Engineer Plataformas

Portafolio

Servicios de ingeniería con ejemplos de trabajos realizados.

Engineer ApS se registró en 2025. La mayor parte del trabajo que se muestra aquí lo realizó antes nuestra fundadora dentro de equipos de clientes; una parte la ha entregado la propia empresa y otra se ha realizado después por cuenta ajena. Las mismas manos y los mismos estándares: indicamos qué corresponde a cada contexto en lugar de confundirlos.

Encuentre experiencia que merezca aplicar

Utilice esta trayectoria para encontrar experiencia relevante para su proyecto.

Explore nuestros servicios Póngase en contacto →

Lea las decisiones detrás de la trayectoria

Las entradas describen trabajos entregados en distintos servicios. Lea una reseña para conocer la situación y las decisiones que los sustentan.

Casos prácticos Póngase en contacto →

Explorar por categoría

INGENIERÍA DE SOFTWARE

Ingeniería de backend · Ingeniería de frontend · Desarrollo integral · API e integración · Desarrollo web · Pruebas y control de calidad · Optimización del rendimiento · Python · Internacionalización

DATOS Y BASES DE DATOS

Bases de datos · PostgreSQL · SQL · Ingeniería de datos · Flujos de datos (ETL/ELT) · Análisis de datos · Gobernanza de datos · GIS / Información geoespacial

NUBE, INFRAESTRUCTURA Y DEVOPS

Nube · Contenedores (Docker/Kubernetes) · DevOps

Automatización y CI/CD
Infraestructura
Linux y servidores
Administración de sistemas
Redes y VPN
Monitorización y observabilidad
Fiabilidad y copias de seguridad
Seguridad
Arquitectura de plataformas
Migraciones y modernización

CONSULTORÍA Y LIDERAZGO

Dirección técnica
Dirección de equipos
Metodologías ágiles y Scrum
Gestión de proyectos
Producto y requisitos
Partes interesadas e informes
Documentación
Arquitectura de soluciones
Marca y marketing

Todo lo que sigue **se ha entregado** —
no son propuestas ni planes.

Cada entrada indica qué cambió el trabajo y enlaza los servicios que demuestra.

Qué ofrecemos Cuánto cuesta →

{ Nube, infraestructura y DevOps }

Infraestructura en la nube y migración

El lugar donde funciona el software

Servidores dimensionados para el uso real, trasladados a la nube
o fuera de ella, con la factura mensual calculada de antemano.

Segundo servidor aprovisionado desde código

Automaticé el aprovisionamiento de un segundo servidor en un segundo proveedor de nube y creé el cortafuegos antes que la máquina para que naciera protegida, con los cortafuegos de ambos proveedores configurados directamente mediante sus API REST para evitar una colección de terceros.

Contenedores con Podman y Quadlet

Desplegué una plataforma de contenedores con Podman y Quadlet bajo systemd en lugar de Docker, porque la publicación predeterminada de puertos en redes bridge de Docker puede eludir las reglas UFW, y asigné a los despliegues un usuario sin privilegios con una única orden fija, en vez de root.

Servicios públicos en un servidor pequeño

Alojé los servicios públicos de la empresa en un host de 512 MB y un solo núcleo — una forja git, un servidor web para siete dominios, Tor, dos servidores de protocolos alternativos y bloqueo de intrusiones — tomando los 464 MB de memoria utilizable como restricción de diseño; la automatización de copias de seguridad es una implementación aparte.

Infraestructura definida en un repositorio

Desarrollé la infraestructura de la empresa mediante 19 playbooks de Ansible y 34 roles en 12.065 líneas de YAML, haciendo converger un servidor activo hacia un estado declarado con todas las ejecuciones idempotentes.

Infraestructura principal creada desde cero

Como una de las primeras incorporaciones, diseñé y construí desde cero toda la infraestructura principal y los procesos de soporte de una startup SaaS de energía verde, sentando las bases para un crecimiento rápido.

Despliegues completos de la plataforma en Azure

Asumí los despliegues completos de la plataforma en Azure y gestioné las versiones en desarrollo, preproducción y producción.

