Айгуль Кайрулаевна Шайханова* – профессор кафедры информационной безопасности; Евразийский национальный университет имени Л.Н. Гумилева; Республика Казахстан; e-mail: shaikhanova_ak@enu.kz. ORCID: https://orcid.org/0000-0001-6006-4813.

Авторлар туралы мәліметтер

Әділ Ерболұлы Найманов – 1 курс магистранты; Ақпараттық жүйелер мамандығы; Қ. Құлажанов атындағы Қазақ технология және бизнес университеті; Қазақстан Республикасы; e-mail: Adil7473@gmail.com. ORCID: https://orcid.org/ 0009-0000-3935-0866.

Айгуль Кайрулақызы Шайханова* – ақпараттық қауіпсіздік кафедрасының профессор; Л.Н. Гумилёв атындағы Еуразия ұлттық университеті; Қазақстан Республикасы; e-mail: shaikhanova_ak@enu.kz. ORCID: https://orcid.org/0000-0001-6006-4813.

Information about the authors

Naimanov Adil Erboluly – 1st-year master's student; specialty Information Systems; Kazakh University of Technology and Business named after K. Kuanyshbaev; Republic of Kazakhstan; e-mail: Adil7473@gmail.com. ORCID: https://orcid.org/0009-0000-3935-0866.

Aigul Kairulaevna Shaikhanova* – professor of the department of Information Security; Eurasian National University named after L.N. Gumilyov; Republic of Kazakhstan; e-mail: shaikhanova_ak@enu.kz. ORCID: https://orcid.org/0000-0001-6006-4813.

Поступила в редакцию 20.11.2024
Поступила после доработки 25.11.2024
Принята к публикации 26.11.2024

https://doi.org/10.53360/2788-7995-2024-4(16)-16

МРНТИ: 81.93.29.

CC BY 4.0
Check for updates

И.Ж. Мейрамов
Евразийский национальный университет имени Л.Н. Гумилёва,
010000, Республика Казахстан,г. Астана, Пушкина 11
*e-mail: gentelmen3332@mail.ru

РОЛЬ ЗАЩИТНЫХ МЕХАНИЗМОВ ЯДРА В ПРЕДОТВРАЩЕНИИ АТАК НА УРОВНЕ ОПЕРАЦИОННЫХ СИСТЕМ LINUX

Аннотация: *В современных операционных системах Linux безопасность является критически важным аспектом, и ядро играет центральную роль в ее обеспечении. Ядро выступает посредником между аппаратными ресурсами и прикладным программным обеспечением, контролируя доступ к системным ресурсам и управляя выполнением процессов. Одной из ключевых функций ядра является защита системы от разнообразных киберугроз и атак, нацеленных на эксплуатацию уязвимостей на уровне операционной системы.В данной работе рассматриваются основные защитные механизмы, реализованные в ядре Linux, такие как контроль доступа (SELinux, AppArmor), рандомизация адресного пространства (ASLR), защита памяти (DEP, Stack Guard) и ограничения привилегий. Обсуждаются способы, которыми эти механизмы предотвращают или ограничивают воздействие атак, включая переполнение буфера, внедрение вредоносного кода и эскалацию привилегий. Анализируется эффективность этих методов и их роль в общей стратегии обеспечения кибербезопасности систем на базе Linux.*

Кроме того, в работе освещаются текущие тенденции и перспективы развития защитных механизмов ядра, включая интеграцию аппаратных средств безопасности и применение технологий машинного обучения для обнаружения угроз. Подчеркивается важность своевременного обновления ядра и системных компонентов, а также активная роль пользователей и администраторов в поддержании высокого уровня безопасности. Таким образом, статья предоставляет комплексный обзор того, как ядро Linux способствует предотвращению атак на уровне операционной системы и какие меры могут быть предприняты для усиления защиты в будущем.

