Федеральное государственное бюджетное образовательное учреждение
высшего образования
«КАЛИНИНГРАДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ»

В. В. Подтопельный

БЕЗОПАСНОСТЬ ОПЕРАЦИОННЫХ СИСТЕМ

Учебно-методическое пособие по изучению дисциплины
для студентов по специальности
10.05.03 «Информационная безопасность автоматизированных систем»
специализации «Безопасность открытых информационных систем»

Калининград
Издательство ФГБОУ ВО «КГТУ»
2026

УДК 004.056.5(075)

Рецензент
кандидат физико-математических наук, доцент, доцент кафедры информационной безопасности ФГБОУ ВО «Калининградский государственный технический университет» Н. Я. Великите

Подтопельный, В. В.
Безопасность операционных систем: учебно-методическое пособие по изучению дисциплины для студентов специальности 10.05.03 Информационная безопасность автоматизированных систем, специализация «Безопасность открытых информационных систем» / В. В. Подтопельный. – Калининград: Изд-во ФГБОУ ВО «КГТУ», 2026. – 91 с.

Учебно-методическое пособие является руководством по изучению дисциплины «Безопасность операционных систем». В учебно-методическом пособии приведен тематический план изучения дисциплины. Представлены методические указания по изучению дисциплины. Даны рекомендации по подготовке к промежуточной аттестации в форме зачёта и экзамена, по выполнению самостоятельной работы. Пособие подготовлено в соответствии с требованиями утвержденной рабочей программы модуля. Пособие предназначено для студентов 2, 3 курсов специальности 10.05.03 «Информационная безопасность автоматизированных систем».
Табл. 1, список лит. – 27 наименования

Учебно-методическое пособие по изучению дисциплины рекомендовано в качестве локального электронного методического материала для использования в учебном процессе методической комиссией ИЦТ 29 сентября 2026 г., протокол № 7

УДК 004.056.5(075)

© Федеральное государственное бюджетное образовательное учреждение высшего образования «Калининградский государственный технический университет», 2026 г.
© Подтопельный В. В., 2026 г.

ОГЛАВЛЕНИЕ

Введение 4
1. Тематический план 6
2. Методические рекомендации по изучению дисциплины 10
3. Методические рекомендации по подготовке к лабораторным занятиям 72
4. Методические указания по подготовке к самостоятельной работе 73
5. Методические указания по выполнению курсовой работы 76
6. Требования к аттестации по дисциплине 82
Заключение 86
Литература 87

ВВЕДЕНИЕ

Данное учебно-методическое пособие предназначено для студентов специальности 10.05.03 «Информационная безопасность автоматизированных систем», специализация: «Безопасность открытых информационных систем», изучающих дисциплину «**Безопасность операционных систем».**

Наименование компетенции:

ОПК-12. Способен применять знания в области безопасности вычислительных сетей, операционных систем и баз данных при разработке автоматизированных систем

Цель освоения дисциплины: формирование у студентов знаний о принципах построения операционных систем, защиты в операционных системах (ОС), навыки определения и внедрения комплекса мер (правил, процедур, практических приемов, руководящих принципов, методов, средств) для защиты операционных систем.

В результате освоения дисциплины обучающийся должен:

знать:

– способы реализации угроз безопасности в операционных системах;

– способы реализации угроз безопасности в автоматизированных системах;

уметь:

– формировать перечень мероприятий по предотвращению угроз безопасности операционных систем, информации в операционных системах;

владеть:

– навыками выявления уязвимости информационно-технологических ресурсов автоматизированных систем;

– навыками определения комплекса мер (правил, процедур, практических приемов, руководящих принципов, методов, средств) для защиты операционных систем.

Для успешного освоения дисциплины, в соответствии с учебным планом, ей предшествуют **«Основы информационной безопасности», «Технологии и методы программирования»**.

Далее в пособии представлен тематический план, содержащий перечень изучаемых тем, обязательных лабораторных работ, мероприятий текущей аттестации и отводимое на них аудиторное время (занятия в соответствии с расписанием) и самостоятельную работу. При формировании личного образовательного плана на семестр следует оценивать рекомендуемое время на

изучение дисциплины, возможно, вам потребует больше времени на выполнение отдельных заданий или проработку отдельных тем.

В разделе Содержание дисциплины приведены подробные сведения об изучаемых вопросах, по которым вы можете ориентироваться в случае пропуска каких-то занятий, а также методические рекомендации преподавателя для самостоятельной подготовки, каждая тема имеет ссылки на литературу (или иные информационные ресурсы), а также контрольные вопросы для самопроверки.

Раздел «Текущая аттестация» содержит описание обязательных мероприятий контроля самостоятельной работы и усвоения разделов или отдельных тем дисциплины. Далее изложены требования к завершающей аттестации – зачету и/или экзамену.

Помимо данного пособия, студентам следует использовать материалы, размещенные в соответствующем данной дисциплине разделу ЭИОС, в которые более оперативно вносятся изменения для адаптации дисциплины под конкретную группу.

Программное обеспечение:

1. Операционная система AstraLinux SE, ОС Linux.
2. Операционная система Windows 10 (получаемая по программе Microsoft "OpenValueSubscription").
3. Аппаратно-программный комплекс шифрования (допустимо использвание АПКШ «Континент» или др.)
4. DLP-система (допустимо использвание DLP SecureTower)
5. Системы защиты информации (допустимо использвание СЗИ Dallas Lock и др.)
6. Офисное приложение LibreOffice.
7. Google Chrome (GNU) 4. Oracle VM VirtualBox (GNU/Linux, macOS и Windows).
8. Офисное приложение MS Office Standard 2016. (получаемое по программе Microsoft "Open Value Subscription").
9. Oracle VirtualBox 7.1.6 и VirtualBox Extension Pack 7.1.6 for x86_64 hardware.
10. GoogleChrome (GNU).

1. ТЕМАТИЧЕСКИЙ ПЛАН

№ п/п	Раздел (модуль) дисциплины	Тема	Объем аудиторной работы, ч.	Объем самостоятельной работы, ч.
Лекции (4 семестр – 32 ч. ауд.)				
1.1	Современные операционные системы	Тема 1.1 Введение. Основы безопасности операционных систем	4	
1.2		Тема 1.2 Подсистемы безопасности в архитектуре операционной системы	4	10
1.3		Тема 1.3 Специфика информационной безопасности в управление процессами ОС	4	
1.4		Тема 1.4 Специфика информационной безопасности в управлении памятью ОС	4	10
1.5		Тема 1.5 Специфика информационной безопасности в управление прерываниями ОС	4	
1.6		Тема 1.6 Специфика информационной безопасности в управление вводом-выводом ОС	4	10
1.7		Тема 1.7 Особенности обеспечения безопасности файловых систем ОС	8	7,85
Лекции (5 семестр – 48 ч. ауд.)				
2.1	Подсистемы защиты операционной системы	Тема 2.1 Основные подсистемы защиты операционной системы	8	10
2.2		Тема 2.2 Управление доступом в операционных системах семейства Linux	8	10
2.3		Тема 2.3 Подсистема идентификации и аутентификации в ОС Linux	8	10

№ п/п	Раздел (модуль) дисциплины	Тема	Объем аудиторной работы, ч.	Объем самостоятельной работы, ч.
2.4		Тема 2.4 Обеспечение сетевой безопасности в ОС Linux	8	10
2.5		Тема 2.5 Обеспечение безопасности архитектуры, основных и дополнительных сервисов ОС Linux. Основные настройки сервисов безопасности ОС Linux	8	10
2.6	Подсистемы защиты операционной системы	Тема 2.6 Аудит системы безопасности ОС Linux	8	3
			80	**90,85**
		Лабораторные занятия (4 семестр)		
1.	Современные операционные системы	Лабораторная работа № 1. Работа с файлами и дисками в ОС Windows	4	-
2.		Лабораторная работа № 2. Организация пакетных файлов и сценариев в ОС Windows	4	-
3.		Лабораторная работа № 3. Организация консоли администрирования в ОС Windows	4	-
4.		Лабораторная работа № 4. Мониторинг, оптимизация и аудит ОС Windows	4	-
5.		Лабораторная работа № 5. Работа с Реестром ОС Windows	4	-
6.		Лабораторная работа № 6. Работа с подсистемой безопасности в ОС Windows	4	-
7.		Лабораторная работа № 7. Модель безопасности ОС Windows	4	-
8.		Лабораторная работа № 8. Создание и управление доменной	4	-

№ п/п	Раздел (модуль) дисциплины	Тема	Объем аудиторной работы, ч.	Объем самостоятельной работы, ч.
		политикой		
		Всего за семестр:	32	
Лабораторные занятия (5 семестр)				
1.	Подсистемы защиты операционной системы	Лабораторная работа № 9. Конфигурирование доменной политики	4	-
2.		Лабораторная работа № 10. Конфигурирование и использование EFS. Восстановление данных	4	-
3.	Подсистемы защиты операционной системы	Лабораторная работа № 11. ОС семейства UNIX. Работа с файлами и каталогами. Управление пользователями. Защита файлов. Резервное копирование данных	8	-
4.		Лабораторная работа № 12. Работа с процессами в операционной системе LINUX	8	-
5.		Лабораторная работа № 13. Особенности ОС Linux	8	-
6.		Лабораторная работа № 14. Механизмыбезопасности вLinux	32	-
		Всего за семестр:	**32**	
		Всего	**64**	
		Курсовая работа (проект)		
2.1	Название первого раздела	Контрольная точка 1. Раздел 1	-	-
3.1	Название третьего раздела	Контрольная точка 2. Раздел 2	-	-
		Оформление курсовой работы. Защита	-	-
			34,75	**0**

№ п/п	Раздел (модуль) дисциплины	Тема	Объем аудитор-ной работы, ч.	Объем самост-ной работы, ч.
		Всего:	РЭ – 14	
			КА– 4,4	
		Рубежный (текущий) и итоговый контроль		
2.1	Название второго раздела	Контроль 1 (не предусмотрен)	-	-
3.1	Название третьего раздела	Контроль 2 (не предусмотрен)	-	-
		Итоговый контроль (зачет)		
		Итоговый контроль (экзамен)		
			0	**0**
		Всего	**197,15**	**90,85**
			ИТОГО: 288	

2. МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ИЗУЧЕНИЮ ДИСЦИПЛИНЫ

Раздел 1. Современные операционные системы

Тема 1.1 Введение. Основы безопасности операционных систем

Перечень изучаемых вопросов:

1. Основные понятия: операционная система, безопасность ОС, системное программное обеспечение.
2. Классификация операционных систем.
3. Концептуальные основы операционных систем: концепция процесса, концепция ресурса, концепция виртуальности, концепция прерывания.
4. Ключевые аспекты безопасности ОС: конфиденциальность, целостность, доступность.
5. Угрозы безопасности ОС и современные механизмы защиты

Методические указания к изучению

Операционная система (ОС) представляет собой фундаментальный слой программного обеспечения, который управляет аппаратными ресурсами компьютера, обеспечивает выполнение приложений и взаимодействие пользователя с вычислительной системой. Безопасность ОС – это совокупность механизмов, направленных на защиту системы от несанкционированного доступа, вредоносных программ, а также на обеспечение стабильного и предсказуемого функционирования всех её подсистем. Безопасность ОС является основой для построения защищённых автоматизированных систем, поскольку любые прикладные механизмы защиты (антивирусы, межсетевые экраны, системы обнаружения вторжений) в конечном счёте опираются на доверие к ядру и подсистемам ОС. Если ядро скомпрометировано, то ни один прикладной механизм защиты не может гарантировать корректность своей работы.

Предварительно требуется определить понятие системного программного обеспечения. Под системным программным обеспечением понимается совокупность программ, предназначенных для управления аппаратными ресурсами, обеспечения работы прикладных программ и взаимодействия пользователя с вычислительной системой. К системному ПО относятся операционные системы, драйверы устройств, утилиты, оболочки, системы управления файлами, сетевые подсистемы и т. д. Именно ОС занимает центральное место в системном ПО, так как она распределяет ресурсы между всеми остальными программами и обеспечивает базовые механизмы изоляции и разграничения доступа.

Требуется рассмотреть классификации ОС по областям применения, по типам, по поддержке многозадачности, по поддержке многонитевости, по особенностям методов построения, по режимам обработки данных. По типу ядра выделяют монолитные ОС (Linux, Unix), где все компоненты работают в едином адресном пространстве, что обеспечивает высокую производительность, но усложняет защиту от ошибок; микроядерные ОС (QNX, MINIX), в которых критически важные функции вынесены в микроядро, а остальные работают в пользовательском режиме, что повышает отказоустойчивость; гибридные ОС (Windows NT, macOS), сочетающие элементы обоих подходов. По числу пользователей различают однопользовательские ОС (Windows 10 Home) и многопользовательские ОС (Linux, WindowsServer), поддерживающие одновременную работу нескольких пользователей с разграничением прав. По числу задач ОС делятся на однозадачные (MS-DOS) и многозадачные (все современные ОС), позволяющие параллельно исполнять несколько процессов. По назначению ОС могут быть универсальными (Windows, Linux) и специализированными (ОС для банкоматов, медицинского оборудования, IoT-устройств), оптимизированными под конкретные задачи.

Требуется рассмотреть особенности реализации концепции процесса (процесс, задача, программа, задание). Процесс – это экземпляр выполняемой программы, обладающий собственным адресным пространством, регистрами процессора и системными ресурсами. Перечень основных состояний процесса: создание, готовность, выполнение, блокировка, завершение. Обратить внимание на классификации процессов: системные и пользовательские, интерактивные и фоновые, приоритетные и непривилегированные. Безопасность процессов обеспечивается за счёт изоляции процессов (каждый процесс работает в своём виртуальном адресном пространстве), разграничения прав доступа (DAC – дискреционное управление доступом, MAC – мандатное управление доступом), контроля целостности (проверка цифровых подписей исполняемых файлов).

Требуется рассмотреть концепцию ресурса. Ресурсы ОС включают процессорное время, оперативную память, устройства ввода-вывода и файлы. Управление ресурсами предполагает планирование доступа (алгоритмы планирования CPU, такие как RoundRobin или приоритетное планирование), предотвращение конфликтов (например, deadlock – взаимоблокировка процессов), квотирование ресурсов (ограничение памяти или CPU для отдельных процессов). Классификация ресурсов включает физические и виртуальные, разделяемые и выделенные, локальные и удалённые, а также ресурсы, доступные только в монопольном режиме.

Требуется рассмотреть концепцию виртуальности. Виртуализация позволяет создавать абстракции физических ресурсов, таких как виртуальная память или виртуальные машины. Это повышает безопасность за счёт изоляции

сред (гостевые ОС в VMware или KVM не имеют прямого доступа к аппаратуре), эмуляции устройств (виртуальные сетевые адаптеры, диски), контейнеризации (Docker, LXC – изоляция процессов без полной виртуализации).

Требуется рассмотреть концепцию прерывания, действия ОС при выполнении прерывания, классы прерываний. Прерывания – это сигналы, требующие немедленной реакции процессора. Они делятся на аппаратные (например, нажатие клавиши на клавиатуре) и программные (системные вызовы, исключения). Безопасность обработки прерываний обеспечивается за счёт привилегированных режимов работы CPU (ring 0 – ядро, ring 3 – пользовательские процессы) и верификации источников прерываний (защита от подделки).

Ключевыми аспектами безопасности ОС являются:

a. **Конфиденциальность** – предотвращение несанкционированного доступа к данным.

b. **Целостность** – защита информации от несанкционированного изменения.

c. **Доступность** – обеспечение бесперебойной работы системы для авторизованных пользователей.

Угрозы безопасности ОС включают вредоносное ПО (вирусы, трояны, руткиты), эксплуатацию уязвимостей (например, переполнение буфера или racecondition), а также атаки на аутентификацию (перехват паролей, подбор учётных данных). Современные ОС используют комплексные механизмы защиты, такие как разграничение прав доступа, системы мониторинга и криптографические методы. В книге «Защита и безопасность в сетях Linux» подчёркивается, что безопасность – это не продукт, а процесс, и что нельзя достичь абсолютной защиты; задача администратора – сделать систему достаточно защищённой, чтобы злоумышленник предпочёл искать более лёгкую цель.

Для глубокого понимания основ безопасности ОС рекомендуется сочетать теоретическое изучение с практической деятельностью. Изучите базовые принципы работы ОС, механизмы управления процессами и ресурсами. Рекомендуемая Литература Таненбаум «Современные операционные системы», Вирт «Алгоритмы и структуры данных». Используйте эмуляторы (QEMU) и системы виртуализации (VirtualBox) для анализа работы ОС. Попробуйте настроить разграничение прав в Linux (chmod, SELinux).

При работе над курсовой работой обратите внимание на:

– описание уязвимостей, которые требуется приводить в первой (теоретической) главе.

Литература

1. Окороков, В. А. Безопасность операционных систем / В. А. Окороков. – Санкт-Петербург: Лань, 2024. – 228 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/367472 (дата обращения: 06.12.2024). – ISBN 978-5-507-48297-9. – Текст : электронный. (Глава 1).

2. Киренберг, Г. А. Информационная безопасность современных операционных систем: учеб. пособие / Г. А. Киренберг. – Кемерово: КузГТУ им. Т. Ф. Горбачева, 2022. – 138 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/295736 (дата обращения: 06.12.2024). – ISBN 978-5-00137-320-9. – Текст : электронный. (Глава 1).

3. Потерпеев, Г. Ю. Безопасность операционных систем: учеб. пособие / Г. Ю. Потерпеев, В. С. Нефедов, А. А. Криулин. – Москва: РТУ МИРЭА, 2021. – 93 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/182416 (дата обращения: 06.12.2024). – ISBN 978-5-7339-1393-3. – Текст : электронный. (Глава 1).

4. Зверева, О. М. Операционные системы: учеб. пособие / О. М. Зверева науч. ред. Л. Г. Доросинский; Уральский федеральный университет им. первого Президента России Б. Н. Ельцина. – Екатеринбург: Издательство Уральского университета, 2020. – 223 с. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=699030 (дата обращения: 03.11.2024). – ISBN 978-5-7996-3146-8. – Текст : электронный. (Глава 1).

Контрольные вопросы

1. Приведите классификацию операционных систем.
2. Приведите классификации процессов.
3. Дайте определение процесса.
4. Охарактеризуйте классификацию ресурсов.
5. Приведите порядок действий ОС при выполнении прерывания.

Тема 1.2 Подсистемы безопасности в архитектуре операционной системы

Перечень изучаемых вопросов

1. Совместимость и множественные прикладные среды.

2. Привилегированный режим работы ядра.
3. Многоуровневая организация операционных систем.
4. Аппаратная независимость и переносимость.
5. Микроядерная архитектура и безопасность.
6. Совместимость и множественные прикладные среды.

Методические указания к изучению

Современные операционные системы представляют собой сложные программные комплексы, архитектура которых определяет их функциональность, производительность и, что особенно важно, уровень безопасности. Центральным элементом любой ОС является ядро – основной компонент, обеспечивающий взаимодействие между аппаратным обеспечением и прикладными программами. Ядро операционной системы выполняет критически важные функции управления ресурсами, включая распределение процессорного времени, организацию работы с памятью и управление устройствами ввода-вывода.

Вспомогательные модули операционной системы расширяют базовую функциональность ядра, предоставляя дополнительные сервисы и интерфейсы для приложений. К таким модулям относятся драйверы устройств, подсистемы управления файлами, сетевые протоколы и другие компоненты. Важно отметить, что в современных ОС прослеживается чёткая тенденция к выносу максимального количества функциональности из пространства ядра в пользовательское пространство, что повышает стабильность и безопасность системы в целом.

Привилегированный режим работы ядра

Особенностью архитектуры современных процессоров является поддержка различных уровней привилегий выполнения кода. Ядро операционной системы работает в привилегированном режиме (часто называемом режимом ядра или Ring 0 в архитектуре x86), что предоставляет ему полный доступ ко всем ресурсам системы. Этот режим позволяет ядру выполнять привилегированные команды процессора, напрямую работать с аппаратурой и управлять виртуальной памятью.

Безопасность операционной системы во многом зависит от корректности работы кода, выполняемого в привилегированном режиме. Любая ошибка в этом коде может привести к серьёзным последствиям, включая крах всей системы или появление уязвимостей безопасности. Поэтому современные ОС реализуют различные механизмы защиты, такие как проверка целостности драйверов перед их загрузкой, использование аппаратных средств защиты памяти и строгий контроль доступа к системным ресурсам. Например, в Linux для изоляции драйверов используется фреймворкvfio, позволяющий запускать их внутри

виртуальных машин, а в Windows – User-ModeDriverFramework(UMDF), позволяющий выполнять драйверы в пользовательском пространстве.

Многоуровневая организация операционных систем

Современные операционные системы строятся по принципу многоуровневой организации, где каждый уровень предоставляет определённый набор абстракций и сервисов для вышележащих уровней. Такой подход позволяет добиться чёткого разделения функциональности и упрощает разработку и поддержку системы. На нижнем уровне находится аппаратное обеспечение, выше располагается уровень абстракции оборудования, затем ядро ОС, системные библиотеки и, наконец, прикладные программы.

Многослойная архитектура играет важную роль в обеспечении безопасности операционной системы. Каждый уровень может реализовывать собственные механизмы защиты, а чёткое разделение функциональности позволяет локализовать потенциальные уязвимости. Например, в системах с микроядерной архитектурой многие традиционно ядерные функции вынесены в пользовательское пространство, что значительно уменьшает возможную поверхность атаки.

Аппаратная независимость и переносимость

Одной из ключевых характеристик современных операционных систем является их способность работать на различном аппаратном обеспечении. Достигается это за счёт реализации слоя аппаратных абстракций (HAL – HardwareAbstractionLayer), который скрывает специфику конкретного оборудования за унифицированными интерфейсами. Такой подход позволяет переносить ОС на новые платформы с минимальными изменениями в основном коде системы.

Проблема переносимости тесно связана с вопросами безопасности. Различные аппаратные платформы могут предоставлять разные механизмы защиты, и операционная система должна эффективно использовать их все. Например, современные процессоры предлагают такие технологии безопасности, как NX-бит (запрет выполнения кода в определённых областях памяти) или аппаратная поддержка виртуализации, и ОС должна уметь работать с этими возможностями.

Микроядерная архитектура и безопасность

Альтернативой традиционной монолитной архитектуре операционных систем является микроядерный подход, при котором в привилегированном режиме выполняется минимальный набор функций, а остальные компоненты работают в пользовательском пространстве. Такая архитектура предлагает существенные преимущества с точки зрения безопасности, так как значительно уменьшает объём кода, выполняемого с максимальными привилегиями.

Микроядерные системы демонстрируют более высокую устойчивость к сбоям и атакам, поскольку ошибка в любом из компонентов пользовательского пространства не может напрямую повлиять на стабильность ядра. Однако такой подход требует тщательной проработки механизмов межпроцессного взаимодействия, которые могут стать потенциальной точкой уязвимости. Современные исследования в области безопасности операционных систем показывают, что микроядерная архитектура может обеспечить более высокий уровень защиты при правильной реализации.

Совместимость и множественные прикладные среды

Современные операционные системы сталкиваются с необходимостью поддержки приложений, разработанных для других платформ и сред. Решение этой задачи достигается различными способами: от полной эмуляции чужой среды до создания совместимых подсистем. Каждый из этих подходов имеет свои последствия для безопасности системы.

