОЫНА ТЕЕНМ
Несколько большей стойкостью к раскрытию обладает метод одиночной перестановки по ключу. Он отличается от предыдущего тем, что столбцы таблицы переставляются по ключевому слову, фразе или набору чисел длиной в строку таблицы. Используя в качестве ключа слово ЛУНАТИК, получим следующую таблицу:

Л	У	Н	А	Т	И	К			А	И	К	Л	Н	Т	У
4	7	5	1	6	2	3			1	2	3	4	5	6	7
Н	О	Н	С	Б	Н	Я			С	Н	Я	Н	Н	Б	О
Е	Е	О	Я	О	Е	Т			Я	Е	Т	Е	О	О	Е
Я	С	В	Е	Л	П	Н			Е	П	Н	Я	В	Л	С
С	Т	И	Щ	Е	О	Ы			Щ	О	Ы	С	И	Е	
Н	А	Т	Е	Е	Н	М			Е	Н	М	Н	Т	Е	А

До перестановки После перестановки

В верхней строке левой таблицы записан ключ, а номера под буквами ключа определены в соответствии с естественным порядком соответствующих букв ключа в алфавите. Если в ключе встретились бы одинаковые буквы, они бы нумеровались слева направо Получается шифровка: СНЯНН БОЯЕТ ЕООЕЕ ПНЯВЛ СЩОЫС ИЕТЕН МНТЕА. Для обеспечения дополнительной скрытности можно повторно шифровать сообщение, которое уже было зашифровано. Для этого размер второй таблицы подбирают так, чтобы длины ее строк и столбцов отличались от длин строк и столбцов первой таблицы. Лучше всего, если они будут взаимно простыми.
Кроме алгоритмов одиночных перестановок применяются алгоритмы двойных перестановок. Сначала в таблицу записывается текст сообщения, а потом поочередно переставляются столбцы, а затем строки. При расшифровке порядок перестановок был обратный Пример данного метода шифрования показан в следующих таблицах:

16	3	2	13			О	И	Р	Т
5	10	11	8			З	Ш	Е	Ю
9	6	7	12			_	Ж	А	С
4	15	14	1			Е	Г	О	П

П Р И Е З Ж А Ю _ Ш Е С Т О Г О
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16Число магических квадратов очень резко возрастает с увеличением размера его сторон: для таблицы 3*3 таких квадратов -1; для таблицы 4*4 - 880; а для таблицы 5*5-250000.

2.1.2. Шифры простой замены

Система шифрования Цезаря - частный случай шифра простой замены. Метод основан на замене каждой буквы сообщения на другую букву того же алфавита, путем смещения от исходной буквы на К букв.
Известная фраза Юлия Цезаря VEN1 VINI VICI - пришел, увидел, победил, зашифрованная с помощью данного метода, преобразуется в SBKF SFAF SFZF (при смещении на 4 символа).
Греческим писателем Полибием за 100 лет до н.э. был изобретен так называемый полибианский квадрат размером 5*5, заполненный алфавитом в случайном порядке. Греческий алфавит имеет 24 буквы, а 25-м символом является пробел. Для шифрования на квадрате находили букву текста и записывали в шифротекст букву, расположенную ниже ее в том же столбце. Если буква оказывалась в нижней строке таблицы, то брали верхнюю букву из того же столбца.

2.1.3. Шифры сложной замены

Шифр Гронсфельда состоит в модификации шифра Цезаря числовым ключом. Для этого под буквами сообщения записывают цифры числового ключа. Если ключ короче сообщения, то его запись циклически повторяют. Шифротекст получают примерно также, как в шифре Цезаря, но отсчитывают не третью букву по алфавиту (как в шифре Цезаря), ц ту. которая смещена по алфавиту на соответствующую цифру ключа.

Пусть в качестве ключа используется группа из ipex цифр- 314, тогда Сообщение СОВЕРШЕННО СЕКРЕТНО Ключ 3143143143143143143 Шифровка ФПИСЬИОССАХИЛФИУСС

2.1.4. Шифр многоалфавитной замены

Для шифрования каждого символа исходного сообщения применяется свой шифр простой замены (свой алфавит).

	АБВГДЕЕЖЗИКЛМНОПРСТУФХЧШЩЪЫЬЭЮЯ_
А	АБВГДЕЕЖЗИКЛМНОПРСТУФХЧШЩЪЫЬЭЮЯ_
Б	_ АБВГДЕЕЖЗИКЛМНОПРСТУФХЧШЩЪЫЬЭЮЯ
В	Я_АБВГДЕЕЖЗИКЛМНОПРСТУФХЧШЩЪЫЬЭЮ
Г	ЮЯ_АБВГДЕЕЖЗИКЛМНОПРСТУФХЧШЩЪЫЬЭ
.	……..……. ……..……...
Я	АБВГДЕЕЖЗИКЛМНОПРСТУФХЧШЩЪЫЬЭЮЯ_АБ
_	БВГДЕЕЖЗИКЛМНОПРСТУФХЧШЩЪЫЬЭЮЯ_А

Каждая строка в этой таблице соответствует одному шифру замены аналогично шифру , Цезаря для алфавита, дополненного пробелом. При шифровании сообщения его выписывают в строку, а под ним ключ. Если ключ оказался короче сообщения, то его циклически повторяют. Шифротекст получают, находя символ в колонке таблицы по букве текста и строке, соответствующей букве ключа. Например, используя ключ АГАВА, из сообщения ПРИЕЗЖАЮ ШЕСТОГО получаем следующую шифровку:

Сообщение	ПРИЕЗЖАЮ_ШЕСТОГО
Ключ	АГАВААГАВААГАВАА
Шифровка	ПНИГЗЖЮЮЮАЕОТМГО

В компьютере такая операция соответствует сложению кодов ASCII символов сообщения и ключа но модулю 256.