Ключевые слова: *ядро Linux, безопасность системы, защитные механизмы, контроль доступа, ограничения прав, атаки, переполнение буфера, внедрение кода.*

Введение

Современный мир все больше зависит от информационных технологий. Корпорации и государственные органы хранят огромные объемы конфиденциальных данных, проводят важные финансовые операции и принимают критически важные решения в цифровом пространстве. Это делает их привлекательной мишенью для киберпреступников. Киберугрозы становятся все более сложными из-за увеличения числа устройств, усложнения атак, повышения квалификации злоумышленников и экономической мотивации.

Операционная система – это фундамент, на котором строится вся информационная инфраструктура организации. Компрометация операционной системы может привести к серьезным последствиям: потере данных, финансовым потерям, репутационным рискам и сбоям в работе критически важных систем. Linux, благодаря своей надежности, безопасности и открытости, широко используется в корпоративном и государственном секторах. Однако, даже такие системы не застрахованы от атак. Открытость кода, с одной стороны, позволяет сообществу находить и исправлять уязвимости, но с другой – делает систему более прозрачной для злоумышленников. Защита операционных систем, особенно в корпоративном и государственном секторах, является одной из наиболее актуальных задач информационной безопасности. Постоянно развивающиеся киберугрозы требуют постоянного совершенствования защитных механизмов. Использование многоуровневой защиты, включая аппаратные и программные средства, а также обучение персонала, является ключевым фактором обеспечения безопасности информационных систем.

В следующих разделах мы рассмотрим более подробно защитные механизмы ядра Linux и то, как они помогают предотвратить различные типы атак.

Методы исследования.

В ходе исследования роли защитных механизмов ядра в предотвращении атак на уровне операционных систем Linux был применён комплексный методологический подход. Сначала проведён детальный анализ литературы и документов, включая научные публикации, техническую документацию ядра Linux и отчёты о безопасности, что позволило собрать актуальную информацию о существующих механизмах защиты, таких как SELinux, AppArmor, ASLR и других, а также понять их теоретические основы и практическое применение. Далее выполнен сравнительный анализ различных защитных механизмов ядра. Оценивалась их эффективность, влияние на производительность системы и сложность настройки. Это помогло выявить сильные и слабые стороны каждого механизма и определить наиболее подходящие для разных сценариев использования.

Для понимания базовых этапов исследования, разберём определения механизмов значения.

SELinux (Security-Enhanced Linux) – это система управления доступом на основе политик безопасности, встроенная в ядро Linux. Она предоставляет механизм для ограничения доступа к ресурсам на уровне ядра, используя контекстную модель управления доступом (MAC – Mandatory Access Control). В отличие от традиционной системы управления доступом (DAC – Discretionary Access Control), где контроль принадлежит пользователям, SELinux позволяет администраторам задавать строгие политики, определяющие, какие процессы и пользователи могут взаимодействовать с конкретными файлами, сетевыми портами и системными объектами.

AppArmor (Application Armor) – это система безопасности на основе профилей, которая ограничивает действия приложений в операционных системах на основе Linux. Она использует механизм управления доступом (MAC – Mandatory Access Control) для применения наборов разрешений (профилей), описывающих, какие ресурсы (файлы, каталоги, сетевые соединения) доступны конкретным приложениям. В отличие от SELinux он более прост в настройке, так как использует файловую модель профилей, где ограничения задаются в виде текстовых файлов. Профили можно создавать в ручном режиме или автоматически с помощью инструментов обучения, которые анализируют поведение приложения.

ASLR (Address Space Layout Randomization) – это механизм защиты памяти, используемый в операционных системах для предотвращения атак, основанных на

предсказуемости адресов в памяти, таких как переполнение буфера или возврат в либ (return-to-libc). Он случайным образом изменяет базовые адреса ключевых участков памяти, включая стек, кучу, сегменты данных и код, а также загружаемые динамические библиотеки. Благодаря этому злоумышленнику становится практически невозможно предсказать, где в памяти находится нужный ему код или данные, что значительно затрудняет эксплуатацию уязвимостей, например, для реализации атак типа return-to-libc или использования шеллкода. ASLR особенно эффективен в сочетании с другими защитными механизмами, такими как NX-бит, делая атаки на память гораздо сложнее или вовсе невозможными.