Например, подсистема Windows для Linux (WSL) реализует совместимость на уровне системных вызовов, тщательно контролируя взаимодействие между двумя средами. Аналогично, технологии виртуализации позволяют запускать целые операционные системы внутри защищённых сред, обеспечивая высокий уровень изоляции. Разработчики ОС должны тщательно проектировать такие механизмы, чтобы не создавать новых векторов атак при обеспечении совместимости.

Для глубокого понимания архитектуры операционных систем и связанных с ней аспектов безопасности рекомендуется сочетать теоретическое изучение с практической деятельностью. Особое внимание следует уделить анализу исходных кодов открытых операционных систем, таких как Linux или FreeBSD. Лабораторные задания могут включать исследование механизмов системных вызовов, анализ работы планировщика задач или изучение реализации подсистемы виртуальной памяти.

Полезным упражнением будет сравнение архитектурных решений в различных операционных системах. Например, можно проанализировать различия в реализации безопасности между монолитным ядром Linux и микроядерной архитектурой QNX. Также рекомендуется экспериментировать с настройкой параметров безопасности на разных уровнях операционной системы.

Предварительно требуется определить структурную организацию ОС на основе различных программных модулей. Рассмотреть типы ядер ОС.

Требуется рассмотреть**два режима работы ядра ОС:**

− пользовательский режим (usermode);

− привилегированный режим, который также называют режимом ядра (kernelmode), или режимом супервизора (supervisormode).

Требуется рассмотреть Многослойную структуру ОС (**средства аппаратной поддержки ОС, машинно-зависимые компоненты ОС, менеджеры ресурсов, интерфейс системных вызовов (API)**).

Требуется рассмотреть аппаратную зависимость и переносимость ОС.

Требуется рассмотреть**микроядерную архитектуру**, набор функций микроядра.

При работе над курсовой работойобратите внимание на:

– описание уязвимостей ОС, которые требуется приводить в первой (теоретической) главе.

Литература

1. Окороков, В. А. Безопасность операционных систем / В. А. Окороков. – Санкт-Петербург: Лань, 2024. – 228 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/367472 (дата обращения: 06.12.2024). – ISBN 978-5-507-48297-9. – Текст : электронный. (Глава 2).

2. Киренберг, Г. А. Информационная безопасность современных операционных систем: учеб. пособие / Киренберг, Г. А. – Кемерово: КузГТУ им. Т. Ф. Горбачева, 2022. – 138 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/295736 (дата обращения: 06.12.2024). – ISBN 978-5-00137-320-9. – Текст : электронный. (Глава 1).

3. Потерпеев, Г. Ю. Безопасность операционных систем: учеб. пособие / Г. Ю. Потерпеев, В. С. Нефедов, А. А. Криулин. – Москва: РТУ МИРЭА, 2021. – 93 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/182416 (дата обращения: 06.12.2024). – ISBN 978-5-7339-1393-3. – Текст : электронный. (Глава 2).

4. Зверева, О. М. Операционные системы: учеб. пособие / О. М. Зверева; науч. ред. Л. Г. Доросинский; Уральский федеральный университет им. первого Президента России Б. Н. Ельцина. – Екатеринбург: Издательство Уральского университета, 2020. – 223 с. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=699030 (дата обращения: 03.11.2024). – ISBN 978-5-7996-3146-8. – Текст : электронный. (Глава 10).

Контрольные вопросы:

1. Каковы основные функции ядра операционной системы и как они связаны с вопросами безопасности?

2. Почему привилегированный режим работы ядра является одновременно необходимым и потенциально опасным с точки зрения безопасности?

3. Как многоуровневая архитектура операционной системы способствует повышению её защищённости?

4. Какие механизмы используются в современных ОС для обеспечения аппаратной независимости?

5. В чём заключаются основные преимущества микроядерной архитектуры с точки зрения безопасности?

6. Каким образом операционные системы обеспечивают совместимость с приложениями, разработанными для других платформ?

7. Как реализована защита межпроцессного взаимодействия в микроядерных операционных системах?

8. Какие аппаратные технологии современных процессоров наиболее важны для обеспечения безопасности ОС?

9. Как влияет на безопасность системы вынос функциональности из ядра в пользовательское пространство?

10. Какие методы изоляции используются при реализации множественных прикладных сред в одной операционной системе?

Тема 1.3 Специфика информационной безопасности в управление процессами ОС

Перечень изучаемых вопросов

1. Понятие процесса и потока, их роль в безопасности.
2. Управление процессами и потоками. Модель изоляции процессов.
3. Алгоритмы планирования и их влияние на безопасность.
4. Синхронизация процессов и потоков, уязвимости состояния гонки.

Методические указания к изучению

Управление процессами – это одна из фундаментальных функций операционной системы, от которой напрямую зависит как производительность, так и безопасность вычислительной системы. Каждый процесс, будучи экземпляром выполняемой программы, обладает собственным виртуальным адресным пространством, что обеспечивает базовый уровень изоляции. Потоки, как более легковесные единицы выполнения, разделяют ресурсы процесса, что создаёт как преимущества для параллельного программирования, так и риски безопасности, такие как состояния гонки (raceconditions). Понимание этих механизмов необходимо для анализа угроз, связанных с несанкционированным доступом к памяти, эскалацией привилегий и отказами в обслуживании.

Предварительно требуется определить понятия «процесс» и «поток». Рассмотреть различия между ними с точки зрения ресурсов и безопасности. Изучить структуру блока управления процессом (PCB) и его роль в идентификации и аутентификации субъектов доступа.

Требуется рассмотреть механизмы управления процессами, включая их создание, планирование и завершение. Обратить внимание на системные вызовы, используемые для этих целей в ОС Linux (fork(),execve()) и Windows (CreateProcess()). Особое внимание следует уделить системе изоляции процессов, реализуемой через виртуальную память и аппаратные механизмы (MMU), которые предотвращают несанкционированный доступ одного процесса к памяти другого.

Требуется рассмотреть основные алгоритмы планирования, такие как First-Come, First-Served (FCFS), RoundRobin (RR) и приоритетное планирование. Проанализировать, как недостатки этих алгоритмов (например, «голодание» задач) могут быть использованы для атак типа «отказ в обслуживании» (DoS).

Требуется рассмотреть механизмы синхронизации процессов и потоков, такие как мьютексы, семафоры и условные переменные. Важно понять, что некорректное использование этих примитивов ведёт к уязвимостям. Например, состояние гонки (racecondition) может позволить злоумышленнику получить доступ к файлу с повышенными привилегиями, если проверка прав доступа и последующая операция над файлом выполняются неатомарно. В книге «Защита и безопасность в сетях Linux» приводится пример с командойmkdir, которая выполняется с правамиroot, но при этом создаёт каталог для обычного пользователя. Если между моментом создания каталога и моментом смены его владельца злоумышленник сможет удалить этот каталог и создать символическую ссылку на другой файл (например, /etc/shadow), то этот файл в итоге окажется принадлежащим злоумышленнику. Такие уязвимости демонстрируют, что безопасность управления процессами — это не только изоляция, но и тщательная проработка логики выполнения привилегированных операций.

Ключевыми аспектами безопасности управления процессами являются:

a. **Изоляция процессов** – обеспечение того, чтобы один процесс не мог напрямую читать или изменять память другого процесса. Реализуется через аппаратные механизмы виртуальной памяти.

b. **Разграничение прав доступа (DAC/MAC)** – определение того, какие ресурсы и операции доступны процессу на основе его идентификаторов (UID/GID) и назначенных политик безопасности.

c. **Контроль целостности** – проверка того, что исполняемый код процесса не был изменён злоумышленником. Это может достигаться с помощью

цифровых подписей и систем контроля целостности файлов (например, Tripwire).

Угрозы безопасности, связанные с процессами, включают атаки на переполнение буфера, внедрение кода (codeinjection), эскалацию привилегий через уязвимости в привилегированных процессах и атаки, направленные на исчерпание ресурсов (например, «fork-бомбы»). Современные ОС используют комплексные механизмы защиты, такие как рандомизация адресного пространства (ASLR), запрет выполнения кода в областях данных (DEP), а также системы контроля целостности исполняемых файлов.

При работе над курсовой работой обратите внимание на:

– описание процессов ОС, которые требуется приводить в первой (теоретической) главе.

Литература

1. Окороков, В. А. Безопасность операционных систем / В. А. Окороков. – Санкт-Петербург: Лань, 2024. – 228 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/367472 (дата обращения: 06.12.2024). – ISBN 978-5-507-48297-9. – Текст : электронный. (Главы 2–3)

2. Киренберг, Г. А. Информационная безопасность современных операционных систем: учеб. пособие / Киренберг, Г. А. – Кемерово: КузГТУ им. Т. Ф. Горбачева, 2022. – 138 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/295736 (дата обращения: 06.12.2024). – ISBN 978-5-00137-320-9. – Текст : электронный. (Глава 1).

3. Потерпеев, Г. Ю. Безопасность операционных систем: учеб. пособие / Г. Ю. Потерпеев, В. С. Нефедов, А. А. Криулин. – Москва: РТУ МИРЭА, 2021. – 93 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/182416 (дата обращения: 06.12.2024). – ISBN 978-5-7339-1393-3. – Текст : электронный. (Глава 2).

4. Зверева, О. М. Операционные системы: учеб. пособие / О. М. Зверева; науч. ред. Л. Г. Доросинский; Уральский федеральный университет им. первого Президента России Б. Н. Ельцина. – Екатеринбург: Издательство Уральского университета, 2020. – 223 с. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=699030 (дата обращения: 03.11.2024). – ISBN 978-5-7996-3146-8. – Текст : электронный. (Глава 3).

Контрольные вопросы:

1. Объясните различия между процессом и потоком с точки зрения ресурсов и безопасности.
2. Какие структуры данных использует ОС для управления процессами?
3. Как алгоритм RoundRobin предотвращает «голодание» задач?
4. Опишите сценарий взаимной блокировки (deadlock) и методы её разрешения.
5. Почему приоритетное планирование может снижать безопасность системы?
6. Какие аппаратные механизмы процессора участвуют в изоляции процессов?
7. Как операционная система обнаруживает попытки несанкционированного создания процессов?
8. Объясните роль мьютексов в предотвращении состояния гонки.

Тема 1.4 Специфика информационной безопасности в управлении памятью ОС

Перечень изучаемых вопросов:

1. Иерархия памяти и её влияние на безопасность.
2. Управление памятью. Типы адресации и защита.
3. Виртуальная память и свопинг: риски безопасности.
4. Алгоритмы управления памятью и их уязвимости.

Методические указания к изучению

Управление памятью является второй ключевой функцией ОС, критически важной для безопасности. Эффективное распределение, изоляция и защита памяти предотвращают утечки данных, атаки типа переполнения буфера и несанкционированный доступ к критическим ресурсам. Современные ОС используют сложные механизмы, включающие аппаратную поддержку, что требует глубокого понимания их архитектуры и потенциальных уязвимостей.

Предварительно требуется рассмотреть иерархию памяти, начиная от регистров процессора и кэш-памяти (L1, L2, L3) и заканчивая оперативной памятью (ОЗУ) и устройствами долговременного хранения (HDD, SSD). Важно понимать, что с точки зрения безопасности каждый уровень иерархии создаёт риски. Например, атаки Spectre и Meltdown эксплуатируют спекулятивное выполнение инструкций и кэш-память для чтения данных, к которым у процесса нет прав доступа.

Требуется рассмотреть функции ОС по управлению памятью в мультипрограммной системе. Это включает в себя:

- **Отслеживание свободной и занятой памяти**.
- **Выделение и освобождение памяти**по запросу процессов.
- **Изоляцию адресных пространств процессов**.

Требуется изучить типы адресации памяти:

- **Физическая адресация** – прямой доступ к ячейкам ОЗУ (используется ядром).
- **Виртуальная адресация** – абстракция, предоставляемая процессам для изоляции. Трансляция виртуальных адресов в физические выполняется с помощью блока управления памятью (MMU) и таблиц страниц. Именно на этом уровне реализуются права доступа (чтение, запись, выполнение) к страницам памяти. Ошибки в трансляции адресов (например, CVE-2018-18281 в ядре Linux) позволяют обходить защиту и получать доступ к памяти ядра.

Требуется рассмотреть концепцию виртуальной памяти и свопинга (выгрузки неактивных страниц на диск). Важно понимать, что этот механизм создаёт риски для конфиденциальности, так как данные в своп-файле могут быть восстановлены после перезагрузки системы. Для защиты применяется шифрование всего диска (LUKS) или отключение свопинга в системах с высокими требованиями к безопасности.

Требуется рассмотреть алгоритмы управления памятью, такие как FIFO, LRU (LeastRecentlyUsed) и оптимальный алгоритм. Проанализировать, как эти алгоритмы могут быть использованы для атак. Например, атаки на кэш могут искусственно увеличить количество промахов кэша (cachemisses), что приведёт к замедлению системы.

Ключевыми аспектами безопасности управления памятью являются:

a. **ASLR (AddressSpaceLayoutRandomization)** – рандомизация адресного пространства, которая делает непредсказуемым расположение ключевых областей памяти (стек, куча, библиотеки) и усложняет эксплуатацию уязвимостей переполнения буфера.

b. **DEP (DataExecutionPrevention) / NX-бит** – аппаратная технология, которая помечает определённые области памяти (например, стек и кучу) как неисполняемые, что предотвращает выполнение внедрённого кода.

c. **Защита от переполнения буфера** – использование «канареек» (stackcanaries) для обнаружения перезаписи стека и безопасных функций для работы со строками.

Угрозы безопасности, связанные с памятью, включают переполнение буфера, использование памяти после освобождения (use-after-free), утечки памяти и атаки через побочные каналы (side-channelattacks).

При работе над курсовой работой обратите внимание на:

– описание памяти ОС, которые требуется приводить в первой (теоретической) главе.

Литература

1. Окороков, В. А. Безопасность операционных систем / В. А. Окороков. – Санкт-Петербург: Лань, 2024. – 228 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/367472 (дата обращения: 06.12.2024). – ISBN 978-5-507-48297-9. – Текст : электронный. (Главы 3–5).

2. Киренберг, Г. А. Информационная безопасность современных операционных систем: учеб. пособие / Г. А. Киренберг. – Кемерово: КузГТУ им. Т. Ф. Горбачева, 2022. – 138 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/295736 (дата обращения: 06.12.2024). – ISBN 978-5-00137-320-9. – Текст : электронный. (Глава 1).

3. Потерпеев, Г. Ю. Безопасность операционных систем: учеб. пособие / Г. Ю. Потерпеев, В. С. Нефедов, А. А. Криулин. – Москва: РТУ МИРЭА, 2021. – 93 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/182416 (дата обращения: 06.12.2024). – ISBN 978-5-7339-1393-3. – Текст : электронный. (Глава 2, с. 31–39).

4. Зверева, О. М. Операционные системы: учеб. пособие / О. М. Зверева; науч. ред. Л. Г. Доросинский; Уральский федеральный университет им. первого Президента России Б. Н. Ельцина. – Екатеринбург: Издательство Уральского университета, 2020. – 223 с. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=699030 (дата обращения: 03.11.2024). – ISBN 978-5-7996-3146-8. – Текст : электронный. (Глава 5).

Контрольные вопросы

1. Приведите классификацию распределения памяти.
2. Приведите классификации алгоритмов управления памятью без использования механизма виртуальной памяти.
3. Дайте определение синхронизации.
4. Приведите классификацию алгоритмов управления памятью с использованием виртуальной памятью
5. Объясните роль MMU в обеспечении изоляции процессов.
6. Как ASLR предотвращает атаки переполнения буфера?
7. Какие риски безопасности связаны с использованием свопинга?
8. Как повреждение таблиц страниц может привести к уязвимостям?

9. Какие методы шифрования применяются для защиты данных в своп-файле?

Тема 1.5 Специфика информационной безопасности в управление прерываниями ОС

Перечень изучаемых вопросов

1. Понятие прерывания, классы прерываний.
2. Механизм обработки прерываний.
3. Роль централизованного диспетчера прерываний.
4. Системные вызовы как механизм взаимодействия с ядром.

Методические указания к изучению:

Прерывания представляют собой фундаментальный механизм взаимодействия между аппаратным обеспечением, программным обеспечением и операционной системой. Они позволяют процессору реагировать на внешние события (например, нажатие клавиши) или внутренние исключения (деление на ноль) без постоянного опроса устройств. Однако некорректная обработка прерываний может стать источником уязвимостей, таких как перехват управления системой или несанкционированный доступ к данным. Понимание архитектуры прерываний критически важно для обеспечения безопасности операционной системы.

Предварительно требуется определить понятие прерывания. Прерывание – это сигнал, инициируемый аппаратурой или программным обеспечением, который требует немедленного внимания процессора. Аппаратные прерывания генерируются устройствами (клавиатура, таймер, сетевой адаптер) для уведомления ОС о завершении операции или возникновении ошибки. Программные прерывания вызываются исполняемым кодом для запроса сервисов ядра (системные вызовы) или обработки исключений (ошибка доступа к памяти).

Требуется рассмотреть три больших класса прерываний:

1. **Программные прерывания** – инициируются выполняемым кодом для запроса сервисов ядра (системные вызовы) или обработки исключений (например, деление на ноль).

2. **Внешние (аппаратные) прерывания** – генерируются устройствами (клавиатура, таймер, сетевая карта) для уведомления ОС о завершении операции или возникновении ошибки.

3. **Внутренние (исключения)** – возникают внутри процессора при выполнении инструкций (например, ошибка доступа к памяти).

С точки зрения безопасности, прерывания создают «окна уязвимости», когда контекст выполнения переключается с пользовательского режима на режим

ядра. Злоумышленники могут эксплуатировать это, внедряя вредоносные обработчики прерываний или манипулируя векторами прерываний. Например, атака IOMMU CVE-2017-2636 в ядре Linux позволяла перехватывать прерывания от периферийных устройств.

Требуется рассмотреть два основных способа выполнения прерывания: опрашиваемый (polled) и векторный (vectored). В современных системах используется векторный метод, при котором каждому типу прерывания соответствует свой номер, и адрес обработчика хранится в таблице векторов прерываний (IDT — InterruptDescriptorTable). С точки зрения безопасности, IDT является критически важной структурой данных, и доступ к её изменению должен быть строго ограничен.

Требуется рассмотреть механизм обработки прерываний, который начинается с сохранения контекста текущего процесса (регистров, указателя инструкций) и переключения процессора в привилегированный режим. Обработчик прерывания выполняется в отдельном стеке ядра, чтобы избежать переполнения пользовательского стека. После завершения обработки восстанавливается контекст, и процесс продолжает работу.

Безопасность механизма прерываний обеспечивается:

1. **Защитой таблицы векторов** – только ядро ОС имеет право изменять IDT.
2. **Верификацией обработчиков** – цифровые подписи драйверов в Windows предотвращают внедрение неавторизованного кода.
3. **Изоляцией контекста** – обработчики выполняются в отдельном стеке ядра, чтобы избежать переполнения пользовательского стека.

Требуется рассмотреть роль централизованного диспетчера прерываний (IRQ Manager в Windows или irqchip в Linux), который отвечает за распределение прерываний между процессорами, управление приоритетами и балансировку нагрузки. Он предотвращает конфликты при одновременных запросах от устройств и обеспечивает своевременное выполнение критических задач (например, обработки сетевых пакетов). В многопроцессорных системах диспетчер использует алгоритмы вроде IRQ affinity, чтобы закрепить определённые прерывания за конкретными ядрами. Это снижает задержки, но создаёт риски неравномерной нагрузки и атак типа DoS через исчерпание ресурсов одного ядра. Для защиты применяется динамическое перераспределение прерываний и ограничение их частоты.

Требуется рассмотреть процедуры обработки прерываний, вызванные из текущего процесса. Когда прерывание возникает во время выполнения процесса, процессор сохраняет его состояние (регистры, флаги) и передаёт управление обработчику. После завершения обработки восстанавливается контекст, и процесс продолжает работу. Однако если прерывание требует длительной обработки

(например, загрузка данных с диска), ОС может перепланировать задачи, переключившись на другой процесс. Опасность возникает, когда обработчик прерывания обращается к ресурсам, используемым текущим процессом. Например, атака на двойное извлечение (doublefetch) эксплуатирует гонку между проверкой прав доступа в ядре и фактическим использованием данных. Для предотвращения таких уязвимостей используются атомарные операции и блокировки.

Требуется рассмотреть системные вызовы как особый вид программных прерываний, через которые приложения запрашивают сервисы ядра (открытие файла, создание процесса). В архитектуре x86 для этого используется инструкцияint 0x80илиsyscall, передающая управление заранее определённому обработчику в ядре. Безопасность системных вызовов обеспечивается:

1. **Проверкой параметров** – ядро валидирует указатели и размеры буферов, переданные из пользовательского пространства.
2. **Изоляцией адресных пространств** – данные процесса не доступны ядру напрямую, копирование выполняется через специальные функции (copy_from_user).
3. **Аудитом вызовов** – системы вроде SELinux или AppArmor отслеживают подозрительные операции.

Уязвимости в системных вызовах (например, CVE-2021-4034 вpkexec) позволяют повысить привилегии или выполнить произвольный код.

При работе над курсовой работой обратите внимание на:

– описание прерываний ОС, которые требуется приводить в первой (теоретической) главе.

Литература

1. Окороков, В. А. Безопасность операционных систем / В. А. Окороков. – Санкт-Петербург: Лань, 2024. – 228 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/367472 (дата обращения: 06.12.2024). – ISBN 978-5-507-48297-9. – Текст : электронный. (Главы 4–6).

2. Киренберг, Г. А. Информационная безопасность современных операционных систем: учеб пособие / Г. А. Киренберг. – Кемерово: КузГТУ им. Т. Ф. Горбачева, 2022. – 138 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/295736 (дата обращения: 06.12.2024). – ISBN 978-5-00137-320-9. – Текст : электронный. (Глава 1).

3. Потерпеев, Г. Ю. Безопасность операционных систем: учеб. пособие / Г. Ю. Потерпеев, В. С. Нефедов, А. А. Криулин. – Москва: РТУ МИРЭА, 2021. – 93 с. – Режим доступа: для авториз. пользователей. — Лань : электронно-

библиотечная система. – URL: https://e.lanbook.com/book/182416 (дата обращения: 06.12.2024). – ISBN 978-5-7339-1393-3. – Текст : электронный. (Глава 2).

4. Зверева, О. М. Операционные системы: учеб. пособие / О. М. Зверева; науч. ред. Л. Г. Доросинский; Уральский федеральный университет им. первого Президента России Б. Н. Ельцина. – Екатеринбург: Издательство Уральского университета, 2020. – 223 с. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=699030 (дата обращения: 03.11.2024). – ISBN 978-5-7996-3146-8. – Текст : электронный. (Глава 6)

Контрольные вопросы

1. Какие типы прерываний существуют и как они влияют на безопасность ОС?
2. Объясните роль таблицы векторов прерываний (IDT) в архитектуре x86.
3. Как централизованный диспетчер прерываний предотвращает конфликты между устройствами?
4. Почему сохранение контекста процесса критически важно при обработке прерываний?
5. Какие уязвимости могут возникнуть при некорректной обработке системных вызовов?
6. Как инструкциясyscallотличается отint 0x80c точки зрения производительности и безопасности?
7. Какие механизмы защищают обработчики прерываний от подмены злоумышленником?
8. Почему атаки типа "двойное извлечение" эффективны против ядра ОС?
9. Как SMAP предотвращает доступ ядра к пользовательской памяти?
10. Какие методы аудита используются для отслеживания подозрительных системных вызовов?

