Кузьмина Анна Михайловна
студентка
Маврин Сергей Алексеевич
канд. пед. наук, доцент

ФГБОУ ВО «Самарский государственный
социально-педагогический университет»
г. Самара, Самарская область

АНАЛИЗ ПРОИЗВОДИТЕЛЬНОСТИ UBUNTU SERVER С ИСПОЛЬЗОВАНИЕМ EBPF

Аннотация: *в статье рассматривается технология eBPF как средство углубленного мониторинга производительности Ubuntu Server. Показаны ограничения классических утилит (top, iostat, netstat), не позволяющие идентифицировать конкретные файлы, функции и короткоживущие процессы. Описаны архитектура eBPF, механизмы проб и верификатор ядра. Приведена инструкция по установке BCC и bpftrace в Ubuntu Server с примерами проверки работоспособности. Сделан вывод, что внедрение eBPF позволяет администратору перейти от реактивного устранения проблем к проактивному анализу узких мест, а доступность инструментов в стандартных репозиториях снижает порог входа.*

Ключевые слова: *системные вызовы, eBPF, Ubuntu Server, мониторинг производительности, BCC, bpftrace, трассировка ядра, диагностика Linux, kprobe, производительность серверов.*

При диагностике проблем производительности в операционных системах семейства Linux начинающие системные администраторы неизбежно сталкиваются с типовыми вопросами, среди которых можно выделить следующие вопросы. Чем обусловлено аномальное снижение отклика сервера. Какой процесс создаёт избыточную нагрузку на подсистему ввода-вывода. Какое приложение использует конкретный сетевой порт.

Классические средства мониторинга – top, iostat, vmstat, netstat, htop – предоставляют оперативную, но агрегированную информацию о состоянии системы. Их ключевой недостаток заключается в том, что они оперируют усреднёнными значениями и не позволяют выполнить следующие задачи. Установить, какие именно файлы вовлечены в операции чтения или записи при высокой утилизации диска. Идентифицировать конкретную функцию внутри исполняемого файла, ответственную за пиковую загрузку процессора. Детектировать короткоживущие (эфемерные) процессы, время жизни которых не превышает интервала дискретизации данных, например 100 мс. Такой процесс может быть инициирован, выполнить целевую операцию и завершиться до того, как инструмент класса top выполнит очередной сбор показателей. Следовательно, для решения указанного класса задач требуются механизмы событийно-ориентированного наблюдения с минимальными накладными расходами, работающие на уровне ядра операционной системы.

Одним из таких механизмов является технология eBPF (Extended Berkeley Packet Filter), позволяющая выполнять верифицированные программы в ядре Linux в ответ на различные системные события. eBPF предоставляет возможность наблюдения за работой ядра без остановки сервера. Для наглядности можно провести аналогию. Ядро Linux подобно крупной производственной площадке. Традиционные инструменты мониторинга обходят площадку раз в секунду и фиксируют обобщённые показатели. eBPF позволяет разместить миниатюрные датчики непосредственно на любом оборудовании, которые срабатывают в момент наступления события и не вмешиваются в ход производственного процесса. Безопасность обеспечивается верификатором ядра. Перед запуском каждая программа eBPF проходит строгую проверку, гарантирующую отсутствие бесконечных циклов, обращений к недопустимой памяти и опасных операций. Для начинающего администратора Ubuntu Server наиболее полезны следующие сценарии. Мониторинг дисковых операций, отслеживание задержек чтения и записи, выявление медленных операций. Сетевой мониторинг, отслеживание но-

вых соединений и анализ сетевой активности по процессам. Трассировка системных вызовов, наблюдение за системными вызовами процессов. Обнаружение аномалий, выявление подозрительной активности.

Рассмотрим несколько примеров практического применения eBPF. Программа eBPF представляет собой небольшой фрагмент кода, загружаемый в ядро и выполняемый при наступлении определённого события. События (hooks, пробы) – это места в коде ядра или пользовательских приложений, к которым можно «прикрепить» программу eBPF. Основные типы событий включают kprobe и kretprobe (точки входа и выхода функций ядра, гибко, но может быть нестабильно), tracepoints (заранее определённые «зацепки», более стабильны), uprobe (точки входа в функции пользовательских программ). Карты (maps) – это структуры данных в ядре для хранения счётчиков, временных меток и результатов работы программ eBPF. Верификатор – это компонент ядра, проверяющий каждую программу eBPF на безопасность перед запуском.

