DOI: 10.22184/1992-4178.2023.230.9.144.150

Надежность и безопасность операционных систем различной архитектуры

Часть 2

С. Назаров, д. т. н.[1], А. Барсуков, к. т. н.[2]

УДК 621.3.012 | ВАК 2.2.11

Во второй части статьи представлены модели двух основных архитектур операционных систем, которые наиболее широко распространены в реальных системах различных производителей. Приведены их представления в форме графов состояний и переходов, позволяющие определить основные параметры надежности функционирования систем по правилам составления систем алгебраических уравнений для установившегося состояния системы. Дана информация по принципу кибериммунного построения системы, положенного в основу операционной системы Касперского.

МОДЕЛЬ ОПЕРАЦИОННОЙ СИСТЕМЫ С МНОГОУРОВНЕВЫМ МОДУЛЬНЫМ ЯДРОМ

Граф состояний и переходов компьютерной системы с многоуровневой модульной операционной системой (типа Linux) представлен на рис. 2. Перечислим состояния и соответственно вероятности нахождения системы в этих состояниях: P_1 – вероятность работы ядра операционной системы ОСЯ в привилегированном режиме; P_2 – вероятность работы модулей операционной системы ОСП в пользовательском режиме; P_3 – вероятность работы пользовательских приложений ПП; P_4 – вероятность отказа системы.

Определим интенсивности переходов системы из одного состояния в другое (здесь и далее параметры моделей выбраны на основе измерений в реальных операционных системах [2]): λ_1 – интенсивность системных вызовов со стороны прикладных программ, примем значение $\lambda_1 = 10\,000$ 1/с; λ_2 – интенсивность системных вызовов со стороны прикладных программ, в выполнении которых участвуют устройства системы с соответствующими драйверами; пусть каждый 200-й системный вызов требует участия драйверов устройств, тогда значение $\lambda_2 = 50$ 1/с; λ_3 – интенсивность выполнения системных вызовов драйверами устройств. Пусть на выполнение одного вызова требуется 5 мс, тогда $\lambda_3 = 200$ 1/с; λ_4 – интенсивность передачи выполненных системных вызовов в прикладные программы. Эти вызовы выполняются только ОСЯ (назовем их короткими) или совместно с ОСП или ОСЯ передает результат работы ОСП (назовем их длинными). Примем среднее время передачи выполненных системных вызовов 0,1 мс. Тогда значение $\lambda_4 = 10\,000$ 1/с; λ_5 – интенсивность отказов ОСЯ. Выше была принята интенсивность отказов ядра Minix равной 100 000 ч при 6 программных ошибках. Размер ОСЯ был принят равным 10 млн строк. Число программных ошибок в данном случае следует принять равным не менее 3 ошибок на 1000 строк. Таким образом, ОСЯ содержит не менее 30 000 ошибок. Считая, что наработка на отказ обратно пропорциональна числу ошибок, получаем наработку на отказ для ОСЯ, равную 20 ч. При таких допущениях $\lambda_5 = 0{,}000014$ 1/с.

λ_6 – интенсивность отказов ОСП. Размер ОСП был принят равным 20 млн строк. Число программных ошибок в данном случае следует принять равным не менее 10 на 1000 строк кода (как отмечено выше, число ошибок в драйверах в три и более раз превышает число ошибок в программах ядра. Число ошибок в ОСП при принятых допущениях не менее 300 000. Однако следует заметить, реально используется в работе не более 1/10 возможностей ОСП. С учетом этого допущения можно принять $\lambda_6 = 0{,}000014$ 1/с. λ_7 – интенсивность отказов ПП. Считая средний размер пользовательского приложения 20 000 строк, можно считать, что в нем (при 20 ошибках на 1000 строк) порядка 400 ошибок. Отсюда $\lambda_7 = 0{,}0000047$ 1/с. λ_8 – интенсивность перезагрузки

[1] Московский научно-исследовательский телевизионный институт, главный научный сотрудник, профессор.

