Engineer Платформа

Linux та сервери

27 досягнень

Поєднайте частини системи

Роботи за темою «Linux та сервери» показують одну частину дієвої системи. Ми можемо допомогти пов'язати її з рештою вашої діяльності. Моніторинг і сповіщення, які попереджають, поки збій ще малий, і цифри, що показують, як часто так і залишається.

Переглянути наші послуги Зв'язатися з нами →

Знайдіть схоже завдання

Відкрийте приклад, схожий на вашу ситуацію. Огляд пояснює рішення, що привели до результату, а не лише називає технологію. Запустила перший дашборд спостережуваності — перший огляд власних систем компанії

Запустила перший дашборд спостережуваності Зв'язатися з нами →

Запустила перший дашборд спостережуваності

Розробила та запустила перший у компанії дашборд спостережуваності, що забезпечував аналітику продуктивності системи в реальному часі та візуалізацію даних на великому офісному телевізорі.

Підтримувала 10 серверів PostgreSQL і MS SQL

Спроєктувала, розгорнула та підтримувала 10 серверів PostgreSQL і MS SQL на Ubuntu Linux VPS, забезпечивши оптимальну продуктивність і надійність серверів.

Написала 1 000 правил RBAC для Linux-серверів

Посилила безпеку даних, впровадивши 1 000 правил RBAC для розробників, інстансів застосунків, PostgreSQL, MS SQL та інших Linux-серверів, запобігши несанкціонованому доступу; задокументувала їх як автоматизацію в Ansible.

Керувала 30 інстансами Ubuntu Linux VPS

Керувала 30 інстансами Ubuntu Linux VPS, впровадивши стратегії аварійного відновлення та забезпечивши оптимальні конфігурації мережі.

Керувала 8 з'єднаннями WireGuard VPN та IPsec VPN

Керувала 8 з'єднаннями WireGuard VPN та IPsec VPN й усувала в них несправності, забезпечивши безпечний зв'язок між системами Google Cloud та Linux.

Адмініструвала мережеву інфраструктуру

Адмініструвала мережеву інфраструктуру для понад 1 000 серверів, забезпечивши оптимальне розгортання систем, безпеку та усунення несправностей.

Автоматизувала SSL/TLS для 100 застосунків Docker

Автоматизувала створення сертифікатів SSL/TLS для 100 застосунків Docker, забезпечивши безпечні з'єднання на хостах Ubuntu Linux.

Адмініструвала 100 серверів Bare Bone

Адмініструвала 100 серверів Bare Bone, фізичні мережі та системи IP-телефонії, забезпечивши надійну інфраструктуру для зростання компанії.

Адмініструвала 40 вебсайтів на Apache і Nginx

Адмініструвала 40 вебсайтів на хостинг-серверах Ubuntu Linux з Apache та Nginx, забезпечивши високу доступність і продуктивність.

Підтримувала 24/7 близько 1 000 серверів IPTV/OTT

Забезпечувала підтримку інфраструктури в режимі 24/7 для IPTV/OTT-стрімінгової платформи, адмініструючи ~1 000 серверів та системи клієнтів для глобальних замовників у Китаї, США та Німеччині.

Забезпечувала безперебійну передачу сигналів IPTV

Забезпечувала безперебійну передачу сигналів IPTV-стрімінгу між постачальниками та клієнтами, цілодобово відстежуючи та підтримуючи стрімінгову мережу й IP-телефонію.

Інфраструктура, описана в репозиторії

Побудувала власну інфраструктуру компанії у вигляді 19 плейбуків Ansible і 34 ролей обсягом 12 065 рядків YAML, які приводять робочий хост до задекларованого стану, причому кожен play ідемпотентний.

Публічні служби на невеликому сервері

Розмістила публічні служби компанії на одноядерному хості з 512 МБ — git-сервер, вебсервер для семи доменів, Tor, два сервери альтернативних протоколів і блокування вторгнень, — використовуючи 464 МБ доступної пам'яті як проєктне обмеження; автоматизація резервування є окремою реалізацією.

Три замки на дверях, і жоден не замкнений

Знайшла й виправила три механізми захисту SSH від перебору паролів, які ніколи не працювали: jail блокування, що стежив за портом 22, тоді як демон слухав порт 1986; обмеження частоти, яке перекривало ширше правило, розташоване вище; і дію блокування, бінарний файл якої так і не знаходився, тож жодне блокування жодного разу не спрацювало.

Посилила SSH до 24 перевірених директив

Посилила захист SSH: 24 директиви, кожну з яких перевіряє assert, і триетапна валідація — файл-кандидат, зібрана конфігурація, а потім зворотне зчитування самим демоном, — після того як таке зчитування виявило, що запущений сервер мовчки перевизначав дві з двадцяти чотирьох.