Для работы с eBPF существуют готовые инструменты. Для начинающих наиболее удобным вариантом является BCC (BPF Compiler Collection). Это коллекция готовых инструментов eBPF командной строки (около 100 штук), каждый из которых решает одну конкретную задачу и работает «из коробки». Если готового инструмента нет, используется язык bpftrace. Его синтаксис, вдохновлённый языками DTrace и AWK, позволяет быстро писать простые программы eBPF для нестандартных задач. Далее рассмотрим процесс установки и настройки этих инструментов на примере операционной системы Ubuntu Server. Перед установкой любого нового ПО необходимо обновить локальный кэш списка доступных пакетов, чтобы система знала о последних версиях и зависимостях. Это стандартный первый шаг при установке пакетов в Ubuntu. В выводе отображаются строки загрузки метаданных и итоговая сводка об успешном завершении. Обновление списка пакетов выполняется с помощью команды: «sudo apt update» (рисунок 1).

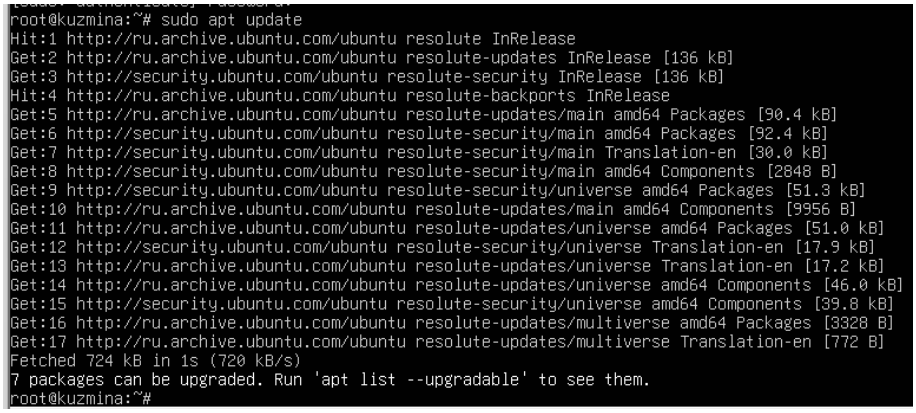

Рис. 1. Обновление списка пакетов

Рассмотрим установку BCC и заголовков ядра с помощью команды: «sudo apt install -y bpfcc-tools linux-headers-$(uname -r)” (рисунок 2). bpfcc-tools – коллекция готовых инструментов eBPF (например, biolatency-bpfcc, tcpconnect-bpfcc), которые позволяют сразу начать мониторинг системы без написания кода. linux-headers – необходимы для компиляции программ eBPF и взаимодействия с ядром. Флаг -y автоматически подтверждает установку.

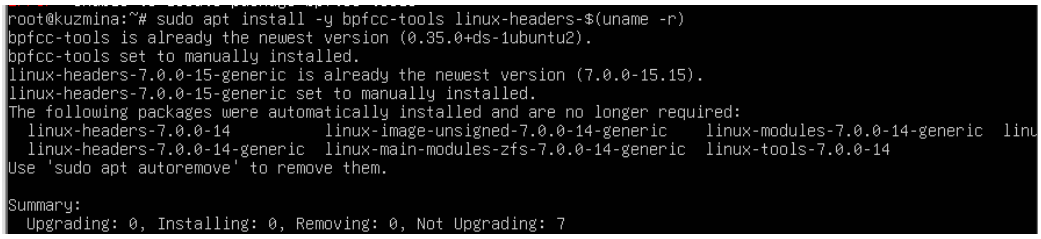

Рис. 2. Установка BCC и заголовков ядра

Проверим количество установленных инструментов BCC командой «ls /usr/sbin/*bpfcc* | wc -l» (рисунок 3). Проверка, что установка прошла успешно и все инструменты BCC доступны. Команда ls выводит список файлов, а wc -l считает количество строк – так можно быстро убедиться, что пакет установился корректно. Вывод представляет собой число, отражающее количество утилит BCC в системе.