NX-бит (No-eXecute бит) – аппаратная технология защиты памяти, используемая в современных процессорах для предотвращения выполнения кода в областях памяти, которые предназначены только для хранения данных. NX-бит помогает защитить систему от атак, таких как переполнение буфера, где злоумышленники пытаются записать и выполнить свой вредоносный код в памяти.

Когда NX-бит активирован, процессор маркирует определённые участки памяти (например, стек или кучу) как «неисполняемые». Если попытаться выполнить код из этих областей, процессор заблокирует выполнение, что предотвращает эксплуатацию уязвимости. Технология является важной частью современных механизмов защиты операционных систем (например, DEP – Data Execution Prevention в Windows).

Она эффективно снижает риск выполнения произвольного кода, добавляя дополнительный уровень защиты от атак.

Механизм активно используется в современных ОС, таких как Linux, Windows и macOS, в качестве меры повышения сложности атак и уменьшения вероятности их успешного выполнения.

Ключевым этапом исследования стало экспериментальное тестирование. В специально созданной тестовой среде были настроены различные защитные механизмы, после чего смоделированы различные виды атак, включая переполнение буфера, эскалацию привилегий и обход аутентификации. Дополнительно проведено нагрузочное тестирование для оценки влияния внедрённых механизмов защиты на производительность системы и отклик приложений. Для более глубокого анализа использовались инструменты фаззинга и статического анализа кода. Фаззинг помог выявить потенциальные уязвимости в ядре, генерируя случайные или некорректные входные данные и наблюдая за реакцией системы. Статический анализ исходного кода ядра позволил обнаружить возможные логические ошибки и уязвимости, связанные с управлением памятью и обработкой данных. Были проведены интервью с экспертами в области информационной безопасности и системными администраторами, имеющими опыт работы с защитными механизмами ядра Linux. Их практические знания и наблюдения предоставили ценные инсайты о реальных проблемах и эффективных методах защиты в производственных средах.

Статистический анализ инцидентов безопасности, связанных с операционными системами Linux, позволил выявить тенденции в методах атак и оценить эффективность существующих мер защиты. Анализ данных о зарегистрированных уязвимостях и их эксплуатации помог сфокусироваться на наиболее актуальных угрозах и разработать рекомендации по их предотвращению.

Результаты исследования

В результате проведённого исследования были выявлены ключевые аспекты эффективности защитных механизмов ядра Linux в предотвращении атак на уровне операционной системы. Анализ научной литературы и технической документации показал, что интеграция механизмов контроля доступа, таких как SELinux и AppArmor, существенно повышает уровень безопасности системы. Эти инструменты, основанные на мандатной модели контроля доступа, позволяют задавать детальные политики безопасности, ограничивающие действия процессов и пользователей, что снижает риск эксплуатации уязвимостей из-за неправомерных действий или ошибок конфигурации. Сравнительный анализ различных защитных механизмов продемонстрировал, что сочетание нескольких методов, таких как ASLR (Address Space Layout Randomization) и NX-бит (No-eXecute bit), обеспечивает более высокую устойчивость к атакам, направленным на переполнение буфера и выполнение произвольного кода (табл. 1). Экспериментальные тестирования в контролируемой среде подтвердили, что использование ASLR затрудняет злоумышленникам

предсказание расположения исполняемого кода и данных в памяти, тем самым уменьшая вероятность успешной эксплуатации уязвимостей:

Таблица 1 – Сравнительный анализ защитных механизмов ядра Linux

Механизм	Принцип действия	Преимущества	Недостатки
SELinux	Основан на мандатной модели контроля доступа (MAC), задаёт строгие политики безопасности.	Гибкость, высокая степень контроля над действиями процессов.	Сложен в настройке, требует глубоких знаний для конфигурации.
AppArmor	Использует MAC, профили проще в настройке, ограничивает доступ процессов и пользователей.	Простота настройки, подходит для сред с умеренными требованиями к безопасности.	Менее гибок в сложных сценариях, ограниченные возможности настройки.
ASLR	Случайно распределяет адреса областей памяти, усложняет предсказание расположения данных и кода.	Усложняет успешное выполнение атак на память, таких как переполнение буфера.	Может быть неэффективен при низкой энтропии или утечке адресов.
NX-бит	Маркирует области памяти как «неисполняемые», предотвращает выполнение кода из этих областей.	Защищает от исполнения вредоносного кода в уязвимых областях памяти.	Не защищает от атак, связанных с утечкой данных или предсказанием адресов.

Практические испытания с моделированием атак показали, что механизмы защиты памяти, такие как Stack Canaries эффективно предотвращают попытки переполнения буфера и манипуляции стеком вызовов. В ходе тестирования была зафиксирована значительная сложность в обходе этих защитных барьеров, что подтверждает их действенность в реальных условиях эксплуатации.

Stack Canaries – это защитный механизм, который добавляет специальное значение (канарейку) перед адресом возврата в стеке. Если переполнение буфера затрагивает канарейку, её значение изменяется, что позволяет обнаружить атаку до выполнения вредоносного кода. При вызове функции в стек перед адресом возврата записывается специальное значение (канарейка), генерируемое случайным образом при загрузке программы. Перед завершением функции это значение проверяется. Если канарейка была изменена, программа распознаёт это как попытку переполнения и аварийно завершает выполнение, предотвращая эксплуатацию уязвимости. Существует несколько типов канареек: статические, случайные и терминальные, которые используют байты, прерывающие строковые операции.

Разберем пример, указанный на рисунке 1:

1. Программа получает ввод из 100 символов «A».
2. В точке останова в функции askUser() происходит анализ стека (x/16x $rbp-32), где видно:
   - значение 0x75c55e80bc05af00 представляет собой канарейку,
   - адрес возврата (0x00005555555551db) остаётся неизменным.
3. Защита работает корректно: ввод «A» не затронул значение канарейки, что говорит о том, что переполнения буфера не произошло.

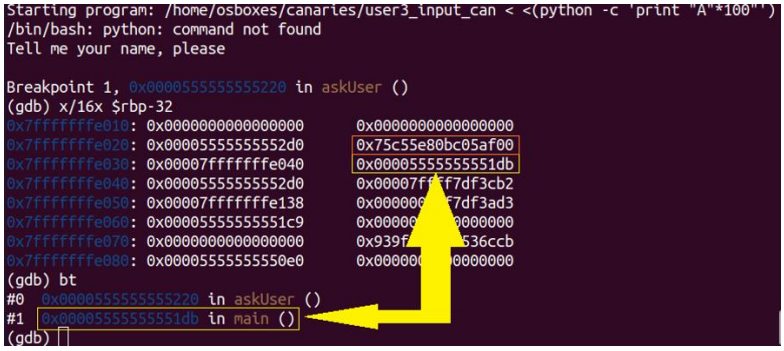

Рисунок 1 – Отладка программы в GDB (GNU Debugger) с целью анализа работы защиты Stack Canaries

Интервью с экспертами в области информационной безопасности подчеркнули важность регулярного обновления ядра и системных компонентов для поддержания высокого уровня защиты. Специалисты отметили, что своевременное применение патчей и обновлений устраняет известные уязвимости и снижает риск успешных атак. Кроме того, была подчеркнута необходимость правильной настройки защитных механизмов и обучения персонала, ответственного за безопасность системы. Статистический анализ инцидентов безопасности подтвердил тенденцию снижения количества успешных атак на системы с правильно настроенными и обновлёнными механизмами защиты ядра. Однако было отмечено, что человеческий фактор и ошибки конфигурации по-прежнему остаются значимыми факторами риска, что непосредственно указывает на необходимость не только технических, но и организационных мер, направленных на повышение осведомлённости пользователей и администраторов о современных угрозах и методах их предотвращения.