Довела, що блокування вторгнень працює щоразу

Довела працездатність усього ланцюжка блокування вторгнень під час кожного запуску посилення захисту: блокувала зарезервовану тестову адресу, зчитувала створене правило ядра й знімала блокування в гарантованому блоці очищення, тож jail, який перестав працювати, обриває запуск помилкою, а не звітує про справність.

Перевірила правила фаєрвола за позицією

Перевірила правила фаєрвола за позицією, а не лише за наявністю, зчитуючи пронумерований перелік правил і чинний ланцюжок фільтрації пакетів, бо наявне правило ще не означає, що до нього доходить хоч один пакет.

Автоматизація резервування з явним способом відновлення

Реалізувала автоматизацію резервування restic з політикою зберігання, перевіркою цілісності та щомісячним тестом відновлення; задокументувала активацію сховища й перевірку справжнього відновлення як окремі операційні кроки.

Побудувала моніторинг хоста з dead man's switch

Побудувала моніторинг за принципом dead man's switch, який надсилає пінг, лише поки пам'ять і диск у нормі, тож хост у погіршеному стані здіймає тривогу своїм мовчанням, — і виявила шість імен змінних зі словом «free» там, де перевірка правильно вимірювала «available»: на машині з 464 МБ ці значення різняться на порядок.

Домоглася, щоб check mode в Ansible казав правду

Домоглася, щоб check mode показував правду на всій платформі, після того як знайшла шість перевірок, що ухвалювали рішення за значенням, якого хост ніколи не повертав: модуль command в Ansible у режимі --check звітує про успіх, узагалі не виконуючи команду.

Посилила ізоляцію systemd для кожного юніта

Знизила оцінку відкритості (exposure) у пісочниці systemd для кожного юніта, який встановлює платформа: служба dead man's switch — з 9,6 UNSAFE до 1,5, відкритий в інтернет git-сервер — з 8,3 EXPOSED до 1,5, — і додала перевірку парсером під час converge, після того як знайшла у трьох шаблонах юнітів директиву з орфографічною помилкою, яку systemd мовчки ігнорував.

Побудувала 7 ролей звітності лише для читання

Побудувала сім ролей звітності лише для читання, які описують робочий хост у Markdown — факти, доступ, git, метрики, трафік, безпека та інвентар провайдера, — за правилом: не виводиться жодне число, якого запуск не виміряв.

Ваш код на вашому власному сервері

Розгорнула власний git-сервер компанії на Soft Serve, приватний за замовчуванням і без вебпанелі, з портом SSH, прив'язаним до loopback за проміжним хостом (jump host), і зробила цільову сторінку перед ним артефактом збірки основного сайту, а не копією, яку підтримують вручну.

Запустила контейнери на Podman і Quadlet

Розгорнула контейнерну платформу на Podman і Quadlet під systemd, а не на Docker, бо типове публікування портів Docker у мережі bridge може обходити правила UFW, — а для розгортань виділила непривілейованого користувача з однією фіксованою командою замість root.

Розділила всі операційні секрети за хостами

Розділила чотири операційні секрети так, щоб кожен хост мав власні, з'ясувавши, що два хости зі спільним dead man's switch сповіщають про збої рідше, ніж два окремі перемикачі, а не частіше, і що спільна парольна фраза резервних копій може відкрити доступ до копій обох хостів навіть в окремих репозиторіях, якщо хтось отримає її та зможе прочитати зашифровані дані й файли ключів обох репозиторіїв.

Перевірила CI-хук git-сервера, знайшла 2 дефекти

Відрепетирувала CI-хук на боці git-сервера й знайшла два дефекти, яких не виявити читанням файлу: запасний варіант, що передавав на вхід хука аргумент, який неможливо розв'язати, і власну змінну середовища git, яка потрапляла слідом за гейтом у checkout і робила 19 тестів червоними.

Опублікувала читабельне onion-дзеркало в Tor

Опублікувала сайт як onion-дзеркало в Tor за читабельною адресою, що починається з engineer, з ключем, згенерованим поза хостом, тож він ніколи не потрапив ні до цього репозиторію, ні на ноутбук, і залишила цей вхід єдиним із чотирьох, відвідування якого ніколи не рахуються.

Спрямуйте інженерні зусилля

Опишіть шлях розгортання, експлуатаційні ризики та чергову команду. Ми можемо визначити зміну з перевірками, шляхом відновлення та чіткою відповідальністю.

Дивитися, скільки це коштує Зв'язатися з нами →

Розгляньте тему у своєму контексті

Розкажіть, як «Linux та сервери» стосується вашої системи. Ми допоможемо обрати відповідну послугу та визначити обсяг роботи.

Переглянути наші послуги

welcome@engineer.company +45 50 10 61 84 Копенгаген, Данія

Візьміть цю сторінку із собою

Візитівка

Дбайливо зроблено з любов'ю — D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Авторське право © 2025 – дотепер · Engineer ApS · Усі права захищено.

Надруковано з https://platform.engineer.company/uk/categories/linux/