Тема 1.6 Специфика информационной безопасности в управление вводом-выводом ОС

Перечень изучаемых вопросов

1. Организация взаимодействия ОС с устройствами ввода-вывода.
2. Многослойная модель подсистемы ввода-вывода.
3. Менеджеры ввода-вывода и их роль в безопасности.
4. Драйверы устройств: безопасность на уровне ядра.

Методические указания к изучению

Управление вводом-выводом (I/O) представляет собой один из ключевых механизмов операционных систем, обеспечивающий взаимодействие между программным обеспечением и аппаратными устройствами. Этот процесс охватывает не только передачу данных, но и управление ресурсами, синхронизацию операций и защиту от несанкционированного доступа. В условиях роста сложности кибератак, направленных на периферийные устройства и драйверы, безопасность подсистемы ввода-вывода становится критической для обеспечения целостности и конфиденциальности данных. Уязвимости в этой области, такие как некорректная обработка прерываний или недостаточная изоляция драйверов, могут быть использованы для внедрения вредоносного кода, перехвата информации или нарушения работы системы.

Предварительно требуется рассмотреть организацию взаимодействия ОС с устройствами ввода-вывода. Взаимодействие начинается на этапе инициализации оборудования. Во время загрузки ОС ядро идентифицирует подключённые устройства через шинные интерфейсы (PCI, USB, SATA) и загружает соответствующие драйверы. Современные ОС, такие как Windows и Linux, используют технологию Plug-and-Play (PnP), позволяющую динамически обнаруживать и настраивать устройства без необходимости ручного вмешательства. Однако эта автоматизация несёт риски: злоумышленник может подключить вредоносное устройство (например, RubberDucky, имитирующее клавиатуру), которое будет воспринято системой как легитимное. Для противодействия таким угрозам применяются механизмы контроля целостности драйверов. Например, в Windows функция SecureBoot и DriverSignatureEnforcement блокируют загрузку драйверов без цифровой подписи, сертифицированной Microsoft. В Linux аналогичные задачи решаются через модульdm-verity, проверяющий целостность драйверов на основе хэш-сумм.

Важным аспектом безопасности является управление правами доступа. Операционные системы используют модели MandatoryAccessControl (MAC), такие как SELinux или AppArmor, которые ограничивают взаимодействие процессов с устройствами на уровне политик. Например, политика может запрещать веб-браузеру доступ к USB-портам, чтобы предотвратить копирование данных на съёмные носители. В macOS для управления устройствами используется фреймворк EndpointSecurity, который позволяет отслеживать и блокировать подключение неавторизованных девайсов на уровне ядра.

Требуется рассмотреть многослойную модель подсистемы ввода-вывода. Архитектура подсистемы ввода-вывода строится на многоуровневой абстракции, что обеспечивает модульность и снижает риски эксплуатации

уязвимостей. Верхний уровень включает интерфейсы прикладного программирования (API), такие как системные вызовы read() и write() в Unix-подобных системах или Win32 API в Windows. Эти интерфейсы предоставляют приложениям унифицированный доступ к устройствам, скрывая аппаратные особенности. Например, приложение, записывающее данные на жёсткий диск, не должно знать, используется ли магнитная запись или флеш-память.

Средний уровень подсистемы представлен менеджерами ввода-вывода, которые отвечают за буферизацию, кэширование и планирование операций. Менеджеры оптимизируют производительность, минимизируя количество обращений к физическим устройствам. Однако здесь возникают риски, связанные с утечкой данных через кэш. Для их устранения применяются методы шифрования буферов. Например, в современных файловых системах (e.g., ZFS) используется TransparentEncryption, автоматически шифрующий данные перед записью в кэш.

Нижний уровень включает драйверы устройств и непосредственное взаимодействие с аппаратурой. Драйверы транслируют логические запросы в физические команды, понятные контроллерам устройств. Безопасность на этом уровне обеспечивается за счёт изоляции драйверов. В микроядерных ОС (например, QNX) драйверы выполняются в пользовательском пространстве, что предотвращает крах всей системы при их сбое. В Linux для изоляции драйверов используется фреймворк vfio(VirtualFunction I/O), позволяющий запускать их внутри виртуальных машин.

Особое внимание уделяется защите от атак через DMA (DirectMemoryAccess). Устройства с DMA могут напрямую обращаться к памяти, минуя процессор, что создаёт угрозу чтения или модификации данных. Для блокировки таких атак применяются технологии IOMMU (Input-OutputMemoryManagementUnit), которые транслируют физические адреса в виртуальные и контролируют права доступа. Например, в процессорах AMD технология AMD-Vi, а в Intel – VT-d.

Требуется рассмотреть роль менеджеров ввода-вывода в обеспечении безопасности. Менеджеры ввода-вывода выполняют роль посредников между приложениями и драйверами, обеспечивая синхронизацию и управление ресурсами. Одной из их ключевых задач является предотвращение состояний гонки (raceconditions). Например, при одновременной записи двух процессов в один файл менеджер блокирует доступ к сектору диска до завершения первой операции. В распределённых системах для этого используются алгоритмы типа LamportTimestamps или DistributedLocks.

Ещё одной функцией менеджеров является контроль целостности данных. При передаче информации между устройствами применяются механизмы

проверки на основе CRC (CyclicRedundancyCheck) или хеш-функций. Например, при чтении данных с сетевой карты менеджер может проверять контрольные суммы пакетов, чтобы исключить их подмену. В системах хранения данных используются технологии RAID с зеркалированием и чётностью, позволяющие восстанавливать информацию при частичном повреждении.

В контексте безопасности менеджеры также реализуют политики QoS (QualityofService), ограничивающие пропускную способность устройств для предотвращения DDoS-атак. Например, сетевой менеджер может ограничить количество запросов к веб-серверу с одного IP-адреса.

Требуется рассмотреть безопасность драйверов устройств на уровне ядра. Драйверы, работающие в пространстве ядра, представляют повышенную угрозу из-за их привилегированного доступа к ресурсам системы. Уязвимости в драйверах, такие как переполнение буфера или некорректная обработка исключений, позволяют злоумышленникам выполнить произвольный код с правами ядра. Классическим примером является атака BluePill, использующая уязвимости драйверов виртуализации для внедрения гипервизора-руткита.

Для минимизации рисков применяются следующие стратегии:

1. **Верификация драйверов**: Современные ОС требуют цифровой подписи драйверов. В Windows для этого используется WindowsHardwareCompatibilityProgram, а в Linux – модульная система подписей, основанная на сертификатах.

2. **Изоляция драйверов**: Технологии типа User-ModeDriverFramework (UMDF) в Windows позволяют запускать драйверы в пользовательском пространстве. В Linux аналогичную роль играет usermode-helper API.

3. **Мониторинг активности**: Инструменты вроде eBPF (ExtendedBerkeleyPacketFilter) в Linux позволяют отслеживать системные вызовы драйверов и блокировать подозрительные операции.

4. **Защита памяти**: Механизмы DEP (DataExecutionPrevention) и ASLR (AddressSpaceLayoutRandomization) препятствуют эксплуатации уязвимостей, делая адреса исполняемого кода непредсказуемыми.

Примером успешной защиты является исправление уязвимости EternalBlue, эксплуатировавшей ошибку в драйвере SMBv1. Патч Microsoft исключил переполнение буфера, а также добавил дополнительные проверки целостности пакетов.

При работе над курсовой работой обратите внимание на:

– описание особенностей ввода-вывода ОС, которые требуется приводить в первой (теоретической) главе (в соответствии с темой курсовой работы).

Литература

1. Окороков, В. А. Безопасность операционных систем / В. А. Окороков. – Санкт-Петербург: Лань, 2024. – 228 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/367472 (дата обращения: 06.12.2024). – ISBN 978-5-507-48297-9. – Текст : электронный. (Главы 4–6).

2. Киренберг, Г. А. Информационная безопасность современных операционных систем: учеб. пособие / Г. А. Киренберг. – Кемерово: КузГТУ им. Т. Ф. Горбачева, 2022. – 138 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/295736 (дата обращения: 06.12.2024). – ISBN 978-5-00137-320-9. – Текст : электронный. (Глава 1).

3. Потерпеев, Г. Ю. Безопасность операционных систем: учеб. пособие / Г. Ю. Потерпеев, В. С. Нефедов, А. А. Криулин. – Москва: РТУ МИРЭА, 2021. – 93 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/182416 (дата обращения: 06.12.2024). – ISBN 978-5-7339-1393-3. – Текст : электронный. (Глава 2).

4. Зверева, О. М. Операционные системы: учеб. пособие / О. М. Зверева; науч. ред. Л. Г. Доросинский; Уральский федеральный университет им. первого Президента России Б. Н. Ельцина. – Екатеринбург: Издательство Уральского университета, 2020. – 223 с. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=699030 (дата обращения: 03.11.2024). – ISBN 978-5-7996-3146-8. – Текст : электронный. (Глава 6)

Контрольные вопросы

1. Приведите особенности ввода-вывода по отношению к программному модулю, запросившему операцию.
2. Приведите особенности четырёх слоёв системы ввода-вывода.
3. Приведите особенности верхнего слоя менеджера.
4. Приведите особенности нижнего слоя менеджера.
5. Какие механизмы в Windows и Linux предотвращают загрузку неподписанных драйверов и как они влияют на безопасность?
6. Объясните принцип работы IOMMU и его роль в предотвращении DMA-атак. Приведите пример использования в современных процессорах.
7. Почему микроядерные архитектуры считаются более безопасными для изоляции драйверов? Сравните с монолитным ядром.
8. Как состояние гонки при доступе к устройству может быть использовано для атаки? Опишите метод синхронизации, предотвращающий эту угрозу.

9. Какие риски возникают при использовании технологии Plug-and-Play и как они минимизируются?

10. Опишите сценарий атаки через уязвимый драйвер сетевого адаптера. Какие инструменты можно использовать для обнаружения такой атаки?

11. Как механизмы шифрования на уровне менеджеров ввода-вывода (например, в ZFS) защищают данные в кэше?

12. Какие этические и технические проблемы возникают при разработке драйверов с открытым исходным кодом?

Тема 1.7 Особенности обеспечения безопасности файловых систем ОС

Перечень изучаемых вопросов

1. Организация файловой системы (ФС), типы файлов.
2. Иерархическая структура файловой системы.
3. понятие о монтировании.
4. Физическая организация файловой системы.
5. Общая модель файловой системы.
6. Понятие о журналируемых файловых системах
7. Физическая организация и адресация в файле.
8. Файловая система FAT.

Методические указания к изучению

Безопасность файловой системы является одной из наиболее практически значимых тем курса, поскольку именно файловая система хранит пользовательские данные, конфигурации системы, исполняемые файлы и журналы аудита. От того, насколько корректно реализованы механизмы разграничения доступа, целостности и журналирования, зависит устойчивость всей операционной системы к атакам. В книге «Защита и безопасность в сетях Linux» подчёркивается, что Linux – это, по сути, файловая система: все устройства, включая принтеры, дисководы, жёсткие диски и сетевые интерфейсы, представляются в виде файлов. Следовательно, разграничение доступа к устройствам также осуществляется через файловую систему, что делает её центральным элементом подсистемы безопасности.

Предварительно требуется рассмотреть основные цели использования файла и основные функции файловой системы. Файловая система обеспечивает долговременное хранение данных, организацию совместного доступа, защиту от несанкционированного доступа, а также предоставление удобного интерфейса для работы с данными. Требуется рассмотреть типы файлов: обычные файлы, каталоги, специальные файлы устройств (символьные и блочные), именованные каналы (namedpipes), сокеты и символические ссылки. В Linux информация о

типе файла хранится в индексном дескрипторе (inode) и отображается в первом символе вывода команды ls –l (например, d – каталог, c – символьное устройство, b – блочное устройство, s – сокет, p – именованный канал, l – символическая ссылка, – обычный файл).

Требуется рассмотреть иерархическую структуру файловой системы. В Unix-подобных системах используется единое дерево каталогов, начинающееся с корневого каталога/. Различные дисковые разделы и удалённые файловые системы (например, NFS) монтируются в определённые точки этого дерева. Понятие монтирования означает, что содержимое тома проецируется на выбранный каталог файловой системы. Это создаёт специфические риски: если злоумышленник сможет смонтировать файловую систему поверх критических каталогов (например, /binили/etc), он сможет подменить системные файлы. Поэтому операции монтирования строго контролируются: информация о разрешённых точках монтирования содержится в файле/etc/fstab, а обычные пользователи могут монтировать только те файловые системы, для которых указаны параметрыuserилиusers.

Требуется рассмотреть общую модель файловой системы на логическом и физическом уровнях. На логическом уровне файловая система представляет собой иерархию каталогов и файлов, а также набор атрибутов (права доступа, владелец, группа, временные метки). На физическом уровне файлы размещаются в блоках дисковой памяти. Рассмотреть основные методы размещения файлов: непрерывное размещение (файл занимает последовательность смежных блоков), размещение в виде связанного списка кластеров (каждый блок содержит указатель на следующий), использование связанного списка индексов (inode содержит массив указателей на блоки данных). В Unix-подобных системах используется индексный дескриптор (inode), в котором хранится вся метаинформация о файле: тип, права доступа, владелец, группа, размер, временные метки и указатели на блоки данных. Именно в inode хранятся права доступа, и поскольку inode у каждого файла свой, права доступа у каждого файла уникальны.

Требуется рассмотреть понятие о журналируемых файловых системах. Журналирование – это механизм, при котором перед фактическим изменением данных информация о предстоящей операции записывается в специальный журнал. Это позволяет восстановить целостность файловой системы после сбоя питания или аварийного завершения работы. Наиболее распространённые журналируемые файловые системы: ext3, ext4, XFS, ReiserFS, NTFS. Различают три режима журналирования: journal (журналируются и данные, и метаданные), ordered (журналируются только метаданные, но данные записываются до фиксации метаданных) и writeback (журналируются только метаданные, порядок записи данных не гарантируется). С точки зрения безопасности, журналирование

повышает устойчивость системы к атакам типа «отказ в обслуживании», направленным на повреждение файловой системы, но при этом журнал сам может стать объектом атаки (например, для сокрытия следов взлома).

Требуется рассмотреть физическую организацию и адресацию в файле. Адресация может быть прямой (inode содержит адреса первых блоков файла), косвенной (inode содержит адрес блока, в котором хранятся адреса блоков данных) и двойной/тройной косвенной. Такая схема позволяет эффективно работать как с маленькими, так и с большими файлами. С точки зрения безопасности важно понимать, что повреждение inode или блоков косвенной адресации может привести к потере данных или к возможности несанкционированного доступа.

Требуется рассмотреть файловую систему FAT (FileAllocationTable). FAT – это классическая файловая система, используемая в MS-DOS и Windows. Основные особенности: таблица размещения файлов (FAT) хранит цепочки кластеров, принадлежащих каждому файлу; размер таблицы FAT и разрядность используемых в ней индексных указателей определяется количеством кластеров в области данных. Существуют версии FAT12, FAT16, FAT32 и exFAT. Основные ограничения FAT: отсутствие журналирования, отсутствие поддержки прав доступа (в классическом виде), ограничения на размер файла и раздела, а также уязвимость к фрагментации. Файловая система NTFS, пришедшая на смену FAT в Windows NT, поддерживает журналирование, разграничение доступа (ACL), шифрование (EFS), сжатие и квоты. Главная таблица файлов MFT (MasterFileTable) содержит записи обо всех файлах и каталогах тома.

Требуется рассмотреть файловую систему ext (ExtendedFileSystem), которая является стандартной для Linux. Логическая организация файловой системы ext включает: загрузочный блок, суперблок (содержит общую информацию о файловой системе), группы блоков (каждая группа содержит свою копию суперблока, таблицу inode и блоки данных). Структурная организация включает в себя inode, которые содержат метаданные и указатели на блоки данных. Система адресации данных в файловой системе ext использует прямые, косвенные и двойные/тройные косвенные указатели. Три режима журналирования (journal, ordered, writeback) реализованы в ext3/ext4.

Безопасность подсистемы ввода-вывода операционных систем требует комплексного подхода, сочетающего аппаратную защиту, строгую верификацию драйверов и многоуровневую изоляцию компонентов. Изучение этой темы должно включать не только теорию, но и практику анализа уязвимостей, разработки безопасного кода и работы с инструментами мониторинга. Понимание механизмов управления I/O позволяет создавать системы, устойчивые как к классическим угрозам (переполнение буфера), так и к

современным атакам (DMA, BadUSB). Это направление остаётся критически важным в условиях роста числа устройств Интернета вещей (IoT) и увеличения поверхностей атаки в корпоративных сетях.

При работе над курсовой работой обратите внимание на:

– описание особенностей ФС ОС, которые требуется приводить в первой (теоретической) главе (в соответствии с темой курсовой работы).

Литература

1. Зверева, О. М. Операционные системы: учеб. пособие / О. М. Зверева; науч. ред. Л. Г. Доросинский; Уральский федеральный университет им. первого Президента России Б. Н. Ельцина. – Екатеринбург: Издательство Уральского университета, 2020. – 223 с. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=699030 (дата обращения: 03.11.2024). – ISBN 978-5-7996-3146-8. – Текст : электронный. (Глава 7).

Контрольные вопросы

1. Приведите особенности ФС ОС.
2. Приведите особенности ФС FAT
3. Приведите особенности ФС NTFS.
4. Приведите особенности ФС ext.
5. Какие механизмы в Windows и Linux предотвращают загрузку неподписанных драйверов и как они влияют на безопасность?
6. Объясните принцип работы IOMMU и его роль в предотвращении DMA-атак. Приведите пример использования в современных процессорах.
7. Почему микроядерные архитектуры считаются более безопасными для изоляции драйверов? Сравните с монолитным ядром.
8. Как состояние гонки при доступе к устройству может быть использовано для атаки? Опишите метод синхронизации, предотвращающий эту угрозу.
9. Какие риски возникают при использовании технологии Plug-and-Play и как они минимизируются?
10. Опишите сценарий атаки через уязвимый драйвер сетевого адаптера. Какие инструменты можно использовать для обнаружения такой атаки?
11. Как механизмы шифрования на уровне менеджеров ввода-вывода (например, в ZFS) защищают данные в кэше?

Раздел 2. Подсистемы защиты операционной системы

Тема 2.1 Основные подсистемы защиты операционной системы

Перечень изучаемых вопросов:

1. Понятие подсистемы защиты ОС, её место в архитектуре операционной системы.
2. Политика изолированной программной среды: цели, уровни реализации, механизмы.
3. Политика контроля информационных потоков: модели Белла-ЛаПадулы и Биба, их применение.
4. Политика контроля прав доступа: дискреционная, мандатная и ролевая модели.
5. Мандатный контроль доступа в Linux: SELinux и AppArmor.
6. Гранулярные привилегии: механизм capabilities в Linux.
7. Комплексное применение политик защиты на практике.

Методические указания к изучению

Подсистема защиты операционной системы представляет собой совокупность программных и аппаратных механизмов, обеспечивающих разграничение доступа субъектов к объектам, изоляцию процессов, контроль информационных потоков и аудит событий безопасности. Она является центральным элементом обеспечения информационной безопасности, формируя барьеры между потенциальными угрозами и критически важными ресурсами. Без корректно настроенной подсистемы защиты любые прикладные средства безопасности (антивирусы, межсетевые экраны, системы обнаружения вторжений) оказываются бесполезными, поскольку они опираются на доверие к ядру и базовым механизмам разграничения доступа.

Предварительно требуется определить понятие подсистемы защиты и её место в архитектуре ОС. Подсистема защиты работает на нескольких уровнях: аппаратном (например, кольца привилегий процессора, NX-бит, аппаратная поддержка виртуализации), ядерном (разграничение доступа, изоляция адресных пространств, мандатные политики) и пользовательском (утилиты администрирования, библиотеки аутентификации, системы аудита). Такое многоуровневое построение позволяет локализовать потенциальные уязвимости и обеспечить эшелонированную защиту. Важно понимать, что безопасность системы определяется самым слабым звеном, поэтому пренебрежение любым из уровней приводит к снижению общей защищённости.

Требуется рассмотреть**политику изолированной программной среды**, которая реализуется путём изоляции субъектов системы друг от друга и путём контроля порождения новых субъектов таким образом, чтобы в системе могли активизироваться субъекты только из определённого списка. Изоляция в операционных системах – это стратегия, направленная на ограничение взаимодействия между компонентами системы для предотвращения

горизонтального перемещения угроз. Её реализация охватывает несколько уровней.

На аппаратном уровне современные процессоры предоставляют функции виртуализации (Intel VT-x, AMD-V) и защиты памяти (NX-бит, SMAP, SMEP), которые разделяют выполнение кода ядра и пользовательских процессов. Технология NX (NoExecute) помечает определённые области памяти как неисполняемые, блокируя атаки через переполнение буфера. Механизм SMAP (SupervisorModeAccessPrevention) запрещает ядру доступ к пользовательской памяти без явного разрешения, что усложняет эксплуатацию уязвимостей типа «двойное извлечение».

На уровне виртуальной памяти каждый процесс работает в собственном виртуальном адресном пространстве, транслируемом в физические адреса через таблицы страниц. Это предотвращает чтение или модификацию памяти других процессов. В Linux для усиления изоляции используется механизм KASLR (KernelAddressSpaceLayoutRandomization), случайным образом распределяющий адреса ядра, что усложняет эксплуатацию уязвимостей. Дополнительно применяется рандомизация адресного пространства пользовательских процессов (ASLR).

На уровне контейнеризации и микроядер технологии вроде Docker используют пространства имён (namespaces) и контрольные группы (cgroups) для изоляции файловых систем, сетевых интерфейсов и ресурсов. Микроядерные ОС (например, QNX) выполняют драйверы и сервисы в пользовательском пространстве, снижая риск краха всей системы из-за ошибки в одном компоненте. Примером комплексной изоляции является проект gVisor от Google, который запускает контейнеры в легковесной песочнице, эмулируя системные вызовы для минимизации контакта с хост-ядром. Это предотвращает атаки, эксплуатирующие уязвимости ядра, такие как Dirty COW.

Требуется рассмотреть **политику контроля информационных потоков**, которая основана на разделении всех возможных информационных потоков между объектами системы на два непересекающихся множества: благоприятные и неблагоприятные информационные потоки. Цель данной политики – обеспечить невозможность возникновения в системе неблагоприятных информационных потоков. Контроль информационных потоков регулирует передачу данных между субъектами (процессами, пользователями) и объектами (файлами, сетевыми ресурсами). Эта политика базируется на моделях безопасности, адаптированных под различные сценарии.

Модель Белла-ЛаПадулы фокусируется на конфиденциальности, запрещая чтение данных выше уровня допуска субъекта («noreadup») и запись ниже («nowritedown»). Применяется в системах, где требуется защита секретной информации, например, в государственных и военных системах. Агент с

уровнем «секретно» не может прочитать документ «совершенно секретно», но может записать в файл с более низким уровнем.

Модель Биба обеспечивает целостность, запрещая запись данных в объекты с более высоким уровнем целостности («nowriteup») и чтение из объектов с более низким («noreaddown»). Используется в системах, где критична целостность данных, например, в финансовых системах для предотвращения модификации критических транзакций.

Реализация в современных ОС Linux: SELinux и AppArmor реализуют мандатный контроль доступа (MAC), назначая метки конфиденциальности (SELinux) или профили поведения (AppArmor). Например, политика SELinux может запретить веб-серверу доступ к домашним каталогам пользователей, даже если права DAC разрешают. Тегирование данных позволяет помечать файлы метаданными (например, через расширенные атрибуты в ext4), указывающими уровень секретности. Системы DLP используют эти метки для блокировки передачи данных через неавторизованные каналы. Шифрование на транспортном уровне (TLS, VPN) защищает данные в пути, предотвращая перехват.