Infraestructura Azure como código

Aprovisioné infraestructura Azure como código con Bicep — Container Apps, PostgreSQL Flexible Server, Front Door/WAF y redes — en los entornos de desarrollo, preproducción y producción.

Infraestructura simplificada fuera de AWS

Reorganicé la infraestructura de un cliente internacional, eliminando servicios sin uso, ajustando la capacidad y trasladando cargas fuera de AWS. La fundadora informa de una reducción del gasto en la nube por un factor de 10, sin pérdida de productividad.

20 aplicaciones Docker en Google Cloud y AWS

Desplegué y mantuve 20 aplicaciones en contenedores Docker, resolví incidencias con Podman y Kubernetes y administré aplicaciones basadas en R en Google Cloud y AWS.

{ Nube, infraestructura y DevOps }

Infraestructura como código

Servidores que se pueden reconstruir a partir de una configuración versionada

Toda la configuración escrita en archivos para poder recrearla con exactitud y evitar que exista solo en la memoria de una persona.

Espejo onion de Tor con una dirección legible

Publiqué el sitio como espejo onion de Tor en una dirección legible que comienza por engineer, con la clave generada fuera del servidor para que nunca llegara a este repositorio ni a un portátil, y lo mantuve como la única de las cuatro puertas de entrada que nunca se contabiliza.

Ensayo del hook CI: 2 defectos detectados

Ensayé el hook CI del servidor de la forja y encontré dos defectos imposibles de alcanzar leyendo el archivo: una alternativa que introducía un argumento irresoluble en la entrada del hook y una variable de entorno de git que acompañaba al control hasta el checkout y hacía fallar 19 pruebas.

Secretos operativos separados por servidor

Separé los cuatro secretos operativos de cada servidor tras constatar que dos servidores que comparten una señal de actividad alertan menos que dos señales independientes, y que una contraseña compartida de copias de seguridad puede exponer las copias de ambos servidores, incluso en repositorios separados, si alguien la obtiene y puede leer los datos cifrados y los archivos de claves de ambos repositorios.

Su código en su propio servidor

Desplegué la forja git propia de la empresa con Soft Serve, privada por defecto, sin panel web y con el puerto SSH limitado a loopback detrás de un servidor de salto, e hice de su página de entrada un artefacto de compilación del sitio principal en lugar de una copia mantenida a mano.

Menor exposición en cada unidad systemd

Reduje la exposición de aislamiento systemd en todas las unidades que instala la plataforma — un servicio de monitorización por ausencia de señal pasó de 9,6 UNSAFE a 1,5 y una forja git expuesta a Internet de 8,3 EXPOSED a 1,5 — y añadí una comprobación del analizador durante la convergencia tras encontrar una directiva mal escrita que se ignoraba silenciosamente en tres plantillas de unidades.

Zona DNS de 20 registros como código

Concilié declarativamente una zona DNS de 20 registros con la API de Cloudflare mediante puntos de entrada separados para auditoría y exportación BIND, y desactivé el proxy CDN por motivos de privacidad después de implementarlo.

Comprobación de secretos en un segundo

Añadí una comprobación previa que ejecuta en aproximadamente un segundo el mismo código de convergencia sobre secretos locales del operador, después de que una ejecución de producción parcialmente aplicada fallara en su novena tarea con los ajustes de intercambio ya escritos en el servidor activo.

Modo de comprobación Ansible fiel a la realidad

Hice que el modo de comprobación reflejara la realidad en toda la plataforma tras detectar seis sondeos que decidían a partir de un valor que el servidor nunca había proporcionado, porque el módulo command de Ansible informa de éxito con --check aunque omita la orden por completo.

Monitorización por ausencia de señal

Desarrollé una monitorización por ausencia de señal que solo avisa de actividad mientras la memoria y el disco están en buen estado, de modo que un servidor degradado genera una alerta al quedar en silencio; además detecté seis variables llamadas «free» cuando el control medía correctamente «available», valores separados por un orden de magnitud en una máquina de 464 MB.

Automatización de copias con un método explícito de restauración

Implementé automatización de copias con restic, retención, comprobaciones de integridad y un flujo mensual de prueba de restauración; documenté la activación del almacenamiento y la verificación de restauración real como pasos operativos separados.