В целом, результаты исследования демонстрируют, что защитные механизмы ядра Linux являются эффективными инструментами в предотвращении широкого спектра атак на уровне операционной системы. Комплексный подход, включающий использование различных механизмов защиты, регулярное обновление системы и обучение персонала, значительно повышает общий уровень безопасности. Однако постоянное развитие методов атаки требует непрерывного совершенствования защитных мер и адаптации к новым угрозам, что подчёркивает важность дальнейших исследований в этой области.

Обсуждение научных результатов

Результаты проведенного исследования подтверждают важность и эффективность встроенных защитных механизмов ядра Linux в предотвращении атак на уровне операционной системы (рис. 2). Выявлено, что комбинация различных механизмов безопасности, таких как контроль доступа (SELinux, AppArmor), управление памятью (ASLR, NX-бит) и защита от переполнения буфера (Stack Canaries), обеспечивает многоуровневую защиту системы, существенно снижая вероятность успешной эксплуатации уязвимостей.

Рисунок 2 – Архитектура ядра операционной системы Linux и ее компоненты защиты

Анализ показал, что использование мандатных моделей контроля доступа, реализованных через SELinux и AppArmor, позволяет значительно ограничить возможности злоумышленников, даже в случае компрометации отдельных приложений или сервисов. Детально настроенные политики безопасности препятствуют несанкционированному доступу к критическим ресурсам и предотвращают эскалацию привилегий. Однако, эффективное применение этих механизмов требует глубокой экспертной настройки и понимания принципов их работы, что может быть затруднительно для неквалифицированных администраторов. Экспериментальные данные подтвердили, что механизмы защиты памяти, такие как ASLR и NX-бит, значительно усложняют проведение атак, основанных на переполнении буфера и выполнении произвольного кода. Рандомизация адресного пространства затрудняет злоумышленникам прогнозирование расположения полезной нагрузки, а маркировка областей памяти как невыполняемых предотвращает запуск кода в областях, предназначенных для данных. Тем не менее, были отмечены случаи, когда продвинутые техники эксплуатации могли обходить эти защиты, что свидетельствует о необходимости их дальнейшего совершенствования.

Обнаружено, что человеческий фактор и ошибки конфигурации остаются существенными рисками для безопасности системы. Неправильная настройка защитных

механизмов или игнорирование обновлений может свести на нет преимущества, предоставляемые ядром Linux. Это подчеркивает важность обучения и повышения квалификации администраторов, а также разработки инструментов для упрощения настройки и управления политиками безопасности.

Кроме того, анализ текущих тенденций в области кибербезопасности показал, что злоумышленники активно разрабатывают новые методы атаки, направленные на обход существующих защитных механизмов. Это требует постоянного обновления и адаптации защитных мер, а также внедрения дополнительных слоев безопасности, таких как системы обнаружения вторжений и поведенческий анализ.

Заключение

Проведённый анализ подтверждает, что ядро операционной системы Linux играет ключевую роль в обеспечении безопасности системы. Встроенные защитные механизмы, такие как контроль доступа, управление памятью и аутентификация, формируют многоуровневую защиту от различных видов атак. Они предотвращают несанкционированный доступ, выполнение вредоносного кода, обеспечивают изоляцию процессов и данных, что существенно повышает устойчивость системы к киберугрозам. Контроль доступа, реализованный через дискреционные и мандатные модели, позволяет гибко и эффективно управлять правами пользователей и процессов, минимизируя риск эксплуатации уязвимостей, связанных с человеческим фактором или ошибками конфигурации. Механизмы управления памятью, такие как виртуальная память, сегментация и страничная организация, обеспечивают защиту от атак, направленных на переполнение буфера и нарушения в работе памяти, а также изоляцию процессов. Практические примеры реальных атак и их предотвращения демонстрируют эффективность этих механизмов в действии. Однако следует отметить, что ни одна система не может быть абсолютно защищённой. Постоянное развитие методов атак требует непрерывного совершенствования защитных мер, регулярного обновления системы и повышения квалификации специалистов в области безопасности.