[2] Московский научно-исследовательский телевизионный институт, заместитель начальника научно-технического отдела, с. н. с.

операционной системы. Примем время перезагрузки равным 3 мин, тогда $\lambda_8 = 0{,}0055$ 1/с.

По графу состояний и переходов системы, представленному на рис. 2, можно составить систему уравнений, руководствуясь следующим правилом: слева в уравнениях стоит предельная вероятность данного состояния P_i, умноженная на суммарную интенсивность всех потоков, ведущих из данного состояния, а справа – сумма произведений интенсивностей всех потоков, входящих в i-е состояние, на вероятности тех состояний, из которых эти потоки исходят.

Полученная система уравнений имеет следующий вид:

$$\begin{cases} (\lambda_2+\lambda_4+\lambda_5)P_1=\lambda_1P_3+\lambda_3P_2+\lambda_8P_4; \\ (\lambda_3+\lambda_6)P_2=\lambda_2P_1; \\ (\lambda_1+\lambda_7)P_3=\lambda_4P_1; \\ \lambda_8P_4=\lambda_5P_1+\lambda_6P_2+\lambda_7P_3. \end{cases} \quad (1)$$

В данном случае имеем четыре уравнения в системе (1) при четырех неизвестных. Уравнения однородны (не имеют свободного члена) и определяют неизвестные только с точностью до произвольного множителя. Но можно воспользоваться так называемым нормировочным условием и с его помощью решить систему. При этом одно (любое) из уравнений можно отбросить (оно вытекает как следствие из остальных). Отбросим первое уравнение и, таким образом, получаем следующую систему уравнений:

$$\begin{cases} (\lambda_3+\lambda_6)P_2=\lambda_2P_1; \\ (\lambda_1+\lambda_7)P_3=\lambda_4P_1; \\ \lambda_8P_4=\lambda_5P_1+\lambda_6P_2+\lambda_7P_3; \\ P_1+P_2+P_3+P_4=1. \end{cases} \quad (2)$$

Подставив определенные выше значения коэффициентов при переменных и записав систему уравнений в матричной форме, удобной для решения средствами Excel, получим систему (2) в следующем виде:

$$\begin{cases} 50P_1-200{,}000014P_2+0P_3+0P_4=0; \\ 10\,000P_1+0P_2-10000{,}0000047P_3+0P_4=0; \\ 0{,}000014P_1+0{,}000014P_2+0{,}0000047P_3-0{,}0055P_4=0; \\ 1P_1+1P_2+1P_3+1P_4=1. \end{cases} \quad (3)$$

На решении этой системы уравнений остановимся позже, после построения всех предлагаемых в статье моделей.

МОДЕЛЬ МУЛЬТИСЕРВЕРНОЙ МИКРОЯДЕРНОЙ ОПЕРАЦИОННОЙ СИСТЕМЫ

В качестве примера такой операционной системы возьмем Unix-подобную структуру MINIX 3, предложенную Э. Таненбаумом более 15 лет назад [12]. В данном случае предлагается иметь несколько небольших модулей