SSH reforzado con 24 directivas verificadas

Reforcé SSH con 24 directivas verificadas mediante validación en tres etapas — archivo candidato, configuración ensamblada y lectura de la configuración por el propio demonio —, después de que esta última detectara que el servidor en ejecución anulaba silenciosamente dos de las veinticuatro.

Copias y recuperación ante desastres como código

Configuré la retención de las copias de seguridad de la base de datos como infraestructura como código, audité la capacidad de recuperación y documenté el procedimiento de restauración, dejando explícitas las carencias pendientes para evitar descubrirlas durante un incidente.

1.000 reglas RBAC para servidores Linux

Reforcé la seguridad de los datos mediante 1.000 reglas RBAC para desarrolladores, instancias de aplicaciones, PostgreSQL, MS SQL y otros servidores Linux, evitando accesos no autorizados y documentando la configuración mediante automatización con Ansible.

Este servicio forma parte de trabajos que figuran en otros lugares de esta página:

Segundo servidor aprovisionado desde código · Servicios públicos en un servidor pequeño · Infraestructura definida en un repositorio · Infraestructura Azure como código

{ Nube, infraestructura y DevOps }

DevOps y automatización CI/CD

Publicar cambios sin sobresaltos

Cambios probados, publicados y reversibles, para que lleguen a sus clientes un martes por la tarde en vez de un viernes por la noche.

Un criterio de historial para 4 repositorios

Apliqué un único criterio de historial a cuatro repositorios — commits convencionales, sin emojis ni líneas de atribución y con un hook de mensajes de commit —, junto con 48 documentos de instrucciones que regulan la forma de trabajar.

Conocer la licencia de todo lo que se publica

Audité la compatibilidad de licencias de 644 crates Rust en cada compilación y probé que el control se activa al cambiar la licencia de una crate y al modificar la revisión fijada del núcleo sin regenerar los datos.

La mitad sin utilizar del núcleo, implementada

Implementé la mitad del núcleo de mensajería que la aplicación no utilizaba — transferencia de copias de seguridad, mensajes temporales, edición y reenvío de mensajes, invitaciones verificadas, proxies y política de cifrado —, ejecutando cada prueba contra la biblioteca real, sin simulaciones.

Pruebas reducidas a 8,9 segundos

Pasé de seis pruebas que superaban un límite de sesenta segundos a 135 pruebas satisfactorias en 8,9 segundos al analizar el hilo principal y eliminar las dos llamadas en las que permanecía durante 3.989 de 4.017 muestras.

Analizador que verifica cada llamada FFI

Escribí un analizador que lee la cabecera C real de 7.308 líneas y verifica cada punto de llamada, cada constante de enumeración y que toda clase propietaria de un puntero sea final, después de que una cabecera provisional escrita a mano permitiera compilar y enlazar llamadas a tres funciones eliminadas que causaban fallos al ejecutarse.

Diferencias detectadas entre hooks y controles

Detecté que los hooks de commit y el control de calidad ejecutaban comprobaciones distintas aunque un documento prometía que eran iguales, al comparar ambas listas en una prueba: las cinco que solo se ejecutaban a mano eran las que revisaban la redacción del CV.

Pruebas para los propios verificadores

Escribí pruebas para los propios verificadores tras constatar que un verificador que solo recibe entradas correctas puede acabar informando de éxito por no haber leído nada; introduje una falta de ortografía para confirmar su detección y obtuve un intervalo de identificadores de la base de datos en vez de fijarlo en la prueba.

Cada regla del analizador tratada como error

Activé como errores todas las reglas del analizador Python y resolví 1.815 hallazgos hasta llegar a cero, con un motivo escrito para cada una de las escasas excepciones y dos de ellas respaldadas por una comprobación, además de un comentario.

Generador: 981 pruebas y 92 % de cobertura

Mantuve el generador bajo 981 casos de prueba con un mínimo del 92 % de cobertura de ramas y las advertencias tratadas como errores, y verifiqué la idempotencia ejecutando dos veces todo el flujo de construcción desde un archivo vacío y exigiendo que la segunda pasada no cambiara nada.