Современные тенденции в развитии защитных механизмов ядра Linux включают интеграцию аппаратных средств безопасности, использование методов машинного обучения для обнаружения аномалий и угроз, разработку новых подходов к изоляции и ограничению прав процессов. Важную роль играют пользователи и администраторы системы, чьи действия и решения существенно влияют на общий уровень безопасности.

Список литературы

1. Mauerer W. Professional Linux Kernel Architecture / W. Mauerer. – Indianapolis: Wrox Press, 2008. – 1368 p.
2. Love R. Linux Kernel Development / R. Love. – 3rd ed. – Indianapolis: Addison-Wesley Professional, 2010. – 440 p.
3. Sharma S. Linux Security Architecture / S. Sharma // International Journal of Computer Applications. – 2012. – Vol. 40, № 16. – P. 1-7.
4. Smalley S. Implementing SELinux as a Linux Security Module / S. Smalley, C. Vance, W. Salamon // NAI Labs Report. – 2001. URL: https://www.nsa.gov/portals/75/documents/resources/everyone/digital-media-center/publications/research-papers/implementing-selinux-as-linux-security-module-report.pdf (дата обращения: 15.10.2023).
5. Grimes R. Linux Buffer Overflow Attack Explained / R. Grimes // Security Strategies in Linux Platforms and Applications. – Boston: Jones & Bartlett Learning. – 2010. – P. 285-310.
6. Куркин А.В. Защита операционных систем семейства Linux / А.В. Куркин. – Москва: ДМК Пресс, 2015. – 320 с.
7. Васильев А.П. Механизмы безопасности в ядре Linux / А.П. Васильев, Б.А. Смирнов // Программные продукты и системы. – 2017. – № 4. – С. 120-125.
8. Edge J. Asynchronous Vulnerability Notification and Linux Kernel Security / J. Edge // Communications of the ACM. – 2019. – Vol. 62, № 6. – P. 18-20.
9. Müller T. A Systematic Assessment of the Security of Full Disk Encryption / T. Müller, F.C. Freiling // IEEE Transactions on Dependable and Secure Computing. – 2011. – Vol. 8, № 3. – P. 223-238.
10. Васильев Д.С. Анализ методов рандомизации адресного пространства в Linux / Д.С. Васильев// Информационная безопасность. – 2020. – Т. 23, № 2. – С. 45-51.

References

1. Mauerer W. Professional Linux Kernel Architecture / W. Mauerer. – Indianapolis: Wrox Press, 2008. – 1368 r. (In English).
2. Love R. Linux Kernel Development / R. Love. – 3rd ed. – Indianapolis: Addison-Wesley Professional, 2010. – 440 r. (In English).
3. Sharma S. Linux Security Architecture / S. Sharma // International Journal of Computer Applications. – 2012. – Vol. 40, № 16. – R. 1-7. (In English).
4. Smalley S. Implementing SELinux as a Linux Security Module / S. Smalley, C. Vance, W. Salamon // NAI Labs Report. – 2001. URL: https://www.nsa.gov/portals/75/documents/resources/everyone/digital-media-center/publications/research-papers/implementing-selinux-as-linux-security-module-report.pdf (data obrashcheniya: 15.10.2023). (In English).
5. Grimes R. Linux Buffer Overflow Attack Explained / R. Grimes // Security Strategies in Linux Platforms and Applications. – Boston: Jones & Bartlett Learning. – 2010. – R. 285-310. (In English).
6. Kurkin A.V. Zashchita operatsionnykh sistem semeistva Linux / A.V. Kurkin. – Moskva: DMK Press, 2015. – 320 s. (In Russian).
7. Vasil'ev A.P. Mekhanizmy bezopasnosti v yadre Linux / A.P. Vasil'ev, B.A. Smirnov // Programmnye produkty i sistemy. – 2017. – № 4. – S. 120-125. (In Russian).
8. Edge J. Asynchronous Vulnerability Notification and Linux Kernel Security / J. Edge // Communications of the ACM. – 2019. – Vol. 62, № 6. – R. 18-20. (In English).
9. Müller T. A Systematic Assessment of the Security of Full Disk Encryption / T. Müller, F.C. Freiling // IEEE Transactions on Dependable and Secure Computing. – 2011. – Vol. 8, № 3. – R. 223-238. (In English).
10. Vasil'ev D.S. Analiz metodov randomizatsii adresnogo prostranstva v Linux / D.S. Vasil'ev// Informatsionnaya bezopasnost'. – 2020. – T. 23, № 2. – S. 45-51. (In Russian).