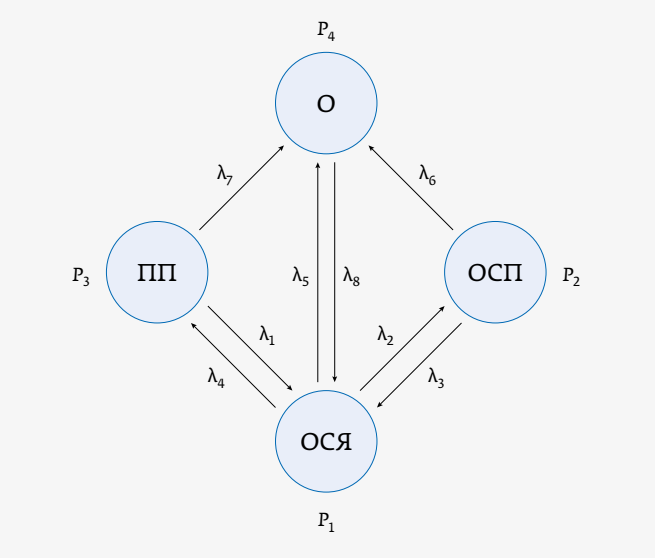

Рис. 2. Граф состояний и переходов системы с многослойным модульным ядром

(микроядро – МЯ и сервер реинкарнации – СР), работающих в режиме ядра, остальная часть операционной системы представляет собой набор полностью изолированных серверов (серверных процессы – СП) и драйверов (ДР), работающих в режиме пользователя. В операционной системе MINIX 3 микроядро обрабатывает прерывания, обеспечивает основные механизмы для управления процессами, реализует межпроцессные взаимодействия и производит планирование процессов. Оно также предоставляет небольшой набор вызовов ядра для авторизованных драйверов и серверов, например, для чтения части заданного пользовательского адресного пространства или записи в авторизованные порты ввода-вывода. В адресном пространстве микроядра работает драйвер таймера, но он планируется как отдельный процесс. Никакие другие драйверы в режиме ядра не работают. Над уровнем микроядра находится уровень драйверов устройств. Для каждого устройства ввода-вывода имеется собственный драйвер, выполняемый в виде отдельного процесса в собственном адресном пространстве, защищенном аппаратурой устройства управления памятью.

Драйверы работают в пользовательском режиме и не могут исполнять привилегированные команды, а также читать из портов компьютера или писать в них. Для получения последней возможности они должны производить вызовы ядра. Такая конструкция повышает надежность, хотя и порождает небольшие дополнительные расходы. Поверх уровня драйверов устройств располагается уровень серверов. Файловый сервер является небольшой (4 500 строк исполняемого кода) программой, которая принимает запросы от пользовательских процессов

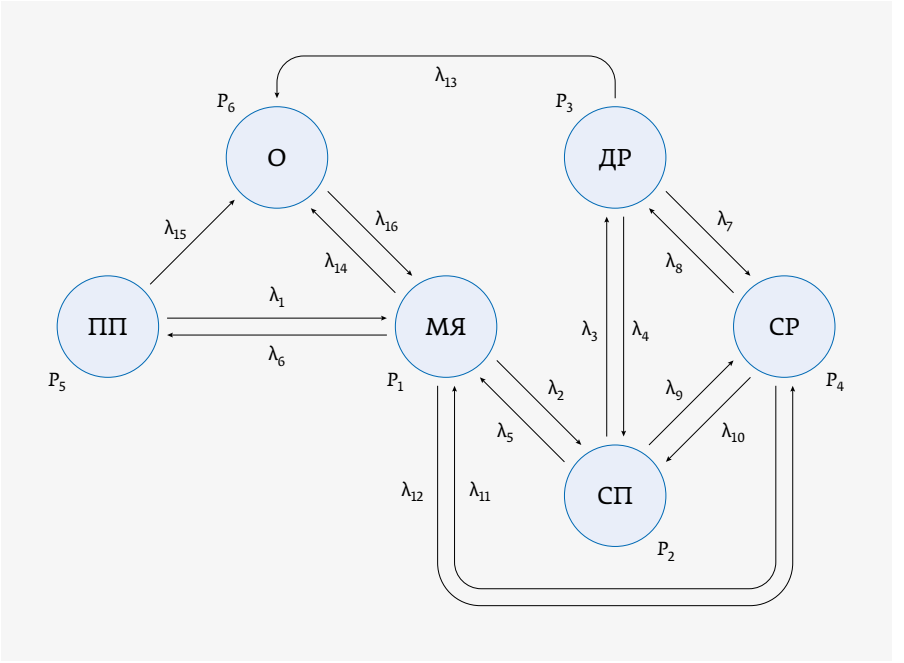

Рис. 3. Граф состояний и переходов мультисерверной микроядерной операционной системы