Сценарий применения: в корпоративной сети DLP-система может анализировать исходящий трафик, блокируя отправку файлов с меткой «конфиденциально» через мессенджеры. Одновременно межсетевой экран (например, iptables) ограничивает доступ к внутренним ресурсам только для IP-адресов филиалов. Аудит информационных потоков позволяет выявить аномалии, например, попытки передачи больших объёмов данных на внешние адреса.

Требуется рассмотреть**политику контроля прав доступа**, которая определяет, кто и как может взаимодействовать с ресурсами системы. Эволюция моделей управления доступом отражает растущие требования к гибкости и безопасности.

Дискреционный контроль доступа (DAC) предполагает, что владелец объекта самостоятельно назначает права. В Unix-системах это реализовано через разрешения rwx и списки ACL (AccessControlLists). Однако DAC уязвим к ошибкам: пользователь может случайно открыть доступ к конфиденциальному файлу. Кроме того, DAC не позволяет реализовать политики, учитывающие контекст (время, местоположение, уровень секретности).

Мандатный контроль доступа (MAC) предполагает, что права определяются системными политиками, а не владельцами объектов. Субъекты и объекты получают метки безопасности, и решение о доступе принимается на основе сравнения этих меток. MAC реализован в SELinux, AppArmor, а также в TrustedBSD. Мандатный контроль позволяет реализовать модель Белла-ЛаПадулы или Биба, обеспечивая более высокий уровень защиты, чем DAC.

Ролевой контроль доступа (RBAC) привязывает права к ролям, а не пользователям. Пользователь получает одну или несколько ролей, и его права определяются этими ролями. RBAC упрощает управление в крупных организациях, но может привести к избыточным привилегиям («ролевой взрыв»). Атрибутивный контроль доступа (ABAC) принимает решения на основе атрибутов (должность, время доступа, местоположение). Например, доступ к бухгалтерской системе разрешён только с корпоративного IP в рабочее время.

Требуется рассмотреть**мандатный контроль доступа в Linux: SELinux и AppArmor**. SELinux (Security-EnhancedLinux) реализует модель принудительного контроля доступа (MAC), где права определяются политиками, а не владельцами. Каждому файлу и процессу назначается метка (контекст безопасности), например, user_u:object_r:httpd_sys_content_t. Политика SELinux определяет, какие действия разрешены для каждого контекста. Например, политика может запретить процессу Apache (httpd_t) доступ к файлам с меткойuser_home_t, даже если классические права разрешают чтение. Инструментыsemanageиaudit2allowпозволяют управлять контекстами и генерировать правила из логов нарушений.

AppArmor использует профили поведения, привязанные к путям исполняемых файлов. Профиль определяет, к каким файлам, сетевым портам и другим ресурсам может обращаться программа. AppArmor проще в настройке, чем SELinux, но менее гибок. Обе системы могут использоваться совместно, хотя на практике обычно выбирают одну.

Требуется рассмотреть**гранулярные привилегии: механизм capabilities в Linux**. Вместо предоставления полного доступа root, Linux позволяет назначать отдельные привилегии через capabilities. Примеры capabilities:CAP_NET_BIND_SERVICE – привязка к портам <1024;CAP_SYS_ADMIN – администрирование системы; CAP_DAC_OVERRIDE – игнорирование прав доступа к файлам. Назначение выполняется командойsetcap, например: setcapcap_net_bind_service=+ep /usr/bin/my_server. Риски: злоупотребление capabilities (например, CAP_DAC_OVERRIDE) может обойти классические права. Поэтому назначение capabilities должно быть тщательно обосновано.

Требуется рассмотреть **комплексное применение политик защиты на практике**. В реальной системе все три политики применяются совместно. Например, на файловом сервере могут быть настроены: изоляция процессов через namespaces и cgroups; мандатные политики SELinux для разграничения доступа к файлам различных отделов; дискреционные права для гибкого управления доступом внутри отдела; capabilities для сервисов, которым требуются отдельные привилегии. Аудит позволяет контролировать соблюдение политик и выявлять нарушения. Важно понимать, что безопас-

ность – это процесс, а не состояние: политики должны периодически пересматриваться, а механизмы защиты – обновляться.

При работе над курсовой работой обратите внимание на:

- описание особенностей политик защиты в ОС Linux, которые требуется приводить в первой (теоретической) главе (в соответствии с темой курсовой работы);
- обоснование выбора конкретных механизмов защиты для рассматриваемого сценария;
- анализ рисков, связанных с неправильной настройкой подсистемы защиты, и меры противодействия.

Литература

1. Окороков, В. А. Безопасность операционных систем / В. А. Окороков. – Санкт-Петербург: Лань, 2024. – 228 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/367472 (дата обращения: 06.12.2024). – ISBN 978-5-507-48297-9. – Текст : электронный. (Глава 7).

2. Киренберг, Г. А. Информационная безопасность современных операционных систем: учеб. пособие / Г. А. Киренберг. – Кемерово: КузГТУ им. Т. Ф. Горбачева, 2022. – 138 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/295736 (дата обращения: 06.12.2024). – ISBN 978-5-00137-320-9. – Текст : электронный. (Глава 3).

3. Потерпеев, Г. Ю. Безопасность операционных систем: учеб. пособие / Г. Ю. Потерпеев, В. С. Нефедов, А. А. Криулин. – Москва : РТУ МИРЭА, 2021. – 93 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/182416 (дата обращения: 06.12.2024). – ISBN 978-5-7339-1393-3. – Текст : электронный. (Главы 1, 4)

4. Зверева, О. М. Операционные системы: учеб. пособие / О. М. Зверева; науч. ред. Л. Г. Доросинский; Уральский федеральный университет им. первого Президента России Б. Н. Ельцина. – Екатеринбург: Издательство Уральского университета, 2020. – 223 с. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=699030 (дата обращения: 03.11.2024). – ISBN 978-5-7996-3146-8. – Текст : электронный. (Глава 11).

Контрольные вопросы

1. Приведите особенности матриц доступа.
2. Приведите особенности контроля информационных потоков

3. Приведите особенности контроля прав доступа.

4. Приведите особенности моделей прав доступа.

5. Какие аппаратные технологии (Intel VT-x, NX bit) обеспечивают изоляцию процессов и как они интегрированы в ОС?

6. Объясните, как модель Биба предотвращает модификацию финансовых транзакций. Приведите пример из банковской системы.

7. Почему ролевая модель (RBAC) может привести к «ролевому взрыву» и как это решается в ABAC?

8. Как KASLR усложняет эксплуатацию уязвимостей ядра? Опишите на примере атаки на драйвер устройства.

9. Какие инструменты в Linux позволяют назначать capabilities вместо полных прав root? Приведите сценарий использования.

10. Как MandatoryIntegrityControl в Windows ограничивает вредоносное ПО с низким уровнем целостности?

11. Опишите процесс настройки DLP для блокировки передачи конфиденциальных данных через USB. Какие метаданные используются?

12. Какие уроки можно извлечь из инцидента с Equifax для улучшения политик контроля прав доступа?

Тема 2.2 Управление доступом в операционных системах семейства Linux

Перечень изучаемых вопросов:

1. Понятие субъекта и объекта доступа в Linux. Идентификаторы пользователя и группы.

2. Файлы учётных записей:/etc/passwd,/etc/shadow,/etc/group,/etc/gshadow.

3. Классическая модель прав доступа: владелец, группа, остальные. Команды chmod, chown, chgrp, umask.

4. Специальные биты: SUID, SGID, StickyBit. Их назначение и риски безопасности.

5. Списки управления доступом (ACL): назначение, команды setfacl, getfacl, маски.

6. Атрибуты файловой системы ext:chattr, lsattr, immutable и append-only.

7. Мандатный контроль доступа: SELinux и AppArmor. Контексты безопасности и политики.

8. Гранулярные привилегии: capabilities в Linux.

9. Управление доступом к устройствам и специальным файлам.

Методические указания к изучению

Управление доступом в операционных системах семейства Linux является основой их безопасности, формируя барьеры между пользователями, процессами и критически важными ресурсами. С момента создания UNIX в 1969 году управление доступом эволюционировало от простой дискреционной модели до сложных систем, включающих мандатный контроль, списки доступа и расширенные атрибуты. Сегодня, в эпоху облачных вычислений и контейнеризации, эти механизмы остаются актуальными, но требуют глубокого понимания для противодействия таким угрозам, как горизонтальное перемещение в сетях, эксплуатация привилегированных процессов и утечки данных.

Построение файловой системы и разграничение доступа к файловым объектам имеет особенности, присущие данному семейству ОС. Все дисковые накопители (тома) объединяются в единую виртуальную файловую систему путём операции монтирования тома. При этом содержимое тома проецируется на выбранный каталог файловой системы. Элементами файловой системы являются также все устройства, подключаемые к защищаемому компьютеру (монтируемые к файловой системе). Поэтому разграничение доступа к ним осуществляется через файловую систему. Каждый файловый объект имеет индексный дескриптор (inode), в котором среди прочего хранится информация о разграничении доступа к данному файловому объекту. Права доступа делятся на три категории: доступ для владельца, доступ для группы и доступ для остальных пользователей. В каждой категории определяются права на чтение, запись и исполнение (в случае каталога – просмотр).

Предварительно требуется определить понятия субъекта и объекта доступа в Linux. Субъект доступа – это активная сущность, которая может выполнять операции над объектами: пользователь, процесс, служба. Объект доступа – это пассивная сущность, над которой выполняются операции: файл, каталог, устройство, сокет, сегмент памяти. Каждый субъект в UNIX-системе идентифицируется уникальным числовым идентификатором пользователя (UID) и списком групп (GID). Эти идентификаторы определяют права доступа к файлам, сетевым ресурсам и системным вызовам.

Структура системных файлов учётных записей:

- /etc/passwd: содержит записи о пользователях в формате username:x:UID:GID:GECOS:/home/dir:/bin/sh. Поле x указывает, что пароль хранится в зашифрованном виде в/etc/shadow, защищённом от чтения обычными пользователями. UID (User ID) и GID (Group ID) определяют права доступа. Корневой каталог и оболочка задают среду пользователя.

- /etc/group: определяет группы и их GID, а также список членов. Например, строка developers:x:1001:alice,bob создаёт группу developersc пользователями Alice и Bob.
- /etc/shadow: хранит хэши паролей, соль (salt) и параметры истечения сроков. Современные системы используют алгоритмы вроде SHA-512 с 8-байтовой солью для замедления атак перебора.
- /etc/gshadow: используется для хранения теневых паролей групп, если они заданы.

Привилегии суперпользователя. Учётная запись root имеет UID 0 и обладает неограниченными правами, включая доступ к/dev/mem(прямое управление памятью) и изменение системных конфигураций. Работа под root крайне рискованна – даже случайная командarm -rf /может уничтожить систему. Инструменты делегирования привилегий: sudoisu. Файл/etc/sudoersопределяет, какие команды пользователи могут выполнять с повышенными правами. Например, строкаalice ALL=(ALL) NOPASSWD: /usr/bin/systemctlrestartapache позволяет Alice перезапускать Apache без ввода пароля.

Процесс наследует реальный UID/GID пользователя, запустившего его. Однако при выполнении файла с SUID-битом (например, /usr/bin/passwd) процесс получает эффективный UID владельца файла (в данном случаеroot). Это позволяет выполнять привилегированные операции без постоянных прав суперпользователя. Если программа с SUID содержит ошибку (например, переполнение буфера), злоумышленник может выполнить произвольный код с правами владельца. Пример – уязвимость CVE-2021-4034 в pkexec, где ошибка обработки аргументов позволила локальную эскалацию привилегий.

Требуется рассмотреть **классическую модель прав доступа**. Классическая модель прав доступа в UNIX базируется на трёх категориях: владелец (user), группа (group) и остальные (others). Каждой категории назначаются права на чтение (r), запись (w) и выполнение (x). Для файлов: r – чтение содержимого, w – модификация или удаление, x – запуск как исполняемого файла. Для каталогов: r – просмотр списка файлов, w – создание или удаление файлов, x – доступ к содержимому файлов (требуется для cd). Пример: права drwxr-xr-- для каталога означают, что владелец может читать, записывать и открывать каталог; группа – читать и открывать; остальные – только читать.

Управление правами осуществляется командами chmod (изменение прав), chown (изменение владельца), chgrp (изменение группы). Значение umask определяет права по умолчанию для вновь создаваемых файлов и каталогов. Ограничения классической модели: негибкость (невозможно назначить права для нескольких пользователей или групп вне основной тройки), отсутствие контекстного доступа (права не учитывают время, местоположение или тип

операции), слабый контроль процессов (нет механизмов ограничения действий процессов).

Требуется рассмотреть **специальные биты: SUID, SGID, StickyBit**. SUID (Set User ID) устанавливается командой chmodu+s file. Пример: passwd требует прав root для изменения/etc/shadow, поэтому имеет SUID-бит:-rwsr-xr-x. Риск: устаревшие SUID-программы (например,mount) могут стать вектором атак. SGID (SetGroup ID) для файлов означает запуск с эффективным GID владельца. Для каталогов новые файлы наследуют группу каталога, а не создателя. Полезно для совместной работы (например, общий каталог/var/wwwc SGID). StickyBit устанавливается командой chmod +t dir. В каталогах (например, /tmp) запрещает удаление файлов невладельцами. Без StickyBit пользователь мог бы удалить файлы других, если у каталога есть правow.

Требуется рассмотреть **списки управления доступом (ACL)**. ACL позволяют назначать права для конкретных пользователей и групп, игнорируя базовые категории. Установка прав: setfacl -m u:alice:rwx,g:developers:r-- file.txt. Просмотр ACL:getfacl file.txt. Маски ACL ограничивают максимальные права для групп (аналогично umask). ACL позволяют реализовать более гибкое разграничение доступа, но требуют поддержки со стороны файловой системы (ext4, XFS, Btrfs поддерживают ACL).

Требуется рассмотреть **атрибуты файловой системы ext**. Утилита chattr управляет атрибутами файлов на уровне файловой системы: immutable (+i) запрещает изменение, удаление и переименование файла даже для root (chattr +i /etc/resolv.conf); append-only (+a) разрешает только добавление данных (полезно для логов); securedeletion (+s) обеспечивает перезапись данных нулями перед удалением. Просмотр атрибутов выполняется командойlsattr. Атрибуты являются мощным средством защиты критических файлов от случайного или злонамеренного изменения.

Требуется рассмотреть **мандатный контроль доступа: SELinux и AppArmor**. SELinux (Security-EnhancedLinux) реализует модель принудительного контроля доступа (MAC), где права определяются политиками, а не владельцами. Каждому файлу и процессу назначается метка (контекст безопасности), например, user_u:object_r:httpd_sys_content_t. Политика SELinux определяет, какие действия разрешены для каждого контекста. Например, политика может запретить процессу Apache (httpd_t) доступ к файлам с меткойuser_home_t, даже если классические права разрешают чтение. Инструменты semanageиaudit2allow позволяют управлять контекстами и генерировать правила из логов нарушений. AppArmor использует профили поведения, привязанные к путям исполняемых файлов. Профиль определяет, к каким файлам, сетевым портам и другим ресурсам может обращаться программа. AppArmor проще в настройке, чем SELinux, но менее гибок.

Требуется рассмотреть**гранулярные привилегии: capabilities в Linux**. Вместо предоставления полного доступа root, Linux позволяет назначать отдельные привилегии через capabilities. Примеры capabilities:CAP_NET_BIND_SERVICE – привязка к портам <1024;CAP_SYS_ADMIN – администрирование системы; CAP_DAC_OVERRIDE – игнорирование прав доступа к файлам. Назначение выполняется командой setcap, например: setcapcap_net_bind_service=+ep /usr/bin/my_server. Риски: злоупотребление capabilities (например, CAP_DAC_OVERRIDE) может обойти классические права. Поэтому назначение capabilities должно быть тщательно обосновано.

Требуется рассмотреть **управление доступом к устройствам и специальным файлам**. В Linux все устройства представляются в виде файлов, расположенных в каталоге/dev. Разграничение доступа к устройствам осуществляется через файловую систему. Права на чтение и запись для устройств должны быть ограничены, поскольку доступ к устройству может означать доступ к данным или возможность нарушения работы системы. Например, доступ к/dev/memпозволяет напрямую читать и записывать физическую память, что является критической привилегией. Доступ к сетевым интерфейсам, дискам, терминалам также регулируется через права доступа к соответствующим файлам устройств. Рекомендуется использовать udev для управления правами на устройства и избегать предоставления обычным пользователям доступа к привилегированным устройствам.

Для преодоления ограничений классической модели UNIX-системы используют расширенные механизмы управления доступом, которые должны применяться комплексно. Например, на файловом сервере могут быть настроены: классические права для базового разграничения, ACL для гибкого управления доступом групп, атрибутыimmutableдля защиты критических конфигураций, SELinux для мандатного контроля, capabilities для сервисов, которым требуются отдельные привилегии. Аудит позволяет контролировать соблюдение политик и выявлять нарушения. Важно понимать, что безопас-ность – это процесс, а не состояние: политики должны периодически пересматриваться, а механизмы защиты – обновляться.

При работе над курсовой работой обратите внимание на:

- описание особенностей политики прав доступа в ОС Linux, которые требуется приводить в первой (теоретической) главе (в соответствии с темой курсовой работы);
- обоснование выбора конкретных механизмов разграничения доступа для рассматриваемого сценария;

– анализ рисков, связанных с неправильной настройкой прав доступа, и меры противодействия.

Литература

1. Окороков, В. А. Безопасность операционных систем / В. А. Окороков. – Санкт-Петербург: Лань, 2024. – 228 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/367472 (дата обращения: 06.12.2024). – ISBN 978-5-507-48297-9. – Текст : электронный. (Главы 7, с. 201–210).

2. Киренберг, Г. А. Информационная безопасность современных операционных систем: учеб. пособие / Г. А. Киренберг. – Кемерово: КузГТУ им. Т. Ф. Горбачева, 2022. – 138 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/295736 (дата обращения: 06.12.2024). – ISBN 978-5-00137-320-9. – Текст : электронный. (Глава 3).

3. Потерпеев, Г. Ю. Безопасность операционных систем: учеб. пособие / Г. Ю. Потерпеев, В. С. Нефедов, А. А. Криулин. – Москва: РТУ МИРЭА, 2021. – 93 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/182416 (дата обращения: 06.12.2024). – ISBN 978-5-7339-1393-3. – Текст : электронный. (Главы 4).

4. Зверева, О. М. Операционные системы: учеб. пособие / О. М. Зверева; науч. ред. Л. Г. Доросинский; Уральский федеральный университет им. первого Президента России Б. Н. Ельцина. – Екатеринбург: Издательство Уральского университета, 2020. – 223 с. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=699030 (дата обращения: 03.11.2024). – ISBN 978-5-7996-3146-8. – Текст : электронный. (Глава 11).

Контрольные вопросы

1. Дайте определение субъекта и объекта доступа в Linux. Какие идентификаторы используются для разграничения прав?
2. Опишите структуру файлов /etc/passwd,/etc/shadow,/etc/group. Какие поля они содержат и для чего используются?
3. В чём отличие реального и эффективного UID? Приведите пример использования SUID-бита.
4. Объясните назначение специальных битов SUID, SGID и StickyBit. Какие риски безопасности с ними связаны?
5. Что такое ACL и в каких случаях их использование предпочтительнее классической модели прав доступа?

6. Какие атрибуты файловой системы ext позволяют защитить файлы от изменения или удаления? Приведите примеры использованияchattr.

7. Сравните SELinux и AppArmor. В чём их сходство и различия? Какой механизм предпочтительнее в каких сценариях?

8. Что такое capabilities в Linux и как они позволяют реализовать принцип наименьших привилегий? Приведите пример назначения capabilities.

9. Как осуществляется управление доступом к устройствам в Linux? Почему доступ к/dev/memявляется критической привилегией?

10. Приведите пример комплексного применения механизмов разграничения доступа для защиты файлового сервера.

Тема 2.3 Подсистема идентификации и аутентификации в ОС Linux

Перечень изучаемых вопросов

1. Стандартная процедура идентификации и аутентификации.
2. Основные механизмы защиты паролей.
3. Шифрование паролей.

Методические указания к изучению

Тема «Подсистема идентификации и аутентификации в ОС Linux» занимает центральное место в курсе «Безопасность операционных систем», поскольку эти процессы являются основой обеспечения контролируемого доступа к ресурсам системы. В условиях современных киберугроз, когда несанкционированный доступ может привести к компрометации данных или нарушению работы системы, глубокое понимание механизмов идентификации и аутентификации становится критически важным. В рамках данной темы рассматриваются стандартная процедура идентификации и аутентификации, основные механизмы защиты паролей и методы их шифрования в контексте ОС Linux.

Предварительно требуется определить понятия идентификации, аутентификации и авторизации, а также понять их различия. Идентификация – это процесс присвоения субъекту уникального идентификатора (имени пользователя, UID), по которому система будет его узнавать. Аутентификация – это процесс проверки подлинности субъекта, то есть подтверждение того, что субъект действительно является тем, за кого он себя выдаёт. Авторизация – это процесс определения того, какие ресурсы и операции доступны аутентифицированному субъекту. Эти три процесса образуют последовательную цепочку: сначала система идентифицирует пользователя, затем проверяет его подлинность, и только после успешной аутентификации предоставляет доступ к ресурсам в соответствии с правами, определёнными при авторизации.

Идентификация и аутентификация в ОС Linux представляют собой двухэтапный процесс, обеспечивающий определение личности пользователя и подтверждение его права доступа к системе. Идентификация начинается с того момента, когда пользователь предоставляет уникальный идентификатор, чаще всего в виде имени пользователя (login), которое система сопоставляет с записями в соответствующих базах данных, таких как файл/etc/passwd. Этот файл содержит информацию о пользователях, включая их имена, идентификаторы (UID), домашние директории и используемые оболочки. Успешная идентификация позволяет системе установить, с кем она имеет дело, но не гарантирует доступа – для этого требуется аутентификация.

Аутентификация в стандартной процедуре предполагает проверку предоставленного пользователем секрета, обычно пароля, который сравнивается с хранимой в системе информацией. В Linux пароли традиционно хранятся в зашифрованном виде в файле/etc/shadow, доступном только привилегированным пользователям. Процесс аутентификации осуществляется через вызовы к библиотекам PAM (PluggableAuthenticationModules), которые предоставляют гибкий и модульный подход к проверке подлинности. PAM позволяет интегрировать различные методы аутентификации – от паролей до биометрических данных – и настраивать их через конфигурационные файлы в директории/etc/pam.d/ или основной файл/etc/pam.conf. После успешной аутентификации система предоставляет пользователю доступ к его окружению, определяемому правами и настройками, указанными в системных файлах.

Требуется рассмотреть файлы учётных записей. Файл /etc/passwd содержит записи о пользователях в формате username:x:UID:GID:GECOS:/home/dir:/bin/bash. Поле x указывает, что пароль хранится в зашифрованном виде в/etc/shadow. UID (User ID) и GID (Group ID) определяют права доступа. Корневой каталог и оболочка задают среду пользователя. Файл/etc/shadow хранит зашифрованные пароли и параметры безопасности в формате username:6salt$hash:last_changed:min_age:max_age:warn:inactive:expire. Поля min_ageимax_age задают минимальный и максимальный срок действия пароля. Поле inactive определяет период блокировки учётной записи после истечения пароля. Поле expire задаёт абсолютную дату истечения срока действия учётной записи. Файл /etc/grouопределяет группы и их GID, а также список членов. Файл /etc/gshadow используется для хранения теневых паролей групп, если они заданы.