И.Ж. Мейрамов
Еуразия ұлттық университеті Л.Н.Гумилев атындағы,
010000, Қазақстан Республикасы, Астана қ., Пушкин көшесі, 11
e-mail: gentelmen3332@mail.ru

LINUX ОПЕРАЦИЯЛЫҚ ЖҮЙЕСІ ДЕҢГЕЙІНДЕ ШАБУЫЛДАРДЫ БОЛДЫРМАУДА ЯДРОНЫҢ ҚОРҒАНЫС МЕХАНИЗМДЕРІНІҢ РӨЛІ

Қазіргі Linux операциялық жүйелерінде қауіпсіздік аса маңызды аспект болып табылады, ал ядро оны қамтамасыз етуде орталық рөл атқарады. Ядро аппараттық ресурстар мен қолданбалы бағдарламалық қамтамасыз ету арасындағы делдал ретінде жүйелік ресурстарға қолжетімділікті бақылайды және процестердің орындалуын басқарады. Ядроның негізгі функцияларының бірі – операциялық жүйе деңгейіндегі осалдықтарды пайдалану мақсатында бағытталған түрлі кибершабуылдардан жүйені қорғау. Бұл мақалада Linux ядросында іске асырылған негізгі қорғаныс механизмдері қарастырылады, мысалы, қолжетімділікті бақылау (SELinux, AppArmor), адрес кеңістігін рандомизациялау (ASLR), жадты қорғау (DEP, Stack Guard) және артық құқықтарды шектеу. Осы механизмдердің буферді толтыру, зиянды кодты енгізу және құқықтарды арттыру секілді шабуылдарды болдырмау немесе олардың әсерін азайту жолдары талқыланады. Сонымен қатар, осы әдістердің тиімділігі және олар Linux негізіндегі жүйелердің киберқауіпсіздігін қамтамасыз ету стратегиясындағы рөлі талданады.

Мақалада ядроның қорғаныс механизмдерін дамытудағы ағымдағы үрдістер мен болашағы да қамтылған, оның ішінде аппараттық қауіпсіздік құралдарын біріктіру және қауіптерді анықтау үшін машиналық оқыту технологияларын қолдану. Ядро мен жүйелік компоненттерді уақытылы жаңартудың, сондай-ақ, қауіпсіздіктің жоғары деңгейін қамтамасыз етудегі пайдаланушылар мен әкімшілердің белсенді рөлінің маңыздылығы атап өтіледі. Осылайша, мақалада Linux ядросының операциялық жүйе деңгейіндегі шабуылдарды болдырмауда қалай көмектесетіні және болашақта қорғауды күшейту үшін қандай шаралар қолданылуы мүмкін екендігі жан-жақты қарастырылады.