Рис. 3. Проверка количества установленных инструментов BCC

Далее выполним установку пакета bpftrace через «sudo apt install» (рисунок 4). Проверка, что установка прошла успешно и все инструменты BCC доступны. Команда ls выводит список файлов, а wc -l считает количество строк – так можно быстро убедиться, что пакет установился корректно. Вывод представляет собой число, отражающее количество утилит BCC в системе.

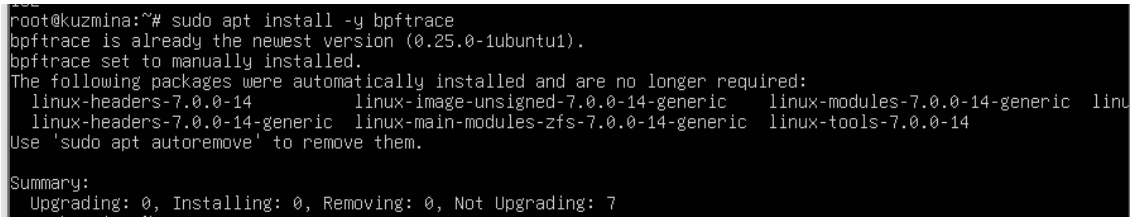

Рис. 4. Установка пакета bpftrace

Проверим работоспособности bpftrace с помощью команды: «sudo bpftrace -e 'BEGIN { printf(«eBPF работает!\n»); exit(); }' ” (рисунок 5). Проверка, что eBPF-подсистема ядра работает корректно и bpftrace может загружать программы в ядро. Если команда не выдаст ошибку и покажет приветственное сообщение – значит, всё настроено правильно. Вывод содержит строку «eBPF работает!», подтверждающую корректную работу eBPF-подсистемы ядра.

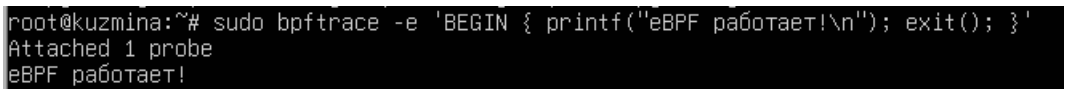

Рис. 5. Проверка работоспособности bpftrace

Технология eBPF меняет способ мониторинга Linux-систем. Для начинающего администратора Ubuntu Server eBPF открывает окно во внутренности операционной системы, которое раньше было доступно только разработчикам ядра. Утилиты семейства BCC (biolatency-bpfcc и другие) и язык bpftrace должны стать частью арсенала каждого системного администратора. Они помогают быстрее находить причины проблем и увереннее чувствовать себя при диагностике производительности. Рекомендуется начать с малого: установить BCC, запустить

sudo biolatency-bpfcc во время штатной работы сервера и проанализировать типичные задержки.

Таким образом, технология eBPF представляет собой парадигмальный сдвиг в подходах к мониторингу и анализу производительности операционных систем Linux. Показано, что классические средства диагностики, основанные на агрегированных данных, не способны обеспечить необходимый уровень детализации для решения современных задач по эксплуатации серверной инфраструктуры. Внедрение инструментов eBPF, таких как BCC и bpftrace, позволяет системному администратору перейти от реактивного устранения последствий сбоев к проактивному анализу узких мест. Для администраторов Ubuntu Server доступность этих инструментов через стандартные репозитории значительно снижает барьер входа, поскольку избавляет от необходимости сборки из исходных кодов. Дальнейшее развитие данного направления видится в интеграции мониторинга на базе eBPF в комплексные системы управления ИТ-инфраструктурой для автоматизации процессов диагностики и обеспечения отказоустойчивости.

References

1. BCC (BPF Compiler Collection) Documentation. – URL: https://github.com/iovisor/bcc (date of access: 23.06.2026).

2. bpftrace Documentation. – URL: https://github.com/iovisor/bpftrace (date of access: 23.06.2026).

3. Ubuntu Server Administration Guide. – URL: https://ubuntu.com/server/docs (дата обращения: 23.06.2026).

4. Linux Man Pages: top(1), iostat(1), vmstat(8), netstat(8). – URL: https://man7.org/linux/man-pages/ (date of access: 23.06.2026).