Требуется рассмотреть парольную аутентификацию и алгоритмы хэширования. Шифрование паролей в ОС Linux является неотъемлемой частью процесса аутентификации, обеспечивая защиту учётных данных от прямого доступа даже в случае компрометации файла /etc/shadow. В отличие от хранения

паролей в открытом виде, что было характерно для ранних систем, современные версии Linux используют криптографические хэш-функции для преобразования паролей в необратимые последовательности символов. Исторически первой широко используемой функцией была DES (DataEncryptionStandard), которая, однако, имела ограничения по длине пароля (до 8 символов) и уязвимость к атакам перебора из-за небольшого размера ключа. Впоследствии DES была заменена более надёжными алгоритмами, такими как MD5, который стал стандартом в 1990-х годах благодаря большей длине хэша и устойчивости к простым атакам.

На сегодняшний день в Linux чаще всего применяется алгоритм SHA-512, интегрированный в библиотекуcrypt, которая используется для шифрования паролей. SHA-512 обеспечивает высокий уровень криптографической стойкости благодаря 512-битному хэшу и использованию соли (salt) – случайной строки, добавляемой к паролю перед хэшированием. Соль предотвращает атаки с использованием радужных таблиц (rainbowtables), где заранее вычисленные хэши сравниваются с украденными данными, поскольку каждый пароль с уникальной солью генерирует уникальный хэш. Например, запись в /etc/shadowможет выглядеть как6<salt>$<hash>, где$6$указывает на SHA-512, за которым следуют соль и результат хэширования. В новых версиях Linux всё чаще применяется алгоритм yescrypt, устойчивый к атакам с использованием GPU.

Требуется рассмотреть механизм «теневых паролей» и защиту файла /etc/shadow. Механизм теневых паролей был разработан для того, чтобы ограничить доступ к хэшированным паролям. Ранее пароли хранились в файле /etc/passwd, который был доступен для чтения всем пользователям. Это создавало риск того, что злоумышленник, получивший доступ к системе с низкими привилегиями, мог скопировать файл паролей и попытаться взломать их в автономном режиме. Механизм теневых паролей переносит хэши паролей в файл /etc/shadow, доступный только пользователю root и группе shadow. Права доступа к файлу /etc/shadowобычно устанавливаются как640или000, что минимизирует риск несанкционированного чтения. В файле /etc/passwdна месте пароля остаётся символх, указывающий на необходимость обращения к теневому файлу.

Требуется рассмотреть подключаемые модули аутентификации PAM. PAM (PluggableAuthenticationModules) предоставляет гибкую инфраструктуру для настройки аутентификации. Конфигурационные файлы в/etc/pam.d/ определяют, какие модули используются для проверки паролей, управления сессиями и контроля доступа. Каждая строка конфигурационного файла содержит тип модуля (auth, account, session, password), управляющий флаг (required, requisite, sufficient, optional) и имя модуля. Например, модуль

pam_unix.so проверяет пароль через /etc/shadow, apam_tally2.so или pam_faillock.so блокирует учётную запись после нескольких неудачных попыток. PAM позволяет легко менять методы аутентификации без перекомпиляции приложений, что делает его мощным и гибким инструментом.

Требуется рассмотреть политики сложности паролей и блокировку учётных записей. Утилита pam_pwquality настраивается через /etc/security/pwquality.conf и контролирует минимальную длину пароля (minlen), наличие разных классов символов (цифры, заглавные буквы, специальные символы), запрет простых последовательностей (12345) или повторяющихся символов (aaaa). Модульpam_faillock.so фиксирует неудачные попытки входа в/var/log/faillock и блокирует пользователя после превышения лимита (например, 5 попыток). Это снижает вероятность успешных атак методом перебора (bruteforce) или использования словарей. Дополнительно используется ограничение срока действия паролей, задаваемое в/etc/shadow через параметры max_age (обычно 90 дней) и warn (период предупреждения перед истечением срока).

Требуется рассмотреть многофакторную аутентификацию в Linux. Для критически важных систем применяется двухфакторная аутентификация (2FA), где пароль дополняется вторым фактором. TOTP (Time-BasedOne-TimePassword) реализуется через интеграцию с GoogleAuthenticator или Authy с использованием модулярam_google_authenticator.so. Аппаратные токены, такие как YubiKey, поддерживаются через модульpam_u2f.so. SSH-ключи являются ещё одним распространённым методом аутентификации: пользователь генерирует пару ключей (публичный и приватный) командойssh-keygen, публичный ключ добавляется в~/.ssh/authorized_keysна сервере, и при подключении клиент доказывает владение приватным ключом через криптографический алгоритм (например, Ed25519). Это позволяет отказаться от паролей и повысить безопасность.

Требуется рассмотреть аутентификацию через SSH и настройку sshd_config. В файле/etc/ssh/sshd_config можно настроить следующие параметры, влияющие на безопасность: PermitRootLogin (запрет прямого входа root), PasswordAuthentication (отключение парольной аутентификации), PubkeyAuthentication (разрешение ключевой аутентификации), MaxAuthTries (ограничение числа попыток), LoginGraceTime (время на аутентификацию), AllowUsers и DenyUsers (списки разрешённых и запрещённых пользователей). Правильная настройка SSH является критически важной, поскольку SSH часто используется для удалённого администрирования серверов.

Требуется рассмотреть централизованную аутентификацию: LDAP, Kerberos, SSSD. В корпоративных средах с большим количеством серверов и пользователей централизованное управление учётными записями становится

необходимостью. LDAP (LightweightDirectoryAccessProtocol) используется для хранения информации о пользователях и группах в центральном каталоге. Kerberos обеспечивает протокол аутентификации с использованием билетов (tickets), защищённых сессионными ключами. SSSD (SystemSecurityServicesDaemon) обеспечивает интеграцию Linux-систем с LDAP и Kerberos, позволяя пользователям входить в систему с доменными учётными данными.

Требуется рассмотреть аудит событий аутентификации. Все попытки входа в систему, как успешные, так и неудачные, регистрируются в журналах. В зависимости от конфигурации, информация может записываться в /var/log/auth.log(Debian/Ubuntu) или /var/log/secure(RedHat/CentOS). Утилиты last и lastlog позволяют просмотреть историю входов пользователей. Анализ журналов аутентификации позволяет выявить попытки подбора паролей, несанкционированный доступ, а также аномальную активность. Утилита Fail2Ban анализирует логи и блокирует IP-адреса, выполняющие подозрительные попытки входа.

При работе над курсовой работой обратите внимание на:

- описание особенностей механизма аутентификации в ОС Linux, которые требуется приводить в первой (теоретической) главе (в соответствии с темой курсовой работы);
- обоснование выбора методов аутентификации для рассматриваемого сценария;
- анализ рисков, связанных с неправильной настройкой подсистемы аутентификации, и меры противодействия.

Литература

1. Окороков, В. А. Безопасность операционных систем / В. А. Окороков. – Санкт-Петербург: Лань, 2024. – 228 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/367472 (дата обращения: 06.12.2024). – ISBN 978-5-507-48297-9. – Текст : электронный. (Главы 7).

2. Киренберг, Г. А. Информационная безопасность современных операционных систем: учеб. пособие / Г. А. Киренберг. – Кемерово: КузГТУ им. Т. Ф. Горбачева, 2022. – 138 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/295736 (дата обращения: 06.12.2024). – ISBN 978-5-00137-320-9. – Текст : электронный. (Глава 3).

3. Потерпеев, Г. Ю. Безопасность операционных систем: учеб. пособие / Г. Ю. Потерпеев, В. С. Нефедов, А. А. Криулин. – Москва: РТУ МИРЭА,

2021. – 93 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/182416 (дата обращения: 06.12.2024). – ISBN 978-5-7339-1393-3. – Текст : электронный. (Главы 1, 4).

4. Зверева, О. М. Операционные системы: учеб. пособие / О. М. Зверева; науч. ред. Л. Г. Доросинский; Уральский федеральный университет им. первого Президента России Б. Н. Ельцина. – Екатеринбург: Издательство Уральского университета, 2020. – 223 с. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=699030 (дата обращения: 03.11.2024). – ISBN 978-5-7996-3146-8. – Текст : электронный. (Глава 11).

Контрольные вопросы:

1. Приведите особенности аутентификации в ОС Linux.
2. Приведите особенности использования паролей в ОС Linux.
3. Приведите особенности механизмы контроля паролей в ОС Linux.
4. Приведите особенности моделей прав доступа в ОС Linux.
5. Как стандартная процедура идентификации и аутентификации в Linux использует файлы /etc/passwdи/etc/shadow, и какие уязвимости могут возникнуть при их некорректной настройке?
6. Каким образом модули PAM обеспечивают гибкость аутентификации, и как ошибки в их конфигурации могут повлиять на безопасность системы?
7. Почему разделение данных между файлами /etc/passwdи/etc/shadow повышает безопасность паролей, и какие меры защищают файл /etc/shadowот несанкционированного доступа?
8. Как механизмы контроля качества паролей, такие какpam_passwdqc, снижают риск атак методом перебора, и какие ограничения они имеют?
9. В чём преимущества использования SHA-512 для шифрования паролей по сравнению с DES и MD5, и как соль усиливает криптографическую стойкость?
10. Какие угрозы безопасности возникают при использовании устаревших алгоритмов шифрования паролей, и как их можно выявить в современной системе Linux?
11. Как ограничение срока действия паролей в/etc/shadowвлияет на защиту системы, и какие проблемы могут возникнуть при его неправильной настройке?
12. Каким образом злоумышленник может использовать слабые пароли для компрометации системы, и какие методы тестирования позволяют оценить их устойчивость к атакам?

Тема 2.4 Обеспечение сетевой безопасности в ОС Linux

Перечень изучаемых вопросов:

1. Основные понятия сетевого взаимодействия в Linux.
2. Управление доступом к сетевым службам: TCP Wrappers.
3. Фильтрация пакетов: ipchains и netfilter.
4. Маскировка IP и перенаправление портов.
5. Безопасность сетевых служб и демонов.

Методические указания к изучению:

Сетевая безопасность в ОС Linux является одной из наиболее сложных и в то же время наиболее востребованных областей обеспечения информационной безопасности. В книге «Защита и безопасность в сетях Linux» подчёркивается, что основной причиной, по которой была написана эта книга, стало возрастающее количество домашних пользователей, которые соединены с Интернетом 24 часа в сутки семь дней в неделю. К этой аудитории присоединяются и небольшие предприятия, осознавшие важность Web и пришедшие к выводу, что поддержка своих web-ресурсов собственными силами обойдётся им дешевле, чем использование услуг сторонних компаний. Однако при этом руководители подобных предприятий понимают, что они не могут позволить себе принять на работу специального сотрудника, который является экспертом в области безопасности. Именно поэтому знание базовых механизмов сетевой безопасности Linux необходимо каждому администратору, даже начинающему.

Основные понятия сетевого взаимодействия в Linux. Любая атака через сеть возможна только через те службы, которые вы делаете доступными для внешнего мира. Если вы завершаете работу службы или запрещаете доступ к ней, вы предотвращаете доступ к системе извне с использованием этой службы. Если на сервере не работает ни одна служба, сервер недоступен для атак через сеть. С другой стороны, если на сервере не работает ни одна служба, сервер не в состоянии обслужить ни одного клиента. Это фундаментальное противоречие определяет всю логику построения защищённой сетевой конфигурации: вы должны предоставить минимально необходимый набор служб и обеспечить их максимально возможную защиту.

Любая сетевая служба в Linux связана с определённым портом. Порты с номерами ниже 1024 считаются привилегированными: только пользователь root может связать службу с таким портом. Порты с номерами выше 1024 могут использоваться любыми пользователями. Информация о соответствии между именами служб и номерами портов хранится в файле /etc/services. Утилитаnetstatявляется наилучшим инструментом анализа состояния сети на

любом сетевом узле. Запущенная с ключом-a, она отображает состояние всех сетевых соединений, включая серверные программы, ожидающие поступления запросов. Если утилита netstat показывает открытый порт, за которым не стоит известная вам служба, это может указывать на наличие «троянского коня» или другого скрытого средства удалённого доступа.

Управление доступом к сетевым службам: TCP Wrappers. Программное средство TCP Wrappers (оболочка TCP) – это программа, специально предназначенная для обнаружения и предотвращения попыток взлома системы. Как правило, это программное средство работает совместно cinetd, однако многие другие серверные программы, которые не запускаются с использованиеminetd, могут быть скомпилированы с использованием библиотеки TCP Wrappers, благодаря чему они станут поддерживать эту функциональность. Чтобы понять, как настраивается TCP Wrappers, необходимо овладеть приёмами конфигурации файлов /etc/hosts.allowи/etc/hosts.deny.

Изначально для функционирования tcpd требуются оба файла: /etc/hosts.allowи/etc/hosts.deny. Однако при желании вы можете объединить всю необходимую информацию о разрешённых и запрещённых соединениях в одном файле/etc/hosts.allowи в дальнейшем работать только с одним этим файлом. Записи файла /etc/hosts.allow обладают следующим форматом: демон(ы) : клиент(ы) : параметр : параметр .... Каждая строка – это разделённый двоеточиями список, состоящий из двух или большего количества колонок. В первой колонке указывается демон или демоны, доступ к которым вы желаете обезопасить. Во второй колонке перечисляются клиенты, в отношении которых действует эта запись. В третьей и последующих колонках перечисляются параметры. Это могут быть программы, которые требуется запустить, или действия, которые требуется выполнить.

TCP Wrappers не является абсолютной защитой, однако с её помощью можно обнаружить попытку взлома и сохранить в журнале сведения об источнике подобных попыток. Благодаря этому вы сможете сделать так, чтобы эти попытки были безуспешными. Однако сама по себе программа TCP Wrappers не делает абсолютно всего. Для активной борьбы с последующими атаками следует использовать другие программы, например, IP Chains или Netfilter. Программу TCP Wrappers не следует использовать совместно с web-сервером Apache, поскольку этот сервер обладает встроенным механизмом TCP Wrappers и использование совместно с ним программыtcpdявляется излишеством.

Фильтрация пакетов: ipchains и netfilter. Термин «брандмауэр» пришёл в компьютерную индустрию из автомобилестроения. Этим термином обозначается огнеупорный щит, используемый в автомобилях для защиты пассажиров от огня, возникающего в случае аварии в моторном отделении. В компьютерной индустрии брандмауэром называют аппаратное устройство или

программное средство, которое защищает пользователей и данные в локальной сети от атак, исходящих из Интернета. Брандмауэр может использоваться также для предотвращения подключения пользователей локальной сети к «запрещённым» узлам Интернета. Кроме того, вы можете использовать брандмауэр для разделения внутренней сети на несколько взаимоизолированных областей.

В среде Linux можно использовать две базовых разновидности брандмауэров. Брандмауэры в среде Linux делятся на пакетные фильтры (packetfilters) и брандмауэры proxy. Каждая из этих двух базовых разновидностей обладает двумя подразновидностями. Пакетные фильтры могут быть пересылающими (forwarding), то есть принимающими решение о том, передавать ли пакет из сети в сеть или нет, и маскирующими (masquerading), то есть модифицирующими адрес-источник и адрес-приёмник. Брандмауэры proxy могут быть стандартными (standard), которые иногда называют непроницаемыми (opaque), в которых клиент подключается через специальный порт, а передаваемые данные перенаправляются в другой порт, или прозрачными (transparent), в которых клиент не использует специальный порт, а программное обеспечение брандмауэра перенаправляет пакеты из сети в сеть прозрачно для пользователей.

Функционирование брандмауэров фильтрации пакетов основано на принципе, что вся информация, которая необходима для принятия решения о том, что делать с пакетом, содержится в заголовке этого пакета. В заголовке пакета содержатся сведения об адресе-источнике, адресе-приёмнике, TTL (timetolive – время жизни пакета) и многие другие. Здесь же содержится контрольная сумма заголовка, благодаря которой можно определить, был ли заголовок повреждён или нет, а также размер полезной нагрузки. Для реализации фильтрации пакетов в OpenLinux используется программаipchains. Если вы намерены создать брандмауэр на основе фильтрации пакетов, вы должны принять решения относительно того, в отношении пакетов каких типов ваш брандмауэр будет действовать и что он должен делать с этими пакетами.

Программаipchainsпозволяет применять в отношении пакетов набор различных критериев. Эта программа может осуществлять анализ входящих пакетов, исходящих пакетов, а также пакетов, передаваемых через брандмауэр (forwardedpackets). Решения могут приниматься на основании адреса, откуда исходят пакеты, адреса, куда они направляются, а также на основании номера порта, для которого они предназначены. Различные правила могут применяться в зависимости от того, является ли пакет пакетом TCP, UDP или ICMP. Наконец, если некоторый тип пакетов не определён явно в рамках набора правил брандмауэра, в его отношении будет применяться определяемая вами политика по умолчанию. В ядрах Linux семейства 2.4.x появилось новое программное

средство netfilter, которое частично работает в режиме ядра. По сравнению с более старым программным средством ipchains программа netfilter более плотно взаимодействует с ядром, поэтому код netfilter работает быстрее.

Маскировка IP и перенаправление портов. В любой ситуации, в которой вам требуется подключить к Интернету более чем один компьютер, вы обнаруживаете, что перед вами встаёт необходимость некоторым образом спроектировать вашу сеть. Многие из тех, кто не занимался этим ранее, не знают, с чего начать и какими возможностями они могут воспользоваться. Проектируя сеть, приходится иметь дело с множеством конфликтующих, подчас противоречащих друг другу обстоятельств, при этом вы должны найти компромиссное решение, которое было бы одновременно и гибким, и в достаточной степени безопасным.

Для домашнего пользователя, у которого не возникает необходимости поддерживать внутреннюю систему доменных имён и который не собирается предлагать свои службы другим пользователям Интернета, выбор весьма прост: использовать базовый сервер IP-маскировки, которого вполне достаточно для обеспечения обмена данными с Интернетом. Маскировка IP (эту технологию так же часто называют NetworkAddressTranslation, NAT) наделяет вас существенным преимуществом: после того как вы настроите IP-адреса в вашей внутренней сети, вам больше не придётся менять адресацию ваших компьютеров. Например, когда вы меняете провайдера, вы должны сменить адреса всех компьютеров, расположенных во внешней зоне DMZ, а также внести в базу данных DNS соответствующие изменения. Однако ваша внутренняя сеть может оставаться в неизменном виде. Вторым преимуществом маскировки IP является то обстоятельство, что, если вы не используете перенаправления портов, никто не сможет проникнуть в вашу внутреннюю сеть извне, не преодолев предварительно брандмауэр. Это означает, что все ваши усилия по обеспечению безопасности можно сосредоточить всего на одном сетевом узле.

Перенаправление портов (portforwarding) – это перенаправление соединения с одного узла на другой узел. С задачей перенаправления портов хорошо справляются брандмауэры proxy. Программаipchainsне позволяет выполнять перенаправление портов. Одним из целевых действий этой программы является REDIRECT (перенаправление), однако это целевое действие предназначено только для локальных перенаправлений и не позволяет перенаправлять данные на другой сетевой узел. Если вы используетеipchainsи желаете организовать перенаправление портов, вы должны использовать модульipportfwи программуipmasqadm.

Безопасность сетевых служб и демонов. В главе 12 книги «Защита и безопасность в сетях Linux» подробно рассматриваются уязвимые службы и протоколы. После завершения установки любого комплекта поставки Linux

сразу же после самой первой загрузки системы по умолчанию в ней будет функционировать набор служб. Состав запущенных в системе служб определяется многими факторами, например, набором пакетов, выбранных вами для установки. В большинстве случаев, если вы указываете программе установки установить в системе некоторую службу, скорее всего, после загрузки системы по умолчанию эта служба будет функционировать с использованием конфигурации по умолчанию. Таково стандартное поведение большинства дистрибутивов Linux. Таким образом, если вы устанавливаете в системе некоторую службу, будьте готовы к тому, что по умолчанию она будет запущена. Вы должны позаботиться о том, чтобы эта служба была должным образом настроена. Если вы не намерены предоставлять какую-либо службу в распоряжение пользователей, удалите соответствующий пакет из системы.

К уязвимым службам и протоколам относятся FTP (порты 21 и 20), telnet (порт 23), smtp (порт 25), domain (порт 53), tftp (порт 69), finger (порт 79), www (порт 80), pop2 и pop3 (порты 109 и 110), sunrpc (порт 111), auth (порт 113), netbios (порты 137–139), imap2 и imap3 (порты 143 и 220), xdmcp (порт 177 UDP), printer (порт 515), команды «r» (rsh, rexec, rlogin – порты 512, 513, 514). Службы, связанные с портами, номера которых меньше 1024, считаются привилегированными. Это означает, что связать службу с портом, номер которого меньше 1024, можно только на уровне привилегий root. Иными словами, любая служба, использующая любой порт с номером меньшим 1024, работает от имени пользователя root. Если вы не предпримете должных мер безопасности, защита вашей системы может оказаться нарушенной.

Нарушить систему защиты через сеть можно только через доступные службы. Службы, которые не функционируют в системе, не могут быть использованы для взлома. Если вы видите, что в вашей системе работает некоторая служба и вы не можете сказать, зачем она нужна, отключите её или сделайте доступной только для локального сетевого узла localhost. Этим вы существенно снизите риск. Программа inetd (интернет-суперсервер) является метадоменом, который следит за поступлением запросов на подключение через порты, не связанные ни с одним из работающих в системе серверных приложений. Когда через один из этих портов поступает запрос на подключение, программа inetd запускает соответствующую серверную программу. Благодаря такому подходу экономятся ресурсы, так как отпадает необходимость постоянно держать в памяти большое количество редко используемых серверных приложений. Информация о том, за какими портами следитinetdи какие серверные программы следует запускать в случае поступления запроса через какой-либо из этих портов, содержится в файле /etc/inetd.conf. Отредактировав этот файл, вы сможете настроить работу inetd в соответствии со своими предпочтениями.

При работе над курсовой работой обратите внимание на:

– описание особенностей механизма аутентификации в ОС, которые требуется приводить в первой (теоретической) главе (в соответствии с темой курсовой работы).

Литература

1. Окороков, В. А. Безопасность операционных систем / В. А. Окороков. – Санкт-Петербург: Лань, 2024. – 228 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/367472 (дата обращения: 06.12.2024). – ISBN 978-5-507-48297-9. – Текст : электронный. (Главы 7, с. 196–201).

2. Киренберг, Г. А. Информационная безопасность современных операционных систем: учеб. пособие / Г. А. Киренберг. – Кемерово: КузГТУ им. Т. Ф. Горбачева, 2022. – 138 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/295736 (дата обращения: 06.12.2024). – ISBN 978-5-00137-320-9. – Текст : электронный. (Глава 3).

3. Потерпеев, Г. Ю. Безопасность операционных систем: учеб. пособие / Г. Ю. Потерпеев, В. С. Нефедов, А. А. Криулин. – Москва: РТУ МИРЭА, 2021. – 93 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/182416 (дата обращения: 06.12.2024). – ISBN 978-5-7339-1393-3. – Текст : электронный. (Главы 1, 4).

4. Зверева, О. М. Операционные системы: учеб. пособие / О. М. Зверева; науч. ред. Л. Г. Доросинский; Уральский федеральный университет им. первого Президента России Б. Н. Ельцина. – Екатеринбург: Издательство Уральского университета, 2020. – 223 с. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=699030 (дата обращения: 03.11.2024). – ISBN 978-5-7996-3146-8. – Текст : электронный. (Глава 11).