Кроме того, поддерживается механизм асинхронного уведомления о событиях. С системой передачи сообщений интегрирована обработка прерываний. Обработчики прерываний используют механизм уведомлений для сигнализации о завершении ввода-вывода. Этот механизм позволяет обработчику установить бит в битовой шкале «необработанных прерываний» драйвера и продолжить выполнение без блокировки. Когда драйвер становится готовым к получению прерывания, ядро преобразует его в обычное сообщение. Среди других особенностей, способствующих повышению надежности, наиболее важным является свойство самовосстановления. Если драйвер производит запись по неверному указателю, впадает в бесконечный цикл или неправильно ведет себя каким-либо другим образом, то сервер реинкарнации автоматически заменит его, часто без влияния на другие процессы. В соответствии с рассмотренной архитектурой микроядерной мультисерверной операционной системой ее граф состояний и переходов может быть представлен, как показано на рис. 3.

по обработке Posix-совместимых вызовов, относящихся к файлам read, write, lseek и stat, и выполняет их. На этом уровне находится и менеджер процессов, который поддерживает управление процессами и памятью и выполняет Posix-совместимые и другие системные вызовы, такие как fork, exec и brk. Сервер реинкарнации является родительским процессом всех других серверов и всех драйверов. Если драйвер или сервер аварийно или по собственной инициативе завершается, либо не отвечает на периодические запросы отклика, то сервер реинкарнации принудительно завершает его, если это требуется, и перезапускает из копии на диске или в основной памяти. В число других серверов входит сервер сети, поддерживающий весь стек TCP/IP; простой сервер имен, используемый всеми остальными серверами; и информационный сервер, способствующий отладке. Наконец, над уровнем серверов находятся пользовательские процессы (ПП). Для пользователей единственным отличием мультисерверной системы от других Unix-систем является то, что библиотечные процедуры для системных вызовов выполняют свою работу путем посылки сообщений серверам. Во всем остальном это обычные пользовательские процессы, в которых может использоваться API Posix. Межпроцессные взаимодействия (IPC) в MINIX 3 поддерживаются на основе передачи сообщений фиксированной длины с использованием принципа рандеву: система копирует сообщение напрямую от отправителя к получателю, когда оба они к этому готовы.

Перечислим состояния и соответственно вероятности нахождения системы в этих состояниях: P_1 – вероятность работы микроядра операционной системы МЯ в привилегированном режиме; P_2 – вероятность работы модулей сервисных процессов (СП) в пользовательском режиме; P_3 – вероятность работы драйверов (ДР) в пользовательском режиме; P_4 – вероятность работы сервера реинкарнации (СР) в пользовательском режиме; P_5 – вероятность работы пользовательских приложений (ПП); P_6 – вероятность отказа системы.

Аналогично тому, как это было сделано в предыдущей модели, определим интенсивности переходов системы из одного состояния в другое, сохраняя при этом характеристики выполняемых в системе программ пользователей ПП: λ_1 – интенсивность системных вызовов со стороны прикладных программ, примем значение $\lambda_1 = 10\,000$ 1/с; λ_2 – интенсивность системных вызовов со стороны прикладных программ, в выполнении которых участвуют сервисные процессы (СП) без устройств компьютерной системы и, следовательно, без драйверов. Учитывая (как принято в модели 1), что каждый 200-й системный вызов требует участия драйверов устройств, тогда значение $\lambda_2 = 50$ 1/с; λ_3 – интенсивность системных вызовов со стороны прикладных программ, требующих участия в их

SemiTEq

АКЦИОНЕРНОЕ ОБЩЕСТВО
«НАУЧНОЕ И ТЕХНОЛОГИЧЕСКОЕ
ОБОРУДОВАНИЕ»