Основные шаги шифровании текстового файла методом гаммирования.

1 . Получить от пользователя ключ, имя входного и выходного файла.

Инициализировать генератор случайных чисел с помощью ключа. Открыть указанные файлы.

Прочитать строку из файла.

Получить случайное число.

Получить ASCII-код очередного символа строки и увеличить его на случайное число, полученное на шаге 4.

Проверить правильность (допустимый диапазон) нового ASCII-кода.

В выходную строку записать очередной символ, соответствующий ASCII-коду, полученному на шаге 6.

Если не достигли конца входной строки, то перейти к шагу 4.

Записать полученную строку в выходной файл.

10. Если не достигнут конец файла, то перейти к шагу 3.И. Закрыть файлы.

Алгоритм дешифрации аналогичен ангоритму шифрации за исключением того, что из ASCII -кода вычитаем 256 и проверяем больше ноля или нет.

Open Filename For Input As # FileNumber-открытие файла для чтения.

В ASCII -коде символы 10 и 13 (возврат каретки).

Out Put -для вывода.

Binary- ключевое слово, открывает файлы как двоичные.

Line Input # FileNumber. AS -переменная строковая.

Print-для записи.

Variant- тип переменной для чтения и записи двоичного файла

Put#NF,,VA
Get#NF,,VA
Close-закрытие файла.

2.1.5. Гаммирование

Процесс зашифрования заключается в генерации гаммы шифра и наложении этой гаммы на исходный открытый текст. Перед шифрованием открытые данные разбиваются на блоки $T(0)^i$, одинаковой длины (по 64 бита). Гамма шифра вырабатывается в виде последовательности блоков $Г(ш)^i$, аналогичной длины ($T(ш)^i = Г(ш)^i + T(0)^i$, где + -побитовое сложение, i=1-m).

Процесс расшифрования сводится к повторной генерации шифра текста и наложение этой гаммы на зашифрованные данные $T(0)^i = Г(ш)^i + T(ш)^i$.

2.2. Асимметричные криптосистемы

2.2.1. Схема шифрования Эль Гамаля

Алгоритм шифрования Эль Гамаля основан на применении больших чисел для генерации открытого и закрытого ключа, криптостойкость же обусловлена сложностью вычисления дискретных логарифмов.

Последовательность действий пользователя:

Получатель сообщения выбирает два больших числа P *и* G, причем $P>G$.

Получатель выбирает секретный ключ - случайное целое число $X< P$.

Вычисляется открытый ключ $Y= G^x \bmod P$.

Получатель выбирает целое число K, $1< K< P-1$.

Шифрование сообщения *(M):* $a= G^K \bmod P$, $b=Y^K M \bmod P$, где пара чисел *(a,b)* является шифротекстом.

3. Порядок выполнения работы

На языке программирования написать программу шифрования и дешифрования текстового файла методом, указанным преподавателем.

4. Содержание отчета

Название работы.
Цель работы.
Тексты программ.
Общие выводы, сделанные в процессе выполнения лабораторной работы.

5. Контрольные вопросы:

5.1. Цель и задачи криптографии.
Симметричные криптосистемы: шифры перестановки.
Симметричные криптосистемы: шифры простой замены.
Симметричные криптосистемы: шифры сложной замены.
Симметричные криптосистемы: гаммирование.
Асимметричные криптосистемы, схема шифрования Эль Гамаля.

10. Глоссарий

Аутентификация Проверка принадлежности субъекту доступа предъявленного им идентификатора, подтверждение подлинности.

База данных Объективная форма представления и организации совокупности данных (статей, расчетов и так далее), систематизированных таким образом, чтобы эти данные могли быть найдены и обработаны с помощью электронной вычислительной.

Безопасность информации Состояние информации, информационных ресурсов и информационных систем, при котором с требуемой вероятностью обеспечивается защита информации (данных) от утечки, хищения, утраты, несанкционированного уничтожения, искажения, модификации (подделки), копирования, блокирования информации и т.п.

Данные Информация, представленная в виде, пригодном для обработки автоматическими средствами при возможном участии человек.

Достоверность Идентичность объекта защиты заявленному. **Доступ несанкционированный к информации** Доступ к информации, нарушающий правила разграничения доступа с использованием штатных средств, предоставляемых средствами вычислительной техники или автоматизированными системами.

Доступность Свойство субъекта и/или объекта доступа быть доступным и используемым по запросу со стороны уполномоченного логического.

Защита информации Деятельность, направленная на предотвращение утечки защищаемой информации, несанкционированных и непреднамеренных воздействий на защищаемую информацию.

Критическая (конфиденциальная, защищаемая) информация Это информация с соответствующими грифами секретности, информация для служебного пользования, информация, являющаяся собственностью организации.

Легальные пользователи Пользователи, имеющие законные основания для доступа к заданным ресурсам и сервисам.

Надежность сети Свойство сети сохранять во времени в установленных пределах значения всех параметров, характеризующих способность выполнять требуемые функции в заданных режимах и условиях использования и технического обслуживания.

Нарушитель (в автоматизированной системе) Субъект, имеющий доступ к работе со штатными средствами автоматизированной системы и средствами вычислительной техники как части автоматизированной системы

Несанкционированный доступ Нарушение регламентированного доступа к объекту защиты (Защита информации.

Обслуживающий персонал Сотрудники, не имеющие доступа к технологическому оборудованию СС, выполняющие функции по обслуживанию заданий, сооружений, технических систем и имеющих возможность физического доступа к оборудованию связи.