Контрольные вопросы

1. Какие порты называются привилегированными и почему?
2. Какую информацию позволяет получить утилитаnetstat -a?
3. Что такое TCP Wrappers и какие файлы используются для его настройки?
4. В чём разница между пакетным фильтром и брандмауэром proxy?
5. Какие целевые действия поддерживает программаipchains?
6. В чём отличиеnetfilterотipchains?
7. Что такое маскировка IP и какие преимущества она даёт?

8. Что такое перенаправление портов и какие программы для этого используются?

9. Какие службы относятся к категории уязвимых и почему?

10. Как отключить ненужную службу в Linux?

Тема 2.5 Обеспечение безопасности архитектуры, основных и дополнительных сервисов ОС Linux. Основные настройки сервисов безопасности ОС Linux

Перечень изучаемых вопросов:

1. Безопасность архитектуры ОС Linux.
2. Настройка основных сервисов безопасности.
3. Управление службами и демонами. Сравнение с функционалом систем защиты инфорамации.
4. Безопасность Samba.
5. Установка и запуск web-сервера Apache.
6. Использование оболочки SecureShell и сетей VPN

Методические указания к изучению

Обеспечение безопасности архитектуры и сервисов ОС Linux представляет собой комплексную задачу, которая требует понимания, как внутреннего устройства системы, так и особенностей конкретных служб, предоставляемых пользователям. В отличие от теоретических основ, рассматриваемых в предыдущих темах, данный раздел посвящён практическим аспектам настройки и сопровождения защищённой конфигурации Linux.

Безопасность архитектуры ОС Linux.Архитектура Linux, как и любой современной операционной системы, строится по многоуровневому принципу. Ядро работает в привилегированном режиме (Ring 0), а пользовательские процессы – в непривилегированном (Ring 3). Безопасность архитектуры во многом зависит от корректности работы кода ядра и механизмов изоляции процессов. Важнейшими элементами архитектурной безопасности являются: разграничение прав доступа (DAC и MAC), изоляция адресных пространств процессов, контроль целостности исполняемых файлов, а также механизмы аудита и протоколирования. В книге «Защита и безопасность в сетях Linux» подчёркивается, что безопасность – это не продукт, а процесс, и что нельзя достичь абсолютной защиты; задача администратора — сделать систему достаточно защищённой, чтобы злоумышленник предпочёл искать более лёгкую цель.

Настройка основных сервисов безопасности. Центральным элементом настройки безопасности Linux является файл /etc/inetd.conf, который определяет,

какие службы запускаются по запросу через интернет-суперсервер inetd. После завершения установки любого комплекта Linux сразу же после первой загрузки системы по умолчанию в ней будет функционировать набор служб. Состав запущенных служб определяется набором пакетов, выбранных при установке. В большинстве случаев, если вы указываете программе установки установить некоторую службу, скорее всего, после загрузки системы по умолчанию эта служба будет функционировать с использованием конфигурации по умолчанию. Таково стандартное поведение большинства дистрибутивов Linux. Таким образом, если вы устанавливаете в системе некоторую службу, будьте готовы к тому, что по умолчанию она будет запущена. Вы должны позаботиться о том, чтобы эта служба была должным образом настроена. Если вы не намерены предоставлять какую-либо службу в распоряжение пользователей, удалите соответствующий пакет из системы.

11. Нарушить систему защиты через сеть можно только через доступные службы. Службы, которые не функционируют в системе, не могут быть использованы для взлома. Если вы видите, что в вашей системе работает некоторая служба и вы не можете сказать, зачем она нужна, отключите её или сделайте доступной только для локального сетевого узла localhost. Этим вы существенно снизите риск. Программаinetd(интернет-суперсервер) является метадоменом, который следит за поступлением запросов на подключение через порты, не связанные ни с одним из работающих в системе серверных приложений. Когда через один из этих портов поступает запрос на подключение, программаinetdзапускает соответствующую серверную программу. Благодаря такому подходу экономятся ресурсы, так как отпадает необходимость постоянно держать в памяти большое количество редко используемых серверных приложений. Информация о том, за какими портами следитinetdи какие серверные программы следует запускать в случае поступления запроса через какой-либо из этих портов, содержится в файле/etc/inetd.conf. Отредактировав этот файл, вы сможете настроить работуinetdв соответствии со своими предпочтениями. Сравнение с функционалом систем защиты инфорамации.: АПКШ «Континент» или др., DLP-систем (на примере DLP SecureTower), системы защиты информации (СЗИ Dallas Lock и др.)

Управление службами и демонами. В Linux управление службами осуществляется через сценарии инициализации, расположенные в каталоге /etc/rc.d/init.d/. Каждый сценарий поддерживает как минимум команды start,stopиrestart. Символические ссылки на эти сценарии размещаются в каталогах, соответствующих уровням выполнения (runlevels). Уровни выполнения – это состояния системы, определяющие, какие службы должны быть запущены. Всего имеется семь уровней выполнения, от 0 до 6, каждый из которых соответствует некоторому режиму функционирования системы.

Уровень 0 – останов системы, уровень 1 – однопользовательский режим, уровень 6 – перезагрузка. Остальные уровни (2–5) можно настраивать свободно. С точки зрения безопасности важно понимать, что все сценарии, расположенные в каталоге/etc/rc.d/ и его подкаталогах и запускаемые в процессе инициализации системы, выполняются от имени суперпользователя. Если злоумышленнику удастся подменить сценарий инициализации или изменить уже существующий, во время инициализации ОС он сможет запустить любую программу с привилегиями суперпользователя. Поэтому правом на запись в файлы и каталоги, используемые при инициализации системы, должен обладать только суперпользователь и никто другой.

Безопасность Samba. Samba – это набор программ, которые позволяют операционным системам семейства Unix (включая Linux) обмениваться данными с узлами Microsoft. В наши дни большинство сетевых рабочих сред представляет собой смесь операционных систем. Как правило, в большинстве сетей небольших и средних компаний используют центральный сервер MicrosoftWindows NT, рабочие станции Windows, а также, возможно, другие серверные или сетевые операционные системы. Чтобы обеспечить взаимодействие с этими системами, программисты Linux были вынуждены преодолеть множество препятствий. Компания Microsoft избегает поддержки NFS (NetworkFileSystem) – сетевой файловой системы, которая является общераспространённой в мире Unix (и которую поддерживает Linux). Вместо этого в мире Microsoft предпочтение отдаётся стандарту NetBEUI, который является расширением спецификации NetBIOS, основанной на протоколе LanManager компании IBM. Все же благодаря стремительному развитию Интернета компания Microsoft была вынуждена обеспечить поддержку передачи данных NetBIOS через каналы связи TCP/IP.

Программное средство SWAT (SambaWebAdministrationTool) – это средство администрирования Samba, основанное на Web. Оно было разработано в ответ на многочисленные запросы администраторов, нуждающихся в более удобном способе управления многочисленными параметрами конфигурационного файлаsmb.conf. При использовании SWAT следует учитывать, что программа swat попытается извлечь из конфигурационного файла smb.conf любую полезную информацию, а затем перезапишет файл smb.conf в своём собственном формате. При этом будут утеряны комментарии, а также такие директивы, как include и copy. Поэтому, если у вас есть составленный вручную файл smb.conf, который вы хотели бы сохранить для последующего использования, сделайте его копию или не запускайтеswat. С точки зрения безопасности, вы должны обеспечить должную защиту файлу smb.conf (а также, очевидно, бинарному файлу swat). Файл smb.conf определяет, каким доступом обладают те, кто пользуется предлагаемыми вами службами Samba и обращается

к вашему узлу OpenLinux. Демон smbd работает на уровне привилегий root, поэтому в результате неправильной настройки файла smb.conf подключающийся к Samba удалённый клиент может получить доступ к вашей системе от лица пользователя root.

Установка и запуск web-сервера Apache.В наши дни web-сервер Apache является наиболее популярным web-сервером Интернета, и это не случайно, ведь Apache является наиболее мощным и наиболее гибко конфигурируемым web-сервером из всех существующих. Прочитав предыдущие главы, вы, должно быть, уже знаете, что за мощь, обилие возможностей и высокую гибкость приходится платить потенциальным снижением уровня защиты. Сервер Apache не является исключением. Однако, как будет показано в данной главе, на самом деле, выполнив должным образом конфигурирование, вы сможете обеспечить высокий уровень защиты вашего web-сервера, сохранив при этом все его преимущества.

В файле httpd.conf большая часть значений по умолчанию вполне приемлема. Обратите внимание на параметры MaxClientsиMaxRequestsPerChild. Первый из них ограничивает число клиентов, которым разрешается подключаться к серверу в любой момент времени. По умолчанию параметр имеет значение 150, однако вы можете увеличить или уменьшить это значение таким образом, чтобы оно соответствовало объёму памяти, установленному на компьютере. Важным параметром также является MaxRequestPerChild. По умолчанию значение этого параметра равно нулю (0), что означает, что клиенту разрешается порождать неограниченное количество дочерних запросов. Такое положение дел нельзя считать вполне безопасным, так как система оказывается беззащитной перед атаками типа DenialofService (DoS). Если вы присвоите этому параметру значение 25, этого, скорее всего, будет вполне достаточно.

Следующими записями, которые представляют для нас интерес, являются записи, в которых определяются пользователь и группа (User andGroup), от лица которых будут обслуживаться клиентские запросы. По умолчанию для этой цели используется пользователь и группа с именемnobody. В результате этого, если web-сервер запускается на уровне привилегий root (а следовательно, он связывается с портами 80 и 443), дочерние процессы, которые выполняют обслуживание поступающих от клиентов запросов, порождаются от имени пользователяnobody. Благодаря этому ограничивается масштаб повреждений, которые эти процессы могут нанести системе. Таким образом, если клиент должен обладать возможностью записи в некоторый каталог, этот каталог должен быть доступен для записи для пользователяnobody.

Использование оболочки SecureShell и сетей VPN. Программа SecureShell (SSH) предназначена для организации защищённого удалённого доступа к системе через сеть в режиме, напоминающем рабочий сеанс telnet.

Одной из наиболее сложных с точки зрения безопасности проблем, с которыми сталкиваются в наше время пользователи и сетевые администраторы, является обеспечение безопасного способа удалённого доступа к расположенным дома или в офисе системам. Сам по себе удалённый доступ к таким системам не является проблемой. В состав каждой системы Linux входят клиент и сервер telnet. Однако использование telnet означает, что имя пользователя и пароль передаются через канал связи в незашифрованном виде. Это обстоятельство не является проблемой в случае, если для передачи данных используется выделенная телефонная линия. Однако если данные передаются далее, за сервер, который обеспечивает соединение между телефонной линией и Интернетом, любой пользовательtcpdumpрили какой-либо другой утилиты, прослушивающей сеть, сможет перехватить пользовательское имя и пароль. В отличие от SSH, программный механизм FreeS/WAN осуществляет шифровку всего трафика между двумя сегментами WAN. Программа SSH (версия 1) распространяется бесплатно для некоммерческого использования, однако если вы намерены использовать версию 2, вы в любом случае должны приобрести лицензию.

При работе над курсовой работойобратите внимание на:

– описание особенностей механизмов ИБ в ОС, которые требуется приводить в первой (теоретической) главе (в соответствии с темой курсовой работы) и во второй.

Литература

1. Окороков, В. А. Безопасность операционных систем / В. А. Окороков. – Санкт-Петербург: Лань, 2024. – 228 с. – Режим доступа: для авториз. пользователей. – Лань: электронно-библиотечная система. – URL: https://e.lanbook.com/book/367472 (дата обращения: 06.12.2024). – ISBN 978-5-507-48297-9. – Текст: электронный. (Главы 7).

2. Киренберг, Г. А. Информационная безопасность современных операционных систем: учеб. пособие / Г. А. Киренберг. – Кемерово: КузГТУ им. Т. Ф. Горбачева, 2022. – 138 с. – Режим доступа: для авториз. пользователей. – Лань: электронно-библиотечная система. – URL: https://e.lanbook.com/book/295736 (дата обращения: 06.12.2024). – ISBN 978-5-00137-320-9. – Текст: электронный. (Глава 3).

3. Потерпеев, Г. Ю. Безопасность операционных систем: учеб. пособие / Г. Ю. Потерпеев, В. С. Нефедов, А. А. Криулин. – Москва: РТУ МИРЭА, 2021. – 93 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. — URL: https://e.lanbook.com/book/182416 (дата обращения: 06.12.2024). – ISBN 978-5-7339-1393-3. – Текст : электронный. (Главы 4).

4. Зверева, О. М. Операционные системы: уче. пособие / О. М. Зверева; науч. ред. Л. Г. Доросинский; Уральский федеральный университет им. первого Президента России Б. Н. Ельцина. – Екатеринбург: Издательство Уральского университета, 2020. – 223 с. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=699030 (дата обращения: 03.11.2024). – ISBN 978-5-7996-3146-8. – Текст : электронный. (Глава 11).

Контрольные вопросы:

1. Какие элементы архитектуры Linux обеспечивают безопасность?
2. Что определяет файл/etc/inetd.conf?
3. Как управлять службами через сценарии инициализации?
4. Что такое Samba и какие риски связаны с её использованием?
5. Как настраивается безопасность web-сервера Apache?
6. В чём преимущества SSH перед telnet?
7. Что такое FreeS/WAN и для чего он используется?
8. Какие параметры httpd.conf влияют на безопасность?
9. Как ограничить доступ к SWAT?
10. Какие порты используются по умолчанию для SSH и Apache?

Тема 2.6 Аудит системы безопасности ОС Linux

Перечень изучаемых вопросов:

1. Процедура аудита применительно к ОС.
2. Требования к аудиту операционной системы.
3. Политика аудита.
4. Реализация аудита в UNIX.
5. Реализация аудита в Windows.

Методические указания к изучению

Аудит операционной системы представляет собой систематический процесс сбора, анализа и интерпретации данных о событиях, происходящих в системе, с целью оценки её безопасности и выявления потенциальных инцидентов. Процедура аудита начинается с определения целей, которые могут включать мониторинг доступа к файлам, отслеживание действий пользователей, регистрацию изменений конфигурации или обнаружение попыток эксплуатации уязвимостей. На этом этапе администратор или специалист по безопасности определяет, какие события подлежат фиксации, исходя из специфики системы и уровня риска, связанного с её эксплуатацией.

Следующим шагом процедуры становится настройка механизмов аудита, встроенных в операционную систему, которые обеспечивают регистрацию событий в виде логов. Эти механизмы включают в себя системные вызовы,

службы журналирования и специализированные утилиты, которые фиксируют данные о действиях пользователей, процессов и ядра. Например, в операционной системе могут регистрироваться попытки входа в систему, запуск программ, изменения прав доступа или ошибки выполнения операций. Собранные данные сохраняются в лог-файлах, таких как системные журналы или специализированные базы данных аудита, которые затем анализируются для выявления аномалий. Анализ может проводиться вручную или с использованием автоматизированных инструментов, таких как системы управления событиями безопасности (SIEM), которые коррелируют события и выявляют паттерны, указывающие на угрозы.

Заключительный этап процедуры аудита предполагает интерпретацию результатов и принятие мер на основе полученной информации. Это может включать усиление политики безопасности, устранение уязвимостей или расследование инцидентов с привлечением дополнительных ресурсов. С точки зрения безопасности, процедура аудита уязвима к атакам, направленным на подмену или удаление логов, что требует защиты самих данных аудита.

Аудит операционной системы включает последовательность этапов, направленных на обеспечение прозрачности и контроля над системными процессами. Первым шагом является планирование, где определяются цели аудита: проверка соответствия политикам безопасности, обнаружение аномалий или подготовка к сертификации. На этом этапе выбираются ключевые объекты аудита – пользовательские действия, доступ к файлам, изменения реестра, сетевые соединения.

Далее следует сбор данных с использованием встроенных инструментов операционной системы (например, auditd в Linux или EventViewer в Windows) и сторонних решений (Splunk, Graylog). Данные записываются в логи, которые должны быть защищены от модификации с помощью механизмов целостности – хеширования, цифровых подписей.

Анализ данных предполагает поиск паттернов, указывающих на угрозы: множественные неудачные попытки входа, несанкционированный доступ к чувствительным файлам, подозрительные процессы. Для автоматизации применяются SIEM-системы (SecurityInformationandEventManagement), которые агрегируют данные и генерируют оповещения.

Завершающий этап – формирование отчёта, включающего выявленные уязвимости, рекомендации по устранению рисков и доказательства соответствия стандартам (ISO 27001, PCI DSS). Отчёт служит основой для корректировки политик безопасности и проведения профилактических мероприятий.

Требования к аудиту операционной системы формируются на основе нормативных документов, стандартов безопасности и специфики эксплуатации операционной системы. Основным требованием является полнота охвата событий, подлежащих регистрации, что подразумевает фиксацию всех значимых действий, влияющих на безопасность системы. Это включает мониторинг входов и выходов пользователей, изменений конфигурационных файлов, запуска привилегированных процессов и доступа к критическим ресурсам. Полнота

обеспечивается правильной настройкой подсистемы аудита, которая должна быть достаточно гибкой, чтобы адаптироваться к различным уровням риска и задачам организации.

Другим важным требованием выступает достоверность данных аудита, гарантирующая, что зарегистрированные события точно отражают реальные действия в системе. Это достигается за счёт защиты логов от модификации или удаления, например, через ограничение прав доступа к файлам журналов и использование механизмов цифровой подписи или хеширования для проверки их целостности.

Требование своевременности подразумевает, что события фиксируются в реальном времени или с минимальной задержкой, что позволяет оперативно реагировать на инциденты. Кроме того, данные аудита должны быть доступны для анализа в удобном формате, что требует их структурирования и хранения в читаемом виде, таком как текстовые файлы или базы данных с возможностью поиска.

С точки зрения безопасности, требования к аудиту включают обеспечение конфиденциальности логов, поскольку они могут содержать чувствительную информацию, такую как имена пользователей или параметры операций. Также важно соответствие нормативным стандартам, например, ISO 27001 или PCI DSS, которые предписывают определённые правила ведения аудита в системах, обрабатывающих финансовые или персональные данные.

Кроме того, следует выделить требование невозможности отказа (non-repudiation), которое гарантирует, что пользователь не сможет отрицать совершённые действия. Это достигается через привязку событий к уникальным идентификаторам – SID в Windows, UID в UNIX. Также важно обеспечить целостность логов, для чего используются алгоритмы хеширования, например, SHA-256.

Политика аудита представляет собой формализованный набор правил и настроек, определяющих, какие события подлежат регистрации, как они обрабатываются и хранятся в операционной системе. Она разрабатывается на основе целей безопасности организации и требований, предъявляемых к системе, и служит руководством для настройки подсистемы аудита. Политика включает определение категорий событий, таких как попытки аутентификации, изменения файлов, запуск процессов или сетевые подключения, а также указание уровня детализации – от минимального, когда регистрируются только критические события, до расширенного, когда фиксируются все действия пользователей и системы. Например, в высокорисковой среде политика может предписывать регистрацию всех операций с привилегированными учётными записями, тогда как в менее критичных системах достаточно отслеживать только неудачные попытки входа.

Кроме того, политика аудита регулирует управление логами, включая их ротацию, архивирование и сроки хранения. Это важно для предотвращения переполнения дискового пространства и обеспечения доступности данных для анализа в случае инцидента. Политика также определяет меры защиты логов от

несанкционированного доступа или подделки, например, через использование шифрования или перенаправление данных на удалённый сервер. С точки зрения безопасности, политика аудита должна быть сбалансирована: избыточная детализация может привести к снижению производительности системы, тогда как недостаточная — к упущению важных событий.

Для обеспечения надёжной защиты операционной системы в журнале аудита должны обязательно регистрироваться следующие события: попытки входа и выхода пользователей из системы, попытки изменения списка пользователей, попытки изменения политики безопасности, в том числе и политики аудита. Система контроля регистрирует события в операционной системе, связанные с защитой информации, записывая их в контрольный журнал, в котором возможна фиксация проникновения в систему и неправильного использования ресурсов.

Windows позволяет регистрировать в журнале аудита события следующих категорий: вход и выход пользователя из системы, доступ субъектов к объектам, использование субъектами доступа опасных привилегий, изменения в списке пользователей, изменения в политике безопасности, системные события, запуск и завершение процессов.

Считаются опасными следующие привилегии субъектов: получать оповещения от файловой системы, добавлять записи в журнал аудита, создавать маркеры доступа, назначать маркеры доступа процессам, создавать резервные копии информации, восстанавливать информацию на дисках с резервных копий, отлаживать программы.

Одним из наиболее сложных вопросов является вопрос о том, какой должна быть политика аудита. Политика аудита настолько сильно связана с особенностями эксплуатации конкретного экземпляра операционной системы, что сформулировать эталонную политику просто невозможно. Даже для конкретного экземпляра операционной системы нельзя сформулировать адекватную политику аудита «на все времена». Политика аудита должна постоянно меняться, реагируя на изменения в конфигурации операционной системы и на зарегистрированные опасные события.

При определении политики аудита следует иметь в виду, что адекватность политики заключается в том, что регистрируется ровно столько событий, сколько необходимо. Если подсистема аудита регистрирует слишком много событий, то, с одной стороны, журнал аудита переполняется слишком быстро, а с другой – аудитору трудно выделить в огромном объёме информации важные события.

4. Реализация аудита в UNIX

Реализация аудита в UNIX-системах, таких как Linux или FreeBSD, базируется на встроенных подсистемах, обеспечивающих гибкость и детализированный контроль событий. Одним из основных инструментов в Linux является подсистема Audit, интегрированная в ядро и управляемая через утилиту auditctl. Эта подсистема позволяет настраивать правила аудита для регистрации системных вызовов (например, open, execve), изменений файлов и действий

пользователей. Логи аудита сохраняются в файлы, такие как /var/log/audit/audit.log, и содержат подробную информацию о времени события, пользователе, процессе и результате операции. Управление правилами осуществляется через конфигурационные файлы в /etc/audit/audit.rules, где можно указать, например, мониторинг доступа к критическим файлам, таким как /etc/passwd.

Дополнительно в UNIX используются традиционные механизмы журналирования, такие как syslog, которые фиксируют системные события в файлах /var/log/messages или /var/log/syslog. Для анализа логов применяются утилиты, такие как ausearch и aureport, которые позволяют фильтровать события по заданным критериям и генерировать отчёты. С точки зрения безопасности, реализация аудита в UNIX уязвима к атакам на логи – удалению или подмене, а также к перегрузке системы при чрезмерной детализации правил.

В UNIX-системах аудит реализуется через подсистему auditd, которая настраивается файлами в каталоге /etc/audit/. Ключевыми компонентами являются: утилита auditctl для управления правилами аудита, утилита ausearch для поиска событий по фильтрам – время, ключ, тип, и утилита aureport для генерации сводных отчётов, таких как статистика по пользователям и системным вызовам.

Например, команда auditctl с параметром -w /etc/passwd -p wa -k user_changes отслеживает запись и изменение атрибутов файла /etc/passwd, помечая события ключом «user_changes». Правило для отслеживания использования sudo может выглядеть следующим образом: auditctl -a always,exit -F arch=b64 -S execve -F path=/usr/bin/sudo -k admin_actions. Это правило регистрирует все запуски sudo, сохраняя их в лог с меткой «admin_actions». Интеграция с rsyslog позволяет дублировать логи на удалённый сервер для предотвращения потери данных при компрометации локальной системы.