STE ICP200

УНИВЕРСАЛЬНАЯ ТЕХНОЛОГИЧЕСКАЯ ПЛАТФОРМА ПЛАЗМОХИМИЧЕСКОГО ТРАВЛЕНИЯ И ОСАЖДЕНИЯ С ОПЦИЕЙ КАССЕТНОЙ ЗАГРУЗКИ

ICP RIE / RIE
ICP PECVD / PECVD

АО «НТО»
194156, Россия, г. Санкт-Петербург,
пр. Энгельса, д. 27
Тел.: +7 (812) 601-06-05
E-mail: sales@semiteq.ru

СДЕЛАНО В РОССИИ

www.semiteq.ru

выполнении драйверов устройств; $\lambda_3 = 50$ 1/с; λ_4 – интенсивность выполнения системных вызовов драйверами устройств; пусть на выполнение одного вызова требуется 5 мс, тогда $\lambda_4 = 200$ 1/с; λ_5 – интенсивность передачи результатов системных вызовов, выполненных самим микроядром МЯ и драйверами совместно с сервисными процессами, в прикладную программу ПП. Примем среднее время передачи 0,01 мс. Тогда значение $\lambda_5 = 100\,000$ 1/с; λ_6 – интенсивность передачи результатов завершения выполнения системных вызовов драйверами совместно с сервисными процессами, в пользовательские процессы ПП (по сути, это разблокировка процессов). Примем среднее время передачи 0,01 мс. Тогда значение $\lambda_6 = 100\,000$ 1/с.

Выше была принята наработка на отказ ядра Minix, равная 100 000 ч при шести программных ошибках. Примем размер драйверов 1 млн строк. Число программных ошибок в данном случае следует принять равным не менее 20 на 1000 строк кода. Таким образом, драйверы содержат не менее 20 000 ошибок. Считая, что наработка на отказ обратно пропорциональна числу ошибок, получаем наработку на отказ для ДР, равную 30 ч. При таких допущениях $\lambda_7 = 0{,}00000923$ 1/с. λ_8 – интенсивность перезапуска драйверов сервером реинкарнации. Перезапуск может происходить из оперативной памяти или (если в данный момент там есть его копия) или с диска (в противном случае). Примем среднее значение времени перезапуска драйвера 0,01 мс. Таким образом, $\lambda_8 = 100\,000$ 1/с; λ_9 – интенсивность отказов серверных процессов (СП). Будем считать, надежность работы СП примерно такая же, как и драйверов. В этом случае $\lambda_9 = 0{,}00000923$ 1/с. λ_{10} – интенсивность перезапуска серверных процессов сервером реинкарнации. Можно считать это действие аналогичным действию перезапуска драйверов. Тогда $\lambda_{10} = 100\,000$ 1/с; λ_{11} – интенсивность отказов сервера реинкарнации (СР). Следует считать, что СР также надежен, как и микроядро. В противном случае пропадает сама суть микроядерной операционной системы. Следовательно, наработка на отказ СР равна 100 000 ч и $\lambda_{11} = 0{,}0000000028$ 1/с; λ_{12} – интенсивность перезапуска сервера реинкарнации.

Предполагая, что этот процесс аналогичен перезапуску ДР и СП, будем считать, что $\lambda_{12} = 100\,000$ 1/с; λ_{13} – интенсивность фатальных отказов СР (восстановление СР с помощью МЯ невозможно). Примем, что интенсивность таких отказов на порядок ниже интенсивности отказов СР, когда восстановление возможно, тогда $\lambda_{13} = 0{,}0000000028$ 1/с; λ_{14} – интенсивность фатальных отказов МЯ. Считая, что надежности СР и МЯ эквивалентны, $\lambda_{14} = 0{,}0000000028$ 1/с; λ_{15} – интенсивность фатальных отказов пользовательских приложений. Примем это значение, равным соответствующему значению в модели 1, то есть $\lambda_{15} = 0{,}0000047$ 1/с. λ_{16} – интенсивность перезагрузки операционной системы. Примем время перезагрузки равное трем минутам, тогда $\lambda_{16} = 0{,}0055$ 1/с. Аналогично тому, как это сделано в модели по рис. 2, составляем систему уравнений для модели по рис. 3:

$$\begin{cases} (\lambda_2+\lambda_6+\lambda_{12}+\lambda_{14})P_1=\lambda_5P_2+\lambda_{11}P_4+\lambda_1P_5+\lambda_{16}P_6; \\ (\lambda_3+\lambda_5+\lambda_9)P_2=\lambda_2P_1+\lambda_4P_3+\lambda_{10}P_4; \\ (\lambda_4+\lambda_7)P_3=\lambda_3P_2+\lambda_8P_4; \\ (\lambda_8+\lambda_{10}+\lambda_{11}+\lambda_{13})P_4=\lambda_{12}P_1+\lambda_9P_2+\lambda_7P_3; \\ (\lambda_1+\lambda_{15})P_5=\lambda_6P_1; \\ \lambda_{16}P_6=\lambda_{14}P_1+\lambda_{13}P_4+\lambda_{15}P_5; \\ P_1+P_2+P_3+P_4+P_5+P_6=1. \end{cases} \quad (4)$$

Исключаем из системы (4) первое уравнение. Учитывая установленные выше значения коэффициентов при переменных и записав систему уравнений в матричной форме, удобной для решения средствами Excel, получим систему (4) в следующем виде:

$$\begin{cases} 950P_1-100\,050P_2+200P_3+100000P_4+0P_5+0P_6=0; \\ 0P_1+50P_2-200P_3+100\,000P_4+0P_5+0P_6=0; \\ 100\,000P_1+0{,}00000923P_2+0{,}00000923P_3- \\ \quad -200\,000P_4+0P_5+0P_6=0; \\ 100\,000P_1+0P_2+0P_3+0P_4-10\,000P_5+0P_6=0; \\ 0{,}0000000028P_1+0P_2+0P_3+0{,}0000000028P_4+ \\ \quad +0{,}0000047P_5-0{,}0055P_6=0; \\ P_1+P_2+P_3+P_4+P_5+P_6=1. \end{cases} \quad (5)$$

Также как и после системы уравнений, описывающих первую модель ОС, на решении этой системы уравнений остановимся позже, после построения всех предлагаемых в статье моделей.

МОДЕЛЬ ОПЕРАЦИОННОЙ СИСТЕМЫ КАСПЕРСКОГО

По словам главы «Лаборатории Касперского» Евгения Касперского [17], сегодня существуют три глобальные проблемы безопасности, с которыми государствам и бизнесу приходится сталкиваться и которые «Лаборатория Касперского» решает, благодаря разработанной концепции кибериммунности и собственной операционной системе. Первая проблема – массовая киберпреступность. Успехи лаборатории в этой области широко известны. Свыше 400 млн пользователей и 270 тыс. корпоративных клиентов во всем мире уже доверяют свою защиту продуктам «Лаборатории Касперского». Вторая проблема – профессиональная киберпреступность, которая становится все более подготовленной, консолидированной и технически обеспеченной. Бороться с ней можно с помощью многофакторной эшелонированной защиты, включающей в себя защиту периметра, сети, трафика и др. Третья проблема – атаки на промышленную и критическую инфраструктуру, которые характеризуются наивысшей

КБТЭМ ПЛАНАР | 60 лет
1962—2022

Оборудование для проведения испытаний в области электробезопасности в соответствии с ГОСТ Р МЭК 61032-2000 (СТБ МЭК 61032-2001)

КОМПЛЕКТ ЩУПОВ ДОСТУПНОСТИ

КОМПЛЕКТ ПАЛЬЦЕВ ИСПЫТАТЕЛЬНЫХ

Комплекты предназначены для испытаний защиты, обеспечиваемой оболочками, от проникновения твердых предметов (включая защиту людей от доступа к опасным частям изделий и защиту оборудования внутри оболочки от попадания посторонних твердых предметов) в соответствии с ГОСТ 14254-2015.