Түйін сөздер: *Linux ядросы, жүйе қауіпсіздігі, қорғаныс механизмдері, қолжетімділікті бақылау, құқықтарды шектеу, шабуылдар, буферді толтыру, кодты енгізу.*

I.Zh. Meyramov
Eurasian National University named after L.N. Gumilyov,
010000, Republic of Kazakhstan, Astana, Pushkin St., 11
e-mail: gentelmen3332@mail.ru

THE ROLE OF KERNEL SECURITY MECHANISMS IN PREVENTING ATTACKS AT THE LINUX OPERATING SYSTEM LEVEL

Security is a critically important aspect of modern Linux operating systems, with the kernel playing a central role in its provision. Acting as an intermediary between hardware resources and application software, the kernel controls access to system resources and manages process execution. One of its key functions is to protect the system from various cyber threats and attacks aimed at exploiting vulnerabilities at the operating system level. This paper examines the main security mechanisms implemented in the Linux kernel, including access control (SELinux, AppArmor), address space randomization (ASLR), memory protection (DEP, Stack Guard), and privilege restrictions. It discusses how these mechanisms prevent or mitigate the impact of attacks such as buffer overflows, malicious code injection, and privilege escalation. The effectiveness of these methods and their role in the overall cybersecurity strategy for Linux-based systems are analyzed.

Additionally, the paper highlights current trends and future prospects for the development of kernel security mechanisms, including the integration of hardware security tools and the use of machine learning technologies to detect threats. The importance of timely kernel and system component updates, as well as the active role of users and administrators in maintaining high-security standards, is emphasized. Thus, the article provides a comprehensive overview of how the Linux kernel contributes to preventing attacks at the operating system level and outlines measures that can be taken to enhance security in the future.

Key words: *Linux kernel, system security, security mechanisms, access control, privilege restrictions, attacks, buffer overflow, code injection.*

Сведения об авторах

Ильхан Жанатович Мейрамов – магистрант специальности «Системы информационной безопасности», Евразийский национальный университет имени Л.Н. Гумилева, Республика Казахстан, г. Астана; e-mail: gentelmen3332@mail.ru.

Автор туралы ақпарат

Илхан Жанатұлы Мейрамов – «Ақпараттық қауіпсіздік жүйелері» мамандығы бойынша магистрант, Еуразия ұлттық университеті Л.Н. Гумилев атындағы, Қазақстан Республикасы, Астана қ.; e-mail: gentelmen3332@mail.ru.

Information about the author

Ilkhan Zhanatovich Meyramov – Master's student in the specialty «Information Security Systems», Eurasian National University named after L.N. Gumilyov, Republic of Kazakhstan, Astana; e-mail: gentelmen3332@mail.ru.

Поступила в редакцию 25.11.2024
Поступила после доработки 02.12.2024
Принята к публикации 03.12.2024

https://doi.org/10.53360/2788-7995-2024-4(16)-17

МРНТИ: 50.43.15

CC BY 4.0
Check for updates

Б.Т. Иманбек[1,2], Ж.Е. Байғараева[1,2*], А.К. Болтабоева[1,2], Ж.Б. Кальпеева[1], А.Б. Копенов[2]
[1]Казахский национальный исследовательский технический университет им. К.И. Сатпаева,
050040,Республика Казахстан, г. Алматы, ул. К. Сатпаева, 22
[2]Казахский национальный университет имени аль-Фараби,
050040, Республика Казахстан, г. Алматы, пр. аль-Фараби, 71
*e-mail: zhanel.baigarayeva@gmail.com

ИНТЕЛЛЕКТУАЛЬНАЯ СИСТЕМА УПРАВЛЕНИЯ МОНИТОРИНГА И УПРАВЛЕНИЯ КЛИМАТОМ В ЧИСТЫХ ПОМЕЩЕНИЯХ НА ОСНОВЕ IOT И SCADA

Аннотация: *В статье представлены результаты исследований по разработке и внедрению интеллектуальной системы управления параметрами чистых помещений на основе технологий Интернета вещей (IoT), выполненных в рамках проекта по созданию автоматизированной системы мониторинга для медицинских и промышленных объектов. Установлены закономерности технологического процесса управления микроклиматом в чистых помещениях, включая контроль за температурой, влажностью и качеством воздуха, что обеспечивает высокую точность и надежность системы. Рациональные режимы работы системы, поддерживаемые IoT-сенсорами и платформой SCADA,*