Control de calidad de 10.242 líneas analizado

Apliqué un formateador y un analizador a 10.242 líneas de JavaScript del control de calidad tras constatar que eran el mayor bloque de código del repositorio y el único que ninguna herramienta revisaba; corregí 13 hallazgos sin suprimir ninguno.

Sitemap corregido: fechas que eran iguales

Corregí un sitemap en el que 172 de 176 URL compartían una misma fecha de modificación, tomando la fecha del historial git tras constatar que la exportación reescribe todos los archivos en cada ejecución.

Espejos del sitio en Gemini y Gopher

Construí salidas Gemini y Gopher a partir de la misma fuente del sitio web; el logro registra 777 documentos en cada formato en aquella etapa, con cero bytes de HTML modificados al añadir los formatos.

Una red de seguridad que ya no cuesta media hora

Reduje el tiempo del control de calidad del sitio basado en navegador de 1.636 segundos a 615 al ordenar las comprobaciones de mayor a menor duración y ejecutarlas en un grupo limitado a cuatro trabajadores, tras medir que el orden alfabético requería 320 segundos frente a 224.

Control de commits con 22 analizadores

Desarrollé un control obligatorio de commits con 22 analizadores de una línea y cinco que requieren una explicación, sin nivel de advertencias ni supresiones en línea permitidas, que cubre HTML, CSS, JavaScript, Python, YAML, Markdown, Shell, enlaces, ortografía, secretos y tipografía.

Análisis continuo de seguridad

Establecí un programa continuo de seguridad — análisis de código, DAST, comprobación de dependencias y vulnerabilidades, generación de SBOM, detección de secretos y acciones fijadas por SHA —, junto con 21 auditorías de seguridad documentadas.

1.022 migraciones con control de coherencia

Mantuve la coherencia del esquema durante 1.022 migraciones con un control CI que crea la base de datos por dos vías — una instalación nueva y una instalación seguida de todas las migraciones — y falla si los resultados difieren.

Monitorización de errores y trazas propias

Desarrollé internamente la monitorización de errores y las trazas OpenTelemetry, en lugar de adquirirlas — saneamiento de cargas, detección de picos y regresiones, resolución de símbolos y una señal de actividad sintética —, con 11 vistas para operaciones.

Motor de 268 controles para el repositorio

Desarrollé el motor de comprobaciones del repositorio — 268 controles de commit registrados, 277 reglas de análisis estático y 15 reglas ESLint propias —, además de 146 pruebas de los propios controles, para que la compilación haga cumplir el criterio de calidad.

Pruebas para Go, SQL y navegador

Desarrollé una batería de pruebas automatizadas por capas — 981 pruebas Go, 543 especificaciones de interfaz y navegador y 494 pruebas de comportamiento SQL — con pruebas de mutación, pruebas basadas en propiedades y un control obligatorio de accesibilidad.

Colaboración del equipo automatizada

Automaticé la colaboración del equipo, la gestión de contraseñas, tareas y tiempo, y desarrollé un sistema semiautomático de presentación de proyectos que elevó la productividad del equipo.

Cero advertencias en seis lenguajes

Fijé un criterio de calidad de cero advertencias en seis lenguajes — Go, TypeScript, SQL, Python, Shell y Markdown — aplicado mediante hooks de pre-commit.

578 tareas de automatización go-task

Redacté 578 tareas de automatización go-task para los modos de desarrollo nativo, Docker y HTTPS, así como para análisis estático, pruebas, bases de datos y despliegue.

Flujos CI/CD en GitHub Actions

Desarrollé flujos CI/CD en GitHub Actions con una imagen de producción distroless para la interfaz y promoción entre varios entornos.

Contrato de API generado desde la base de datos

Generé el contrato de la API desde la base de datos hacia las capas externas — OpenAPI, un cliente TypeScript tipado de 44.076 líneas, 61 manejadores simulados y los límites que aplica la interfaz — con una comprobación en cada paso que falla ante cualquier divergencia.

Datos automatizados con Python y PL/pgSQL

Automaticé tareas de procesamiento de datos con scripts de Shell, PL/pgSQL, Python y Transact-SQL, mejorando la productividad y la eficiencia.