ИСПЫТАТЕЛЬНЫЙ ЦЕНТР
тел.: (+375 17) 226-10-31, e-mail: ok.kokhovich@mail.by

ОАО «Планар»
220033, Республика Беларусь, г. Минск, Партизанский пр-т 2, корп. 2-31;
факс.: +375 17 226-12-05; тел.: +375 17 297-37-09; www.planar.by, office@kbtem-omo.by

planar.by

степенью риска и в случае успеха наносят наиболее сильный ущерб.

В критической инфраструктуре риски зашкаливают, и решить эту проблему только добавленными средствами безопасности невозможно. Именно поэтому лаборатория разработала концепцию кибериммунитета. Кибериммунной Е. Касперский называет такую систему, стоимость организации атаки на которую выше, чем возможный ущерб. Достичь этого позволяет операционная система Kaspersky OS с микроядерной архитектурой, где все связи и взаимодействия проходят уровень безопасности. Причем это разрешительная система безопасности, а не запретительная. Если какой-то ее элемент будет поражен, вредоносное ПО не проникнет дальше. В этом главное отличие Kaspersky OS от традиционных операционных систем. Важно, что этот принцип может быть реализован не только в операционных системах для ПК. На рынке уже представлено несколько устройств и комплексных решений, работающих на Kaspersky OS.

Понятие кибериммунитета основано на концепции Secure by Design (безопасность при разработке), ключевой принцип которой состоит в том, что безопасность должна являться неотъемлемой частью любой разрабатываемой системы, присутствовать в каждом ее компоненте и сопровождать весь цикл разработки, начиная с этапа проектирования. Таким образом, у системы как бы появляется «врожденный иммунитет» к различного рода киберугрозам – как существующим, так и новым. Kaspersky OS – это инструмент, который помогает сторонним разработчикам быстрее создавать более безопасные программные и аппаратные решения. В этом смысле можно определить кибериммунитет как инновационную комбинацию инструментов, методологий и способов разработки программного обеспечения по принципу Secure by Design.

Дальнейшее рассмотрение принципов, положенных в основу построения операционной системы Касперского, и особенностей архитектуры этой системы, базирующейся на кибериммунном подходе, будет дано в заключительной части статьи. В этой же части статьи будут приведены результаты расчета систем уравнений по всем рассмотренным моделям и обсуждены полученные результаты моделирования.