Реализация аудита в Windows основана на встроенной подсистеме WindowsEventLogging, которая регистрирует события в журналах, доступных через EventViewer. Аудит настраивается через локальные политики безопасности (secpol.msc) или групповые политики (GPO) в доменной среде, позволяя фиксировать такие категории событий, как вход в систему, доступ к объектам — файлам, реестру, изменения политик и запуск процессов. Логи сохраняются в журналах Security, System и Application, расположенных в формате EVT/EVTX в каталоге C:\Windows\System32\winevt\Logs. Например, журнал Security фиксирует попытки аутентификации, указывая имя пользователя, время и результат операции.

Для управления аудитом в Windows используется механизм SACL (SystemAccessControlList), который применяется к объектам – файлам, папкам, ключам реестра — и определяет, какие действия с ними подлежат регистрации. Анализ логов осуществляется через EventViewer или с помощью PowerShell-командлетов, таких как Get-EventLog или Get-WinEvent, которые позволяют фильтровать события и экспортировать их для дальнейшего изучения. С точки зрения безопасности, реализация аудита в Windows уязвима к очистке логов,

например, через команду wevtutilcl, или переполнению журнала при недостаточном размере хранилища.

В Windows аудит настраивается через групповые политики и локальные политики безопасности. Основными инструментами являются: EventViewer для просмотра логов из категорий Security, System и Application; утилита Auditpol – командная строка для управления политиками, например, команда auditpol /set /category:"ObjectAccess" /success:enable /failure:enable активирует аудит доступа к объектам; PowerShell – скрипты для автоматизации сбора логов с использованием Get-EventLog и Get-WinEvent.

Настройка аудита доступа к файлам осуществляется через свойства файла или папки, вкладку «Безопасность», кнопку «Дополнительно», вкладку «Аудит», где необходимо добавить субъекта — пользователя или группу – и выбрать типы доступа, такие как чтение или запись. В доменных средах политики аудита централизованно применяются через GPO, что упрощает управление в крупных организациях.

Сравнивая реализацию аудита в UNIX и Windows, можно выделить ряд различий. В UNIX подсистема аудита интегрирована в ядро и управляется через auditctl и файлы правил, тогда как в Windows используется служба WindowsEventLogging с настройкой через GPO и auditpol. Формат логов в UNIX преимущественно текстовый, в Windows – бинарный EVTX. Гранулярность в UNIX достигается на уровне системных вызовов, файлов и процессов, в Windows — на уровне категорий, объектов и привилегий. Защита логов в UNIX обеспечивается через chattr +a и удалённый syslog, в Windows – через WEF (WindowsEventForwarding) и SACL.

Современные тенденции в области аудита операционных систем связаны с интеграцией с SIEM-системами, такими как Splunk, ELK Stack, Graylog и Wazuh, которые обеспечивают централизованный сбор и корреляцию событий. Применение методов машинного обучения позволяет выявлять аномалии в поведении пользователей и процессов, обнаруживать零-day атаки на основе отклонений от базовой линии и снижать количество ложных срабатываний. Также важным направлением является обеспечение соответствия стандартам – регулярный аудит конфигураций по CIS Benchmarks и STIG, автоматизация проверок соответствия и документирование политик аудита.

Защита самой подсистемы аудита предполагает мониторинг состояния auditd и EventLogging, оповещение о попытках отключения или очистки логов, использование неизменяемых хранилищ (WORM – WriteOnceReadMany), регулярное резервное копирование и проверку целостности.

Освоение темы «Аудит операционной системы» требует последовательного подхода, объединяющего теоретическую базу с практическими навыками. На первом этапе студенты должны изучить процедуру аудита, анализируя её этапы – от определения целей до интерпретации логов – на примере типичных сценариев, таких как мониторинг доступа к файлам. Лабораторная работа на этом этапе может включать настройку базового аудита

в тестовой операционной системе с последующим анализом сгенерированных логов, что помогает понять процесс и его значение для безопасности.

Переходя к требованиям к аудиту, студенты должны сосредоточиться на изучении нормативных стандартов, например, ISO 27001, и их влияния на настройку системы. Анализ требований полноты, достоверности и своевременности следует дополнить рассмотрением их реализации в реальных операционных системах. Практическое задание может предусматривать разработку требований для аудита тестовой системы с последующей проверкой их выполнения, что закрепляет понимание баланса между функциональностью и производительностью.

Изучение политики аудита предполагает анализ её структуры и влияния на регистрацию событий. Студентам следует рассмотреть примеры политик для разных типов систем, изучая их настройку и влияние на объём логов. Лабораторная работа может включать создание политики аудита для тестовой среды с последующим анализом результатов через логи, что демонстрирует её роль в управлении безопасностью.

При изучении реализации аудита в UNIX студенты должны погрузиться в работу подсистемы Audit, анализируя настройку правил через auditctl и интерпретацию логов с помощью ausearch. Практическая часть может включать мониторинг изменений в системных файлах с последующим анализом результатов, что подчёркивает гибкость UNIX. Для Windows следует сосредоточиться на настройке аудита через GPO и SACL, изучая журналы через EventViewer и PowerShell. Практическое задание может предусматривать настройку аудита доступа к файлу с моделированием действий и анализом логов, что демонстрирует возможности Windows.

Для интеграции знаний рекомендуется провести итоговый семинар, где студенты представят сравнительный анализ аудита в UNIX и Windows, предложив рекомендации по улучшению безопасности на основе изученных подходов. Такой подход обеспечивает глубокое освоение материала и развитие аналитических навыков.

Аудит операционных систем представляет собой фундаментальный инструмент обеспечения информационной безопасности, позволяющий не только фиксировать события, но и выявлять угрозы, расследовать инциденты и обеспечивать соответствие нормативным требованиям. В условиях нарастающих киберугроз и ужесточения законодательства роль аудита продолжает возрастать. Эффективная подсистема аудита должна быть полной, достоверной, защищённой, интегрированной с SIEM-системами и адаптивной к изменяющейся модели угроз. Освоение данной темы формирует у обучающихся компетенции, необходимые для построения защищённых операционных сред и эффективного реагирования на инциденты информационной безопасности.

При работе над курсовой работойобратите внимание на:

– описание особенностей механизма аутентификации в ОС, которые требуется приводить в первой (теоретической) главе (в соответствии с темой курсовой работы).

Литература

1. Окороков, В. А. Безопасность операционных систем / В. А. Окороков. – Санкт-Петербург: Лань, 2024. – 228 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/367472 (дата обращения: 06.12.2024). – ISBN 978-5-507-48297-9. – Текст : электронный. (Главы 7).

2. Киренберг, Г. А. Информационная безопасность современных операционных систем: учеб. пособие / Г. А. Киренберг. – Кемерово: КузГТУ им. Т. Ф. Горбачева, 2022. – 138 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/295736 (дата обращения: 06.12.2024). – ISBN 978-5-00137-320-9. – Текст : электронный. (Глава 3).

3. Потерпеев, Г. Ю. Безопасность операционных систем: учеб. пособие / Г. Ю. Потерпеев, В. С. Нефедов, А. А. Криулин. – Москва: РТУ МИРЭА, 2021. – 93 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/182416 (дата обращения: 06.12.2024). – ISBN 978-5-7339-1393-3. – Текст : электронный. (Главы 5).

4. Зверева, О. М. Операционные системы: учеб. пособие / О. М. Зверева; науч. ред. Л. Г. Доросинский; Уральский федеральный университет им. первого Президента России Б. Н. Ельцина. – Екатеринбург: Издательство Уральского университета, 2020. – 223 с. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=699030 (дата обращения: 03.11.2024). – ISBN 978-5-7996-3146-8. – Текст : электронный. (Глава 11).

Контрольные вопросы:

1. Приведите особенности аудита в ОС Windows.
2. Приведите особенности аудита в ОС Linux.
3. Приведите особенности построения политики аудита в ОС.
4. Приведите особенности процедур аудита в ОС.
5. Какие этапы включает процедура аудита ОС и как они связаны с обнаружением инцидентов?
6. Объясните, почему аудит должен соответствовать принципу «невозможности отказа».

7. Как политика аудита влияет на выбор событий для регистрации в Windows и UNIX?

8. Опишите процесс настройки аудита доступа к файлу/etc/passwdв Linux с использованием auditctl.

9. Какие категории событий рекомендуется аудировать в Windows для обнаружения брутфорс-атак?

10. Как интеграция с SIEM-системами повышает эффективность аудита?

11. Какие риски возникают при избыточном сборе данных аудита?

12. Сравните возможности auditd (UNIX) и EventViewer (Windows) для расследования инцидентов.

3. МЕТОДИЧЕСКИЕ РЕКОМЕНДАЦИИ ПО ПОДГОТОВКЕ К ЛАБОРАТОРНЫМ ЗАНЯТИЯМ

Лабораторные занятия направлены на углубление знаний и закрепление основных понятий и методов, изучаемых в рамках дисциплины. Основная цель занятий – научиться применять теоретические знания для решения прикладных задач.

Рекомендации к подготовке

1. Изучение лекционного материала и конспектов:

– перед лабораторными занятиями необходимо проработать соответствующие разделы лекционного материала;

– рекомендуется вести конспект занятий и дополнительно пересмотреть его перед началом лабораторного занятия.

2. Проработка учебной литературы:

– используйте основные учебники и методические пособия, рекомендованные преподавателем;

– для глубокого понимания теории рекомендуется обращаться к дополнительной литературе.

3. Решение типовых задач:

– проработать примеры и типовые задачи из учебных материалов, методических указаний;

– выполните задачи, предложенные преподавателем для самостоятельной работы, что обеспечит лучшее понимание методов и подходов к решению задач.

4. Подготовка вопросов:

– составьте список вопросов по материалу, вызвавшему затруднения;

– обсуждение этих вопросов на лабораторном занятии поможет устранить пробелы в знаниях.

5. Вопросы для самоконтроля:

– перед каждым занятием рекомендуется проверять себя с помощью вопросов для самоконтроля из методических указаний к лекционным занятиям. Это позволит оценить уровень своей подготовки.

Тематический план лабораторных занятий приводится в разделе «Тематический план».

4. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ САМОСТОЯТЕЛЬНОЙ РАБОТЫ

В лекциях по предмету излагаются основные знания по курсу дисциплины. Самостоятельная работа имеет особое значение для прочного усвоения материала. Она помогает научиться правильно, ориентироваться в научной литературе, самостоятельно мыслить и находить правильные ответы на возникающие вопросы. В ходе всех видов занятий происходит углубление и закрепление знаний студентов, вырабатывается умение правильно излагать свои мысли.

Самостоятельная работа выполняет ряд функций, к которым относятся:

- развивающая (повышение культуры умственного труда, приобщение к творческим видам деятельности, обогащение интеллектуальных способностей студентов);
- информационно-обучающая (учебная деятельность студентов на аудиторных занятиях, неподкрепленная самостоятельной работой, становится малорезультативной);
- ориентирующая и стимулирующая (процессу обучения придается профессиональное ускорение);
- воспитывающая (формируются и развиваются профессиональные качества специалиста);
- исследовательская (новый уровень профессионально-творческого мышления).

В основе самостоятельной работы студентов лежат принципы: самостоятельности, развивающе-творческой направленности, целевого планирования, личностно-деятельностного подхода.

Самостоятельная работа студентов проводиться с целью:

- систематизации и закрепления полученных теоретических знаний и практических умений студентов;
- углубления и расширения теоретических знаний;
- формирования умений использовать нормативную, правовую,

справочную документацию и специальную литературу;

• развития познавательных способностей и активности студентов: творческой инициативы, самостоятельности, ответственности и организованности;

• формирования самостоятельности мышления, способностей к саморазвитию, самосовершенствованию и самореализации;

• развития исследовательских умений.

Для достижения указанной цели студенты на основе плана самостоятельной работы должны решать следующие задачи:

- изучить рекомендуемые литературные источники:
- изучить основные понятия, представленные в глоссарии;
- ответить на контрольные вопросы:
- решить предложенные задачи, кейсы, ситуации;
- выполнить контрольные и курсовые работы.

Работа студентов в основном складывается из следующих элементов:

1. Изучение и усвоение в соответствии с учебным планом программного материала по всем учебным дисциплинам:
2. Выполнение письменных контрольных и курсовых работ;
3. Подготовка и сдача зачетов, курсовых работ, итоговых экзаменов:
4. Написание и защита дипломной работы.

Самостоятельная работа включает такие формы работы, как:

• индивидуальное занятие (домашние занятия) – важный элемент в работе студента по расширению и закреплению знаний;

• конспектирование лекций;

• получение консультаций для разъяснений по вопросам изучаемой дисциплины;

• подготовка ответов на вопросы тестов;

• подготовка к экзамену;

• выполнение контрольных, курсовых проектов и дипломных работ;

• подготовка научных докладов, рефератов, эссе;

• анализ деловых ситуаций (мини кейсов) и др.

Содержание внеаудиторной самостоятельной работы определяется в соответствии с рекомендуемыми видами заданий в соответствии с рабочей программой учебной дисциплины. Распределение объема времени на внеаудиторную самостоятельную работу в режиме дня студента не регламентируется расписанием.

Виды заданий для внеаудиторной самостоятельной работы, их содержание и характер могут иметь вариативный и дифференциальный характер, учитывать

специфику специальности, изучаемой дисциплины, индивидуальные особенности студента.

Видами заданий для внеаудиторной самостоятельной работы могут быть:

Для овладения знаниями:

- чтение текста (учебника, первоисточника, дополнительной литературы);
- составление плана текста;
- конспектирование текста;
- выписки из текста;
- работа со словарями и справочниками;
- исследовательская работа;
- использование аудио- и видеозаписи;
- работа с электронными информационными ресурсами и ресурсами Internet:

Для закрепления и систематизации знании:

- работа с конспектом лекции (обработка текста);
- повторная работа над учебным материалом (учебника, первоисточника, дополнительной литературы, аудиовидеозаписей):
- составление плана и тезисов ответа;
- выполнение тестовых заданий;
- ответы на контрольные вопросы;
- аннотирование, реферирование, рецензирование текста;
- подготовка сообщений к выступлению на семинаре, конференции; подготовка рефератов, докладов;
- работа с компьютерными программами;
- подготовка к сдаче экзамена;

Для формирования умений:

- решение задач и упражнений по образцу;
- решение вариативных задач и упражнений:
- выполнение расчетно-графических работ;
- решение ситуационных производственных (профессиональных) задач;
- участие в научных и практических конференциях;
- проектирование и моделирование разных видов и компонентов профессиональной деятельности;
- создание проспектов, проектов, моделей;
- экспериментальная работа, участие в НИР;
- рефлексивный анализ профессиональных умений с использованием аудиовидеотехники и компьютерных расчетных программ, и электронных практикумов;
- подготовка курсовых проектов, работ и дипломных работ;

Правильная организация самостоятельных учебных занятий, их систематичность, целесообразное планирование рабочего времени позволяет привить студентам умения и навыки в овладении, изучении, усвоении и систематизации приобретаемых знаний в процессе обучения, обеспечивать высокий уровень успеваемости в период обучения, привить навыки повышения профессионального уровня в течение всей трудовой деятельности.

5. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ КУРСОВОЙ РАБОТЫ

Подробные указания приведены в учебно-методическом пособии по выполнению курсовых работ для студентов специальности 10.05.03 «Информационная безопасность автоматизированных систем» по дисциплине «Безопасность операционных систем»

Цели каждой отдельной курсовой работы должны раскрывать выбранную студентом тему. Курсовая работа предназначена для углубления студентами теоретических и практических навыков в области обеспечения информационной безопасности операционных систем. Современные требования к специалистам предполагают не только глубокое знание теоретических основ и принципов использования информационных технологий. Будущие специалисты должны иметь четкое представление обо всех этапах создания и эксплуатации информационных технологий, уметь осуществлять выбор из широкого арсенала современных средств и методов защиты информации в операционных системах наиболее адекватные поставленной задаче. Курсовая работа – это одна из форм учебной (творческой и научно-исследовательской) работы, ее выполнение является обязательным для всех студентов очной и заочной форм обучения. Выполнение курсовой работы представляет собой самостоятельное решение студентом под руководством преподавателя частной задачи или проведение исследования по одному из вопросов, изучаемых в цикле специальных дисциплин (по ГОС ВПО) или в дисциплинах профессионального цикла (по ФГОС ВПО). Основной целью выполнения курсовых работ (проектов) является закрепление, углубление и обобщение знаний, полученных студентом за время теоретического и практического обучения, расширение объема профессионально значимых умений и навыков. Содержание курсовых работ (проектов) должно отвечать учебным задачам дисциплины, увязываться с последующей работой выпускников по специальности /направлению подготовки.

Поэтому в цели и задачи курсовой работы входят:

1) закрепление практических навыков настройки политик безопасности операционных систем, полученных на лабораторных занятиях по дисциплине «Безопасность операционных систем»;

2) углубление теоретических и практических знаний в области методологии отладки политик безопасности операционных систем;

3) развитие навыков самостоятельного планирования задач защиты операций и ключевой информации операционных систем;

4) получение опыта сбора регистрируемых событий, и обработки регистрируемых событий в операционной системе;

5) приобретение навыков создания резервных копий операционных систем.

Выполнение курсовой работы позволяет расширить и закрепить приобретенные студентом в ходе обучения в вузе теоретические знания и продемонстрировать полученные навыки по самостоятельной постановке и решению конкретной задачи, а также продемонстрировать владение профессиональными навыкам в области защиты информации.

При выполнении курсовой работы обучающимся рекомендуется использование элементов дистанционных образовательных технологий с использованием информационных и учебно-методических ресурсов. При этом график курсовой работы должен определяться количеством часов, указанным в учебном плане.

Важнейшими требованиями при выполнении курсовой работы для студента являются ее самостоятельность и актуальность, связанная с решением вопросов по заданиям или по тематике работ промышленных, коммерческих или научно-исследовательских организаций; использованием современной программной и аппаратной базы; справочных материалов; новейших методов организации расчетов, проектирования и исследований.

Обучающийся выбирает тему курсовой работы из числа предложенных тем. При выборе темы курсовой работы (КР) необходимо учесть возможность дальнейшего ее развития, углубления и конкретизации, а также использования в курсовой работе.

Обучающийся может предложить свою тему с обоснованием целесообразности ее разработки и при согласовании с заведующим кафедрой и/или научным руководителем.

Выбранная тема курсовой работы должна быть согласована с научным руководителем. Изменения темы курсовой работы могут быть внесены только после согласования с научным руководителем.

При выборе темы курсовой работы необходимо учитывать следующие условия:

− соответствие темы курсовой работы содержанию дисциплины, по которой выполняется курсовая работа; актуальность проблемы;

− наличие специальной литературы и возможность получения фактических данных, необходимых для анализа;

− собственные научные интересы и способности обучающегося; преемственность исследований, начатых в предыдущих курсовых работах (проектах) и в период учебных практик;

− исключение по возможности дублирования (дословного совпадения формулировок) тем курсовых работ (проектов), выполняемых обучающимися (группой обучающихся).

Также при самостоятельном определении темы студенту требуется учесть свой опыт в выбранной сфере, наличие соответствующих знаний и навыков, а также имеющихся наработок по предполагаемой тематике. Это, прежде всего, относится к тем, кто долго собирал и обрабатывал материал по той или иной проблематике, участвовал в НИРС, научных конференциях, имеет публикации в научных журналах, сборниках и т. д. Научный руководитель может быть преподаватель выпускающей кафедры

Студенту следует периодически информировать научного руководителя о ходе выполнения курсовой работы, консультироваться по вызывающим затруднения или сомнения теоретическим и практическим вопросам, обязательно ставить в известность о возможных проблемах в выполнении работы и её содержания. Изменение выбранной ранее темы курсовой работы возможно при согласовании с научным руководителем.

Курсовая работа выполняется студентом в период семестра, когда по учебному плану изучается соответствующая дисциплина.

Курсовая работа представляет собой решение практической, научно-исследовательской задачи одной из актуальных проблем в области защиты операционных систем,

Объектами курсовой работы могут быть методы поиска уязвимостей операционных систем, методы анализа уязвимости операционных, способы повышения защищенности операционных систем, специфика комплектования системного обеспечения в целях повышения информационной безопасности.

При выполнении курсовой работы должно быть предусмотрено:

– обоснование актуальности и важности решаемой задачи обеспечения информационной безопасности выбранного объекта;

– анализ проблемной области защиты операционных систем;

– определение, анализ возможных путей и способов исследования и описание выбранных методов и средств решения поставленных задач;

– методы и способы решения проблем безопасности операционных систем.

При определении темы и соответственно порядка разработки курсовой работы можно придерживаться следующего плана:

– точная формулировка темы, целей и задач выполнения курсовой работы;

– изучение специфики проблемной области;

– выявление уже существующих решений и определение их эффективности

– обоснование предложений по решению проблем в области информационной защиты операционных систем;

– реализация предложенных средств и методов защиты, исследования меры защищенности операционных систем и их компонентов;

– проверка работоспособности предложенных мер защиты.

Курсовая работа предусматривает **следующие этапы:**

1. **Подготовка**к выполнению курсовой работы заключается в изучении литературы по выбранной проблеме, сборе исходных данных по рассматриваемым проблемам. На этом этапе изучаются цели функционирования и развития объекта, его обеспеченность средствами защиты, каналы уязвимости. Студент собирает, обобщает и систематизирует материалы, необходимые для разработки предложений Полученные материалы используются во введении и аналитической части работы.

2. Разработка темы. На основе собранных и обобщенных материалов, формулируются способы решения задач и разрабатываются алгоритмы решения задач, определяется специфика и порядок их реализации, реализуются предложенные решения, обосновывается эффективность разработки, исследований, решений.

3. Этапвключает оформление курсовой работы. При этом выполняется:

– систематизация и обработка материалов курсовой работы;

– отбор материала для оформления содержательной части работы и составление структуры ее изложения, подготовка необходимого иллюстративного материала и т. д.;

– определение направлений и основного содержания предложений, выявление необходимости дополнительного сбора материалов; формирование чернового варианта разработки в целом;

– сбор дополнительных материалов, детальная разработка и обоснование выдвинутых предложений;

– уточнение аналитической и исследовательской части работы;

– редактирование и окончательное оформление отобранного материала;

– оформление иллюстративного материала.

4. Заключительным этапом подготовки курсовой работы к защите является предъявление ее преподавателю ИБ. К этому моменту курсовая работа должна быть подписана студентом.

Список типовых (примерных) тем

1. Современные методы беспарольной аутентификации в ОС (FIDO2/WebAuthn, Passkeys) и их интеграция в Windows, macOS и Linux.

2. Анализ механизмов изоляции привилегий в macOS: SystemIntegrityProtection (SIP) и современные векторы атак.

3. Управление учетными записями и привилегиями в гетерогенных средах: сравнительный анализ Linux, Windows и FreeBSD с учетом требований ZeroTrust.

4. Комплексная защита ядра Linux в эпоху Spectre: современные методы смягчения атак по сторонним каналам.

5. Аппаратно-аппаратная безопасность Windows: анализ Virtualization-BasedSecurity (VBS), CredentialGuard и HVCI, их уязвимости и обходы.

6. Безопасность ядра UNIX: современные подходы к изоляции памяти с использованием MemoryTaggingExtensions (MTE) и CHERI.

7. Аудит и мониторинг событий безопасности в ОС: интеграция с SIEM-системами и применение методов машинного обучения.

8. Сравнительный анализ функций безопасности современных файловых систем: ZFS, Btrfs и ext4.