4.000 horas ahorradas con CI/CD GitHub y GitLab

Simplifiqué los procesos de análisis de datos y desarrollo de software y ahorré 4.000 horas mediante prácticas CI/CD con GitHub, GitLab, Bash y Python.

CI/CD optimizado: 4.000 horas ahorradas

Simplifiqué los procesos CI/CD y ahorré 4.000 horas mediante la automatización de los flujos de desarrollo de software.

SSL/TLS automático para 100 aplicaciones Docker

Automaticé la creación de certificados SSL/TLS para 100 aplicaciones Docker, manteniendo conexiones seguras entre servidores Ubuntu Linux.

Procesos renovados: 8.000 horas ahorradas

Reorganicé los procesos internos y ahorré 8.000 horas mediante mejoras en la arquitectura del software, los sistemas y la planificación.

Menor riesgo con Grafana y Prometheus

Reduje los riesgos operativos mediante un panel de monitorización con Grafana y Prometheus, mejorando la fiabilidad del sistema.

100 copias críticas automatizadas con Barman

Automaticé 100 copias de seguridad de datos críticos con Barman, Google Cloud, Bash y Python, preservando la integridad de los datos entre bases de datos.

Entrega automática de 20 flujos de datos GIS

Automaticé la entrega de 20 flujos de datos GIS y procesos ETL de datos de aplicaciones, simplificando la automatización de la infraestructura y la elaboración de informes.

Despliegue GIS SaaS con GitHub Actions

Automaticé el despliegue de una aplicación GIS SaaS, el procesamiento de datos y el sistema de informes con GitHub Actions CI/CD, Python, Bash y SQL.

Este servicio forma parte de trabajos que figuran en otros lugares de esta página:

Ensayo del hook CI: 2 defectos detectados · Contenedores con Podman y Quadlet ·

Su código en su propio servidor · Comprobación de secretos en un segundo ·

Modo de comprobación Ansible fiel a la realidad · Infraestructura definida en un repositorio ·

Infraestructura principal creada desde cero · Despliegues completos de la plataforma en Azure ·

Infraestructura simplificada fuera de AWS · 20 aplicaciones Docker en Google Cloud y AWS

{ Nube, infraestructura y DevOps }

Contenedores y orquestación

Software que funciona igual en todas partes

Software empaquetado para comportarse igual en un portátil, un servidor de pruebas y producción, y ampliar su capacidad cuando aumente la actividad.

Este servicio forma parte de trabajos que figuran en otros lugares de esta página:
Contenedores con Podman y Quadlet · Menor exposición en cada unidad systemd · Flujos CI/CD en GitHub Actions · SSL/TLS automático para 100 aplicaciones Docker · 20 aplicaciones Docker en Google Cloud y AWS

{ Nube, infraestructura y DevOps }

Fiabilidad de sistemas y monitorización

Saberlo antes que sus clientes

Monitorización y alertas que avisan cuando un fallo aún es pequeño, y cifras que muestran con qué frecuencia se mantiene así.

Seis recolectores declaran lo que omitieron

Hice que seis recolectores de informes declararan la parte de sus entradas que nunca habían leído — archivos no abiertos, sondeos no instalados y líneas del diario no reconocidas — y ajusté los umbrales a 116 muestras medidas tras constatar que una recolección ciega y un servidor sano generan la misma página.

13 direcciones medidas sin rastreo

Medí trece direcciones únicamente a partir del registro de accesos del servidor web — sin cookies, rastreadores ni servicios de terceros —, enmascarando la red del cliente al escribir cada línea e incorporando el resultado a un archivo permanente de 21 recuentos diarios y 20 facetas mensuales.

Un flujo de eventos limitado para una aplicación macOS

Limité los búferes de eventos y la generación de registros tras un incidente de macOS documentado con un crecimiento de 41 MB por segundo y 111 GB de páginas comprimidas en una máquina de 36 GB; una ejecución completa de pruebas documentada redujo las líneas de registro del núcleo de 610.996 a 1.411.

7 roles de informes de solo lectura

Desarrollé siete roles de informes de solo lectura que representan un servidor activo en Markdown — inventario técnico, acceso, git, métricas, tráfico, seguridad e inventario de proveedores — bajo la regla de no imprimir ninguna cifra que la ejecución no haya medido.

