Engineer

{ Infrastructure as Code }

Сервери, які можна відтворити з версійованої конфігурації

Уся конфігурація записана у файлах, тож її можна відтворити точно, і вона не живе лише в чиїйсь голові.

Ще визначаєтесь, що саме вам потрібно?

Розкажіть, що йде не так або що ви намагаєтесь побудувати. Ми скажемо, що для цього потрібно, скільки це приблизно коштуватиме і чи ми ті люди, які вам потрібні.

Від 1 400 DKK за годину, без ПДВ — і дешевша година на довшому замовленні.

Зв'язатися з нами Дивитися, скільки це коштує

Вибрані роботи, що демонструють цю послугу.

Перетворіть послугу на результат

Нехай «Infrastructure as Code» слугує меті бізнесу. Ми можемо пов'язати технічну роботу з користувачами, експлуатацією та рішеннями, які вона має підтримувати. Уся конфігурація записана у файлах, тож її можна відтворити точно, і вона не живе лише в чиїйсь голові.

Зв'язатися з нами

Розгляньте відповідні роботи

Використайте роботи на цій сторінці, щоб оцінити наш підхід. Запитайте про компроміси та передумови кожного результату. Написала 1 000 правил RBAC для Linux-серверів — 1,000 правил доступу написано, щоб усунути вразливість

Написала 1 000 правил RBAC для Linux-серверів Зв'язатися з нами →

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.

Розгорнула інфраструктуру Azure як код

Розгорнула інфраструктуру Azure як код за допомогою Bicep — Container Apps, PostgreSQL Flexible Server, Front Door/WAF та мережу — у середовищах development, staging і production.

Налаштувала резервні копії та відновлення як код

Налаштувала зберігання резервних копій бази даних у вигляді інфраструктури як коду (infrastructure-as-code), провела аудит готовності до відновлення й задокументувала процедуру відновлення — чітко назвавши наявні прогалини, а не залишивши їх на час інциденту.

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.

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.

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.

Додала секундну попередню перевірку секретів

Додала попередній плей (preflight), який приблизно за секунду проганяє той самий код, що й converge, на локальних секретах оператора, — після того як продакшн-запуск обірвався на дев'ятому завданні, застосувавшись лише наполовину, а налаштування swap уже було записано на робочий хост.

Описала зону DNS із 20 записів як код

Налаштувала декларативне узгодження зони DNS із 20 записів з API Cloudflare, з окремими точками входу для аудиту та експорту в BIND, а побудувавши проксі CDN, знову вимкнула його з міркувань приватності.

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.

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.

Підготувала другий сервер із коду

Автоматизувала підготовку (provisioning) другого сервера в другого хмарного провайдера, створюючи фаєрвол раніше за машину, щоб вона від самого початку була за ним, а фаєрволи обох провайдерів налаштовуються безпосередньо через їхні REST API, без сторонньої колекції.

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.

Потрібна послуга: Infrastructure as Code?

Оцініть відповідний досвід за пов’язаними роботами. Розкажіть, що потрібно створити; обговоримо обсяг, підхід до виконання та передачу результатів.

Зв'язатися з нами Повне портфоліо

Визначте обсяг до замовлення

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 →

Обговоріть свої вимоги

Розкажіть, що вам потрібно за напрямом «Infrastructure as Code», зокрема про строки та команду, яка підтримуватиме рішення. Обговоримо практичний шлях уперед.

Зв'язатися з нами

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://engineer.company/uk/services/infrastructure-as-code/