9. Реализация функций информационной безопасности в файловых системах Windows: от NTFS к ReFS.

10. Шифрованные файловые системы и полное шифрование диска (FDE) в современных ОС: анализ эффективности и производительности.

11. Безопасность контейнеров в ОС Linux: комплексное применение Seccomp, AppArmor/SELinux и пространств имен (namespaces).

12. Роль модулей безопасности (LSM) в защите ОС Linux: сравнительный анализ SELinux и AppArmor в контексте современных угроз.

13. Доверенная загрузка (SecureBoot) и измеряемая загрузка (MeasuredBoot) с использованием TPM 2.0: современные вызовы и уязвимости.

14. Управление обновлениями и пакетами в Linux и FreeBSD: безопасность репозиториев и целостность пакетов.

15. Сборка ядра ОС Linux и FreeBSD с учетом требований безопасности: минимизация поверхности атаки и hardening.

16. Безопасность AI-Native операционных систем: новые угрозы и архитектурные решения.

17. Интеграция аппаратных модулей безопасности (TPM, SecureEnclave) в ОС для защиты конфиденциальных данных.

18. Применение методов машинного обучения для обнаружения аномалий в поведении операционной системы.

19. Безопасность операционных систем для интернета вещей (IoT): особенности, угрозы и методы защиты.

20. Анализ безопасности протокола Kerberos в современных гетерогенных средах.

21. Безопасность операционных систем с поддержкой Rust-компонентов в ядре: анализ перехода Linux и Windows на memory-safe языки.

22. Атаки на цепочку поставки обновлений ОС: анализ компрометации репозиториев и механизмов доставки патчей.

23. Безопасность операционных систем для конфиденциальных вычислений (ConfidentialComputing): изоляция через SGX, SEV и TDX.

24. Угрозы безопасности операционных систем при использовании периферийных устройств: атаки через DMA, BadUSB и уязвимые драйверы.

25. Безопасность гипервизоров и виртуальных машин: анализ уязвимостей Type-1 и Type-2 гипервизоров.

26. Операционные системы для автомобилей и киберфизических систем: специфика угроз и архитектура защиты.

27. Безопасность мобильных операционных систем: сравнительный анализ Android и iOS в контексте современных атак.

28. Управление уязвимостями в операционных системах: от обнаружения до патчинга в корпоративной среде.

29. Безопасность операционных систем в облачных средах: защита гостевых ОС и гипервизоров.

30. Аппаратно-программные механизмы защиты от эксплойтов: анализ CFI, CET и ShadowStack в современных ОС.

31. Безопасность системы управления пакетами в Linux: анализ атак на цепочку поставок и методы защиты.

32. Операционные системы реального времени (RTOS) и их безопасность в критических приложениях.

33. Безопасность операционных систем на базе микроядер: анализ seL4 и других верифицированных ядер.

34. Интеграция искусственного интеллекта в подсистемы безопасности ОС: возможности и риски.

35. Безопасность журналирования и аудита в современных ОС: защита логов от подделки и компрометации.

36. Операционные системы для периферийных вычислений (Edge OS): особенности безопасности.

37. Безопасность операционных систем в условиях квантовых угроз: постквантовая криптография в ОС.

38. Анализ безопасности протоколов удаленного доступа к ОС: RDP, SSH, VNC и их современные уязвимости.

39. Безопасность операционных систем для встроенных устройств и IoT: от FreeRTOS до Zephyr.

40. Управление конфигурациями безопасности ОС: автоматизация hardening и соответствие стандартам (CIS, STIG).

6. ТРЕБОВАНИЯ К АТТЕСТАЦИИ ПО ДИСЦИПЛИНЕ

Текущая аттестация

В ходе изучения дисциплины студентам предстоит пройти следующие этапы текущей аттестации:

– контроль самостоятельной работы по темам дисциплины (устный опрос, проверка домашних заданий, тестовые задания);

– контроль выполнения лабораторных работ (защита лабораторных работ, отчёт по лабораторной работе).

Преподаватель вправе выбрать методику оценивания знаний студентов: традиционная зачетно-экзаменационная, либо балльно-рейтинговая.

Зачет может проводиться как в традиционной форме, так и в виде экзаменационного тестирования. Тестовые задания для проведения экзаменационного тестирования приведены в фонде оценочных средств по дисциплине.

Выбрана традиционная зачетно-экзаменационная методика оценивания знаний

Предусматривается: зачет, экзамен, курсовая работа.

Преподаватель вправе выбрать методику оценивания знаний студентов: традиционная зачетно-экзаменационная либо балльно-рейтинговая**.** Выбрана традиционная зачетно-экзаменационная методика оценивания знаний. Предусматриваются: зачет, экзамен, курсовая работа.

В ходе изучения дисциплины студентам предстоит пройти следующие этапы текущей аттестации: контроль самостоятельной работы по темам дисциплины, контроль выполнения лабораторных работ.

К оценочным средствам текущего контроля успеваемости относятся:

– тестовые задания открытого и закрытого типов.

Промежуточная аттестация в форме зачета проходит по результатам прохождения всех видов текущего контроля успеваемости. В отдельных случаях (при не прохождении всех видов текущего контроля) зачет может быть проведен в виде тестирования.

Экзамен может проводиться как в традиционной форме, так и в виде экзаменационного тестирования. Тестовые задания для проведения экзаменационного тестирования приведены в фонде оценочных средств по дисциплине.

Критерии оценки результатов освоения дисциплины

Универсальная система оценивания результатов обучения включает в себя системы оценок: 1) «отлично», «хорошо», «удовлетворительно», «неудовлетворительно»; 2) «зачтено», «не зачтено»; 3) 100-балльную/процентную систему и правило перевода оценок в пятибалльную систему (таблица).

Таблица – Система оценок и критерии выставления оценки

Система оценок / Критерий				
	0–40 %	**41–60 %**	**61–80 %**	**81–100 %**
	«неудовлетворительно»	**«удовлетворительно»**	**«хорошо»**	**«отлично»**
	«не зачтено»	**«зачтено»**		
1 Системность и полнота знаний в отношении изучаемых объектов	Обладает частичными и разрозненными знаниями, которые не может научно-корректно связывать между собой (только некоторые из которых может связывать между собой)	Обладает минимальным набором знаний, необходимым для системного взгляда на изучаемый объект	Обладает набором знаний, достаточным для системного взгляда на изучаемый объект	Обладает полнотой знаний и системным взглядом на изучаемый объект
Работа с информацией	Не в состоянии находить необходимую информацию, либо в состоянии находить отдельные фрагменты информации в рамках поставленной задачи	Может найти необходимую информацию в рамках поставленной задачи	Может найти, интерпретировать и систематизировать необходимую информацию в рамках поставленной задачи	Может найти, систематизировать необходимую информацию, а также выявить новые, дополнительные источники информации в рамках поставленной задачи
3 Научное осмысление изучаемого явления, процесса, объекта	Не может делать научно-корректных выводов из имеющихся у него сведений, в состоянии проанализировать только некоторые из имеющихся у него сведений	В состоянии осуществлять научно-корректный анализ предоставленной информации	В состоянии осуществлять систематический и научно- корректный анализ предоставленной информации, вовлекает в исследование новые релевантные задаче данные	В состоянии осуществлять систематический и научно-корректный анализ предоставленной информации, вовлекает в исследование новые релевантные поставленной задаче данные, предлагает новые ракурсы поставленной задачи

Система оценок / Критерий				
	0–40 %	**41–60 %**	**61–80 %**	**81–100 %**
	«неудовлетворительно»	**«удовлетворительно»**	**«хорошо»**	**«отлично»**
	«не зачтено»	**«зачтено»**		
4 Освоение стандартных алгоритмов решения профессиональных задач	В состоянии решать только фрагменты поставленной задачи в соответствии с заданным алгоритмом, не освоил предложенный алгоритм, допускает ошибки	В состоянии решать поставленные задачи в соответствии с заданным алгоритмом	В состоянии решать поставленные задачи в соответствии с заданным алгоритмом, понимает основы предложенного алгоритма	Не только владеет алгоритмом и понимает его основы, но и предлагает новые решения в рамках поставленной задачи

Оценивание тестовых заданий закрытого типа осуществляется по системе зачтено/ не зачтено («зачтено» – 41–100 % правильных ответов; «не зачтено» – менее 40 % правильных ответов) или пятибалльной системе (оценка «неудовлетворительно» – менее 40 % правильных ответов; оценка «удовлетворительно» – от 41 до 60 % правильных ответов; оценка «хорошо» – от 61 до 80 % правильных ответов; оценка «отлично» – от 81 до 100 % правильных ответов).

Тестовые задания открытого типа оцениваются по системе «зачтено/ не зачтено». Оценивается верность ответа по существу вопроса, при этом не учитывается порядок слов в словосочетании, верность окончаний, падежи.

Завершающим этапом изучения дисциплины является итоговая аттестация, представляющая собой экзамен.

Допуск к итоговой аттестации возможен при:

– наличии всех выполненных, сданных (проверенных, защищенных) лабораторных работах, наличии отчётов по ним;

– наличии показателей приемлемого уровня освоения материалов курса: более 50 % посещений от общего числа требуемых по учебному плану.

Примерные вопросы к зачету/экзамену по дисциплине

Вопросы к зачету

1. Пользовательский интерфейс ОС. Классификация программных средств
2. Основные функции ОС. Классификация ОС.
3. Концепция процесса. Типология процессов
4. Концепция ресурсов. Концепция виртуальности.
5. Концепция прерывания. Классы прерываний.

6. Классификация операционных систем. Состав ядра ОС.
7. Модули ядра. Перечислить вспомогательные модули ОС и режимы.
8. Многослойная структура ОС. Микроядерная архитектура.
9. Управление процессами ОС. Понятия задание, задача, поток, нить и процесс.
10. Контекст процесса. Особенности работы нити процесса.
11. Планирование процессов. Концепции планирования процессов. Понятие кванта.
12. Способы организации процесса. Особенности организации процесса. Проблемы выполнения процессов на процессоре.
13. Понятие прерывания. Типы прерывания. Последовательность при обработке прерываний. Способы выполнения прерываний.
14. Особенности управления памятью в ОС.
15. Особенности работы виртуальной памяти и swapping. Алгоритмы распределения памяти. Алгоритмы управления памятью.
16. Механизмы распределения адресов в ОС. Распределение при реальных и виртуальных адресациях.
17. Файловые системы. Общая организация ФС.
18. Особенности ФС FAT и exFAT.
19. Особенности ФС NTFS.
20. Особенности файловых систем ext.
21. Сравнительный анализ файловых систем.

Вопросы к экзамену

1. Пользовательский интерфейс ОС. Классификация программных средств
2. Основные функции ОС. Классификация ОС.
3. Концепция процесса. Типология процессов
4. Концепция ресурсов. Концепция виртуальности.
5. Концепция прерывания. Классы прерываний.
6. Классификация операционных систем. Состав ядра ОС.
7. Модули ядра. Перечислить вспомогательные модули ОС и режимы.
8. Многослойная структура ОС. Микроядерная архитектура.
9. Управление процессами ОС. Понятия задание, задача, поток, нить и процесс.
10. Контекст процесса. Особенности работы нити процесса.
11. Планирование процессов. Концепции планирования процессов. Понятие кванта.
12. Способы организации процесса. Особенности организации процесса. Проблемы выполнения процессов на процессоре.

13. Понятие прерывания. Типы прерывания. Последовательность при обработке прерываний. Способы выполнения прерываний.

14. Особенности управления памятью в ОС.

15. Особенности работы виртуальной памяти и swapping. Алгоритмы распределения памяти. Алгоритмы управления памятью.

16. Механизмы распределения адресов в ОС. Распределение при реальных и виртуальных адресациях.

17. Файловые системы. Общая организация ФС.

18. Особенности ФС FAT и exFAT.

19. Особенности ФС NTFS.

20. Особенности файловых систем ext.

21. Сравнительный анализ файловых систем.

22. Особенности реализации функции безопасности в ОС. Краткие различия организации безопасности Unix и Windows.

23. Организация безопасности в Unix-системах.

24. Аутентификация в Unix-системах.

25. Аспекты механизмов безопасности Windows (ACE, ACL, SACL, DACL).

26. Привилегии субъектов в Windows. Маркеры доступа. Дескриптор защиты.

27. Авторизация в ОС Windows

28. Дополнительные модули безопасности ОС.

29. Аудит в ОС

30. Способы создания и управления процессами в ОС Linux.

31. Основные команды MS-DOS, ОС Linux. Особенности создания Bat-файлов, shell-сценариев.

ЗАКЛЮЧЕНИЕ

Правильная организация учебных занятий, их систематичность, целесообразное планирование рабочего времени позволяет привить студентам умения и навыки в овладении, изучении, усвоении и систематизации приобретаемых знаний в процессе обучения, обеспечивать высокий уровень успеваемости в период обучения, привить навыки повышения профессионального уровня в течение всей трудовой деятельности.

ЛИТЕРАТУРА

Основные источники

1. Окороков, В. А. Безопасность операционных систем / В. А. Окороков. – Санкт-Петербург: Лань, 2024. – 228 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/367472 (дата обращения: 06.12.2024). – ISBN 978-5-507-48297-9. – Текст : электронный.

2. Киренберг, Г. А. Информационная безопасность современных операционных систем: учеб. пособие / Г. А. Киренберг. – Кемерово: КузГТУ им. Т. Ф. Горбачева, 2022. – 138 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/295736 (дата обращения: 06.12.2024). – ISBN 978-5-00137-320-9. – Текст : электронный.

3. Потерпеев, Г. Ю. Безопасность операционных систем: учеб. пособие / Г. Ю. Потерпеев, В. С. Нефедов, А. А. Криулин. – Москва: РТУ МИРЭА, 2021. – 93 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/182416 (дата обращения: 06.12.2024). – ISBN 978-5-7339-1393-3. – Текст : электронный.

4. Зверева, О. М. Операционные системы: учеб. пособие / О. М. Зверева; науч. ред. Л. Г. Доросинский; Уральский федеральный университет им. первого Президента России Б. Н. Ельцина. – Екатеринбург: Издательство Уральского университета, 2020. – 223 с. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=699030 (дата обращения: 03.11.2024). – ISBN 978-5-7996-3146-8. – Текст : электронный.

5. Кобылянский, В. Г. Операционные системы, среды и оболочки: учеб. пособие для вузов / В. Г. Кобылянский. – 3-е изд., стер. – Санкт-Петербург: Лань, 2022. – 120 с. – Режим доступа: для авториз. пользователей. – Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/254651 (дата обращения: 09.10.2024). — ISBN 978-5-507-44969-9. – Текст : электронный.

Дополнительная литература

1. Национальная безопасность: учебник / В. И. Абрамов, М. А. Газимагомедов, К. К. Гасанов [и др.]; под ред. К. К. Гасанова, Н. Д. Эриашвили, О. А. Мироновой. – 3-е изд., перераб. и доп. – Москва : Юнити-Дана, 2023. – 288 с. – Режим доступа: по подписке. – URL: https://biblioclub.ru/index.php?page=book&id=700171 (дата обращения: 05.06.2024). – ISBN 978-5-238-03639-7. – Текст : электронный.

2. Власенко, А. Ю. Операционные системы: учеб. пособие / А. Ю. Власенко, С. Н. Карабцев, Т. С. Рейн. – Кемерово: Кемеровский государственный университет, 2019. – 161 с. – Режим доступа: по подписке. – URL:

https://biblioclub.ru/index.php?page=book&id=574269 (дата обращения: 03.11.2024). – ISBN 978-5-8353-2424-8. – Текст : электронный.

3. Потерпеев, Г. Ю. Сборник практических занятий для дисциплины безопасность операционных систем: Практикум: учеб. пособие / Г. Ю. Потерпеев, О. В. Трубиенко, Д. П. Абрамов. – Москва: РТУ МИРЭА, 2023. – Часть 1. – 2023. – 65 с. – Режим доступа: для авториз. пользователей. — Лань : электронно-библиотечная система. – URL: https://e.lanbook.com/book/368750 (дата обращения: 06.12.2024). – ISBN 978-5-7339-1803-7. – Текст : электронный.

4. Программно-аппаратные средства обеспечения информационной безопасности: лаб. практикум для студентов специальности 10.05.03 «Информ. безопасность автоматизир. систем» / Федер. агентство по рыболовству, Калининингр. гос. техн. ун-т, Балт. гос. акад. рыбопромыслового флота; сост.: А. Г. Жестовский, В. В. Подтопельный. – 2-е изд., перераб. и доп. – Калининград: БГАРФ, 2019. – Режим доступа: для авториз. пользователей. – URL: https://lib.klgtu.ru/web/index.php (дата обращения: 05.11.2024). – ISBN 978-5-238-03639-7. – Текст : электронный. Ч. 1: Защита компьютерной информации и компьютерных систем от вредоносных программ.

5. Жестовский, А. Г. Программно-аппаратные средства обеспечения информационной безопасности: учеб. пособие для студентов высш. учеб. заведений, обучающихся по направлению «Информ. безопасность», по прогр. подгот. бакалавров, магистров, специалистов / А. Г. Жестовский, В. В. Подтопельный; Федер. агентство по рыболовству [и др.]. – Калининград: БГАРФ, 2018. – Текст : непосредственный. Ч. 2: Настройка систем защиты информации от несанкционированного доступа. – 2018. – 100 с.

6. Подтопельный, В. В. Программно-аппаратные средства обеспечения информационной безопасности: учеб. пособие для студентов специальности 10.05.03 «Информ. безопасность автоматизир. систем» всех форм обучения / В. В. Подтопельный. – Калининград: БГАРФ, 2020. – Текст : непосредственный. Ч. 3: Поиск и извлечение вредоносных программ в программной среде. – 2020. – 99 с.

7. Подтопельный, В. В. Программно-аппаратные средства обеспечения информационной безопасности: учеб. пособие для студентов специальности 10.05.03 «Информ. безопасность автоматизир. систем» всех форм обучения / В. В. Подтопельный; Федер. агентство по рыболовству [и др.]. – Калининград: БГАРФ, 2020. – Текст : непосредственный. Ч. 4: Настройка подсистем СЗИ. – 2021. – 97 с.

8. Аверченков, В. И. Аудит информационной безопасности: учеб. пособие / В. И. Аверченков. – 4-е изд., стер. – Москва: ФЛИНТА, 2021. – 269 с. – Режим доступа: по подписке. – URL:

https://biblioclub.ru/index.php?page=book&id=93245 (дата обращения: 05.11.2024). – ISBN 978-5-9765-1256-6. – Текст : электронный.

Учебно-методические пособия, нормативная литература

1. Подтопельный, В. В. Безопасность операционных систем: учебно-методическое пособие по изучению дисциплины для студентов специальности 10.05.03 «Информационная безопасность автоматизированных систем», специализация «Безопасность открытых информационных систем». – Калининград: Изд-во ФГБОУ ВО «КГТУ», 2022. – 51 с. – URL: https://www.klgtu.ru/vikon/sveden/files/vih/UMP_Bezopasnosty_operacionnyx_sistem(1).pdf (дата обращения: 08.12.2024). – Текст : электронный.

2. Подтопельный, В. В. Безопасность операционных систем: учебно-методическое пособие по выполнению лабораторных работ по дисциплине для студентов специальности 10.05.03 «Информационная безопасность автоматизированных систем» специализация «Безопасность открытых информационных систем». – Калининград: Изд-во ФГБОУ ВО «КГТУ», 2022. – 181 с. – URL: https://www.klgtu.ru/vikon/sveden/files/aij/UMP_Bezopasnosty_operacionnyx_sistem_(laboratornye_raboty)(1).pdf (дата обращения: 08.12.2024). – Текст : электронный.

3. Безопасность операционных систем: метод. указания по выполнению курсовых работ для студентов специальности 10.05.03 «Информ. безопасность автоматизир. систем» очной формы обучения / Федер. агентство по рыболовству [и др.]; авт.-сост. В. В. Подтопельный. – Калининград: БГАРФ, 2023. – 53 с. – Текст : непосредственный.

4. «Доктрина информационной безопасности Российской Федерации» (утв. Указом Президентом РФ 05.12.2016 № 646 (в действующей редакции). – Режим доступа: для авториз. пользователей из справ.-правовой системы КонсультантПлюс. – Текст: электронный.

5. «Конституция Российской Федерации» (принята всенародным голосованием 12.12.1993 с изменениями, одобренными в ходе общероссийского голосования 01.07.2020) (в действующей редакции). – Режим доступа: для авториз. пользователей из справ.-правовой системы КонсультантПлюс. – Текст: электронный.

6. Федеральный закон от 28.12.2010 N 390-ФЗ «О безопасности» (в действующей редакции). – Режим доступа: для авториз. пользователей из справ.-правовой системы КонсультантПлюс. – Текст: электронный.

7. Федеральный закон от 27.07.2006 N 149-ФЗ «Об информации, информационных технологиях и о защите информации» (в действующей редакции). – Режим доступа: для авториз. пользователей из справ.-правовой системы КонсультантПлюс. – Текст: электронный.

8. Федеральный закон от 27.07.2006 N 152-ФЗ «О персональных данных» (в действующей редакции). – Режим доступа: для авториз. пользователей из справ.-правовой системы КонсультантПлюс. – Текст: электронный.

9. Закон РФ от 21.07.1993 N 5485-1 «О государственной тайне» (в действующей редакции). – Режим доступа: для авториз. пользователей из справ.-правовой системы КонсультантПлюс. – Текст: электронный.

10. Указ Президента РФ от 06.03.1997 N 188 «Об утверждении Перечня сведений конфиденциального характера» (в действующей редакции). – Режим доступа: для авториз. пользователей из справ.-правовой системы КонсультантПлюс. – Текст: электронный.

11. «ГОСТ Р 50739-95. Средства вычислительной техники. Защита от несанкционированного доступа к информации. Общие технические требования» (принят и введен в действие Постановлением Госстандарта РФ от 09.02.1995 N 49) (в действующей редакции). – Режим доступа: для авториз. пользователей из справ.-правовой системы КонсультантПлюс. – Текст: электронный.

12. «ГОСТ Р 50922-2006. Национальный стандарт Российской Федерации. Защита информации. Основные термины и определения» (утв. и введен в действие Приказом Ростехрегулирования от 27.12.2006 N 373-ст) (в действующей редакции). – Режим доступа: для авториз. пользователей из справ.-правовой системы КонсультантПлюс. – Текст: электронный.

13. «Руководящий документ. Автоматизированные системы. Защита от несанкционированного доступа к информации. Классификация автоматизированных систем и требования по защите информации» (утв. Решением Гостехкомиссии России от 30.03.1992) (в действующей редакции). – Режим доступа: для авториз. пользователей из справ.-правовой системы КонсультантПлюс. – Текст: электронный.

14. «Руководящий документ. Средства вычислительной техники. Защита от несанкционированного доступа к информации. Показатели защищенности от несанкционированного доступа к информации» (утв. Решением Гостехкомиссии России 30.03.1992) (в действующей редакции). – Режим доступа: для авториз. пользователей из справ.-правовой системы КонсультантПлюс. – Текст: электронный.

Локальный электронный методический материал

Подтопельный Владислав Владимирович

БЕЗОПАСНОСТЬ ОПЕРАЦИОННЫХ СИСТЕМ

Редактор С. Кондрашова
Корректор Т. Звада

Уч.-изд. л. 6,7. Печ. л. 5,8.

Издательство федерального государственного бюджетного
образовательного учреждения высшего образования
«Калининградский государственный технический университет».
236022, Калининград, Советский проспект, 1