Bloqueo de intrusiones probado en cada ejecución

Verifiqué de extremo a extremo el bloqueo de intrusiones en cada ejecución de refuerzo, bloqueando una dirección reservada para pruebas, comprobando la regla resultante del núcleo y retirando el bloqueo en un bloque de limpieza garantizado, para que una jail averiada haga fallar la ejecución en vez de presentarse como operativa.

Tres cerraduras en la puerta, ninguna cerrada

Detecté y corregí tres protecciones contra fuerza bruta SSH que nunca habían funcionado: una jail de bloqueo que vigilaba el puerto 22 mientras el demonio escuchaba en el 1986, un límite de tasa oculto bajo una regla más amplia y una acción de bloqueo cuyo binario nunca se encontraba, por lo que no se había aplicado ningún bloqueo.

Límites de tasa y protección contra abusos

Desarrollé controles contra abusos que deniegan el acceso ante fallos — 22 limitadores de tasa con Redis, Cloudflare Turnstile, idempotencia de solicitudes y bloqueo del origen — para rechazar bots y avalanchas de solicitudes sin confiar en quienes las envían.

Trabajo lento trasladado a una cola River

Saqué las tareas lentas del procesamiento de solicitudes y las trasladé a una cola River — 15 módulos de trabajadores, 8 tareas programadas y 20 tareas pg_cron — para responder a las solicitudes mientras el trabajo continúa en segundo plano.

Infraestructura aún en servicio años después

Planifiqué e implementé nuevas funciones de infraestructura para sistemas internos y externos, con soluciones que siguieron funcionando años después con cambios mínimos.

Transmisión IPTV ininterrumpida

Mantuve la entrega ininterrumpida de señales IPTV entre proveedores y clientes, monitorizando y manteniendo la red de distribución y la telefonía IP las 24 horas.

Soporte 24/7 para unos 1.000 servidores IPTV/OTT

Presté soporte de infraestructura 24/7 a una plataforma de distribución IPTV/OTT, administrando unos 1.000 servidores y sistemas de los clientes para usuarios de China, Estados Unidos y Alemania.

Correo con conmutación y registro de entregas

Desarrollé el correo electrónico como capacidad de la plataforma — tres proveedores con conmutación por error, webhooks de entrega, registros de envío y entrega, plantillas y campañas — con una comprobación de arranque que impide iniciar el sistema sin un proveedor.

Un producto cartográfico de principio a fin

Diseñé la arquitectura, desarrollé y mantuve infraestructura, procesamiento de datos y una aplicación de mapas durante 2 años, cubriendo tanto entregas de software como mantenimiento operativo.

40 sitios web con Apache y Nginx

Administré 40 sitios web en servidores de alojamiento Ubuntu Linux con Apache y Nginx, manteniendo una disponibilidad y un rendimiento elevados.

10 servidores PostgreSQL y MS SQL en Ubuntu

Diseñé, desplegué y mantuve 10 servidores PostgreSQL y MS SQL en VPS con Ubuntu Linux, para mantener un rendimiento y una fiabilidad adecuados.

Primer panel de observabilidad de la empresa

Desarrollé y puse en marcha el primer panel de observabilidad de la empresa, con información del rendimiento del sistema en tiempo real y visualizaciones de datos en el televisor grande de la oficina.

Este servicio forma parte de trabajos que figuran en otros lugares de esta página:

Secretos operativos separados por servidor · Modo de comprobación Ansible fiel a la realidad · Monitorización por ausencia de señal · Automatización de copias con un método explícito de restauración · Monitorización de errores y trazas propias · Despliegues completos de la plataforma en Azure · Menor riesgo con Grafana y Prometheus

{ Nube, infraestructura y DevOps }

Copias de seguridad y recuperación ante desastres

Recuperarlo todo después de un mal día

Copias de seguridad que se prueban en vez de darse por buenas y un plan escrito para la mañana en que un servidor no vuelva a encenderse.