ЛИТЕРАТУРА

1. **Назаров С. В.** Эффективность и оптимизация компьютерных систем. Монография / 2-е изд. М.: РУСАЙНС, 2021. 294 с.
2. **Назаров С. В.** Эффективность современных операционных систем // Современные информационные технологии и ИТ-образование. 2017. Т. 13 № 2. С. 9–24.
3. **Назаров С. В., Вилкова Н. Н.** Эффективность систем отображения информации коллективного пользования // Электросвязь. 2015. № 9. С. 29–33.
4. **Назаров С. В., Вилкова Н. Н.** Выбор оптимального варианта системы отображения информации коллективного пользования // Электросвязь. 2017. № 1. С. 60–65.
5. **Таненбаум Э., Вудхалл А.** Операционные системы. Разработка и реализация / 3-е изд. СПб: Питер, 2007. 704 с.
6. **Назаров С. В.** Архитектура и проектирование программных систем: монография / 2-е изд., перераб. и доп. М.: ИНФРА-М, 2016. 376 с.
7. **Таненбаум Э., Хердер Дж., Бос Х.** Построение надежных операционных систем, допускающих наличие ненадежных драйверов устройств. [Электронный ресурс]. URL: http://citforum.ru/operating_systems/microkernel_tanenbaum/
8. **Назаров С. В., Широков А. И.** Технологии многопользовательских операционных систем. М.: Изд. дом МИСиС, 2012. 296 с.
9. **Назаров С. В., Вилкова Н. Н.** Структурный рефакторинг многослойных программных систем // Информационные технологии и вычислительные системы. 2016. № 4. С. 13–23.
10. **Tanenbaum Andrew S., Herder Jorrit N., Bos Herbert.** Vrije Universiteit, Amsterdam. Can We Make Operating Systems Reliable and Secure? Computer (IEEE Computer Society, V. 39, No 5, May 2006).
11. **Хердер Й., Бос Х., Таненбаум Э.** Построение надежных операционных систем, допускающих наличие ненадежных драйверов устройств, 2006. [Электронный ресурс]. URL: http://citforum.ru/operating_systems/reliable_os/
12. **Tanenbaum A.** Introduction to MINIX 3 // OS News is Exploring the Future of Computing. 2006. [Электронный ресурс]. URL: http://www.osnews.com/story/15960/
13. **Игнатов Р.** MINIX 3 – реинкарнация? [Электронный ресурс]. URL: http://www.minix3.ru/articles/Ignatov-_minix_reincarnation.pdf
14. **Таненбаум Э., Бос Э.** Современные операционные системы // 4-е изд. СПб: Питер, 2015. 1120 с.
15. **Кельберт М. Я., Сухов Ю. М.** Вероятность и статистика в примерах и задачах. Т. 2. Марковские цепи как отправная точка теории случайных процессов и их приложения. М.: МЦНМО, 2009. 476 с.
16. **Кемени Дж., Снелл Дж.** Конечные цепи Маркова. М.: Наука, 1970. 273 с.
17. **Блинов А.** Лаборатория Касперского – об основах кибериммунитета и концепции KasperskyOS. [Электронный ресурс]. URL: https://spbit.ru/news/Laboratoriya-Kasperskogo-ob-osnovakh-kiberimmuniteta
18. Архитектура KasperskyOS [Электронный ресурс]. URL: https://support.kaspersky.ru/help/KCE/1.1/ru-RU/overview_architecture.htm
19. Микроядро KasperskyOS. [Электронный ресурс]. URL: https://os.kaspersky.ru/technologies/microkernel/?ysclid=ll3ego7wh4870216945

АКЦИОНЕРНОЕ ОБЩЕСТВО
ЭЛЕКОНД

РАЗРАБОТКА И ПРОИЗВОДСТВО КОНДЕНСАТОРОВ

оксидно-электролитические алюминиевые конденсаторы

К50-15, К50-17, К50-27, К50-29, К50-37, К50-68, К50-77, К50-80, К50-81, К50-83, К50-84, К50-85, К50-86, К50-87, К50-88, К50-89, К50-90, К50-91, К50-92, К50-93, К50-94, К50-95(чип), К50-96, К50-97(чип), К50-98, К50-99, К50-100, К50-101(чип), К50-102, К50-103, К50-104, К50-105, К50-106



объемно-пористые танталовые конденсаторы

К52-1, К52-1М, К52-1БМ, К52-1Б, К52-9, К52-11, К52-17, К52-18, К52-19, К52-20, К52-21, К52-24, К52-26(чип), К52-27(чип), К52-28, К52-29, К52-30



оксидно-полупроводниковые танталовые конденсаторы

К53-1А, К53-7, К53-65(чип), К53-66, К53-68(чип), К53-69(чип), К53-71(чип), К53-72(чип), К53-74(чип), К53-77(чип), К53-78(чип), К53-82



суперконденсаторы (ионисторы)

К58-26, К58-27, К58-28, К58-29, К58-30, К58-31



накопители электрической энергии на основе модульной сборки суперконденсаторов

НЭЭ, МИК, МИЧ, ИТИ



Система менеджмента качества сертифицирована на соответствие требованиям ISO 9001

Россия, 427968, Удмуртская Республика,г. Сарапул, ул. Калинина, 3
Тел.: (34147) 2-99-53, 2-99-89, 2-99-77, факс: (34147) 4-32-48, 4-27-53
e-mail: elecond-market@elcudm.ru, http://www.elecond.ru