Administración de 30 VPS Ubuntu Linux

Administré 30 instancias VPS con Ubuntu Linux, implementé estrategias de recuperación ante desastres y mantuve configuraciones de red adecuadas.

Este servicio forma parte de trabajos que figuran en otros lugares de esta página:

Secretos operativos separados por servidor · Automatización de copias con un método explícito de restauración · Copias y recuperación ante desastres como código · 100 copias críticas automatizadas con Barman

{ Nube, infraestructura y DevOps }

Administración de sistemas

Mantenerlo todo en funcionamiento

El cuidado diario de servidores, correo, cuentas y actualizaciones para que los servicios de los que depende todo el mundo sigan funcionando sin sorpresas.

Reglas de cortafuegos verificadas por posición

Verifiqué las reglas del cortafuegos por su posición, además de por su existencia, leyendo la lista numerada de reglas y la cadena activa del filtro de paquetes, porque una regla presente puede no recibir nunca un paquete.

Gestor de contraseñas para toda la empresa

Integré un sistema de gestión de contraseñas para toda la empresa, reforzando la seguridad y simplificando el control de accesos.

Administración de 100 servidores Bare Bone

Administré 100 servidores Bare Bone, redes físicas y sistemas de telefonía IP, proporcionando una infraestructura sólida para el crecimiento de la empresa.

Configuración y despliegue de 1.000 routers Wi-Fi

Configuré y desplegué 1.000 routers Wi-Fi, mejorando el acceso a la red y su rendimiento para los clientes.

Administración de infraestructura de red

Administré la infraestructura de red de más de 1.000 servidores, atendiendo al despliegue, la seguridad y la resolución de incidencias de los sistemas.

Gestión de 8 conexiones VPN WireGuard e IPsec

Administré y resolví incidencias en 8 conexiones VPN de WireGuard e IPsec, manteniendo comunicaciones seguras entre Google Cloud y sistemas Linux.

Gestión de accesos con M365 y SSO

Prevení brechas de seguridad al dirigir iniciativas de gestión de accesos con M365, 1Password, Red Hat SSO y OKTA SSO.

Este servicio forma parte de trabajos que figuran en otros lugares de esta página:

Espejo onion de Tor con una dirección legible · Seis recolectores declaran lo que omitieron · Ensayo del hook CI: 2 defectos detectados · Su código en su propio servidor · 7 roles de informes de solo lectura · Menor exposición en cada unidad systemd · Zona DNS de 20 registros como código · Monitorización por ausencia de señal · Automatización de copias con un método explícito de restauración · Bloqueo de intrusiones probado en cada ejecución · SSH reforzado con 24 directivas verificadas · Servicios públicos en un servidor pequeño · Infraestructura definida en un repositorio ·

Infraestructura principal creada desde cero · Infraestructura aún en servicio años después · Transmisión IPTV ininterrumpida · Soporte 24/7 para unos 1.000 servidores IPTV/OTT · 40 sitios web con Apache y Nginx · Administración de 30 VPS Ubuntu Linux · 1.000 reglas RBAC para servidores Linux · 10 servidores PostgreSQL y MS SQL en Ubuntu

{ Nube, infraestructura y DevOps }

Configuración de redes y VPN

Conectar sus oficinas y a su personal

Wi-Fi, cortafuegos y acceso remoto configurados para trabajar desde cualquier lugar y permitir el acceso solo a las personas autorizadas.

Este servicio forma parte de trabajos que figuran en otros lugares de esta página:

Segundo servidor aprovisionado desde código · Zona DNS de 20 registros como código · Reglas de cortafuegos verificadas por posición · Tres cerraduras en la puerta, ninguna cerrada · Transmisión IPTV ininterrumpida · Infraestructura Azure como código · Administración de 100 servidores Bare Bone · Configuración y despliegue de 1.000 routers Wi-Fi · Administración de infraestructura de red · Gestión de 8 conexiones VPN WireGuard e IPsec · Administración de 30 VPS Ubuntu Linux

{ Nube, infraestructura y DevOps }

Seguridad y gestión de accesos

Decidir quién puede acceder a qué

Cuentas, contraseñas y permisos organizados para que cada persona tenga exactamente el acceso que necesita y lo pierda el mismo día que deja la empresa.

Regla de alcance tras una fuga de credenciales

Incorporé al repositorio una regla de alcance tras una reorganización que había introducido el inventario, los permisos de cortafuegos y textos de otra empresa, y mantuve los residuos en cuarentena bajo el detector de secretos, sin excluirlos.

Reclamaciones y decisiones de titularidad

Desarrollé el proceso completo de reclamación de titularidad de empresas: un usuario solicita una empresa, un administrador decide y la aprobación actualiza el grafo de autorizaciones que determina quién puede editar cada recurso.

Stripe y acceso por compras en aplicaciones

Desarrollé la capa de pagos y derechos de acceso — Stripe junto con las compras dentro de las aplicaciones de Apple y Google —, regulando el acceso al directorio, la búsqueda y la exportación mediante un modelo de 11 tablas comprobado en el servidor.

Aplicación reforzada con CSP basada en nonces

Reforcé la aplicación con CSP basada en nonces, HSTS, cookies SameSite, roles de base de datos de mínimo privilegio y nuevas comprobaciones de derechos de acceso en el servidor.

Cambio de contexto entre organizaciones

Diseñé un sistema de cambio de contexto entre organizaciones con localStorage en el cliente y una copia en cookies del servidor, que permite actuar en nombre de las organizaciones administradas y aplica autorizaciones de mínimo privilegio.

Mejor comunicación y colaboración del equipo

Mejoré la comunicación y la colaboración del equipo mediante Slack, Mattermost, 1Password y Jira, ahorrando 8.000 horas de trabajo.

Este servicio forma parte de trabajos que figuran en otros lugares de esta página:

Espejo onion de Tor con una dirección legible · 13 direcciones medidas sin rastreo ·
Conocer la licencia de todo lo que se publica · La mitad sin utilizar del núcleo, implementada ·
Secretos operativos separados por servidor · Segundo servidor aprovisionado desde código ·
Contenedores con Podman y Quadlet · Menor exposición en cada unidad systemd ·
Zona DNS de 20 registros como código · Comprobación de secretos en un segundo ·
Reglas de cortafuegos verificadas por posición · Bloqueo de intrusiones probado en cada ejecución ·
SSH reforzado con 24 directivas verificadas · Tres cerraduras en la puerta, ninguna cerrada ·
Análisis continuo de seguridad · Límites de tasa y protección contra abusos ·
Gestor de contraseñas para toda la empresa · Colaboración del equipo automatizada ·
Infraestructura Azure como código · Correo con conmutación y registro de entregas ·
SSL/TLS automático para 100 aplicaciones Docker · Administración de infraestructura de red ·
Gestión de 8 conexiones VPN WireGuard e IPsec · Gestión de accesos con M365 y SSO ·
1.000 reglas RBAC para servidores Linux

{ Nube, infraestructura y DevOps }

Seguridad de agentes de IA

Que su IA actúe solo donde le corresponde

El acceso de cada agente definido y limitado a lo necesario, cada acción registrada donde el agente no puede alterarla y cada protección comprobada en lugar de darse por segura.

Este servicio forma parte de trabajos que figuran en otros lugares de esta página:

Secretos operativos separados por servidor · Regla de alcance tras una fuga de credenciales · Comprobación de secretos en un segundo · Bloqueo de intrusiones probado en cada ejecución · Tres cerraduras en la puerta, ninguna cerrada · Análisis continuo de seguridad · Límites de tasa y protección contra abusos · Gestión de accesos con M365 y SSO · 1.000 reglas RBAC para servidores Linux

Compare su punto de partida

Elija un trabajo comparable e identifique qué cambia en su propia situación.

Vea cuánto cuesta Póngase en contacto →

Hablemos de un enfoque relevante

Pídanos que convirtamos la experiencia relevante en un enfoque que pueda evaluar.

Explore nuestros servicios

welcome@engineer.company +45 50 10 61 84 Copenhague, Dinamarca

Llévese esta página

Tarjeta de contacto

Creado con esmero y cariño por D.

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

Derechos de autor © 2025 – actualidad · Engineer ApS · Todos los derechos reservados.

Impreso desde https://platform.engineer.company/es/portfolio/