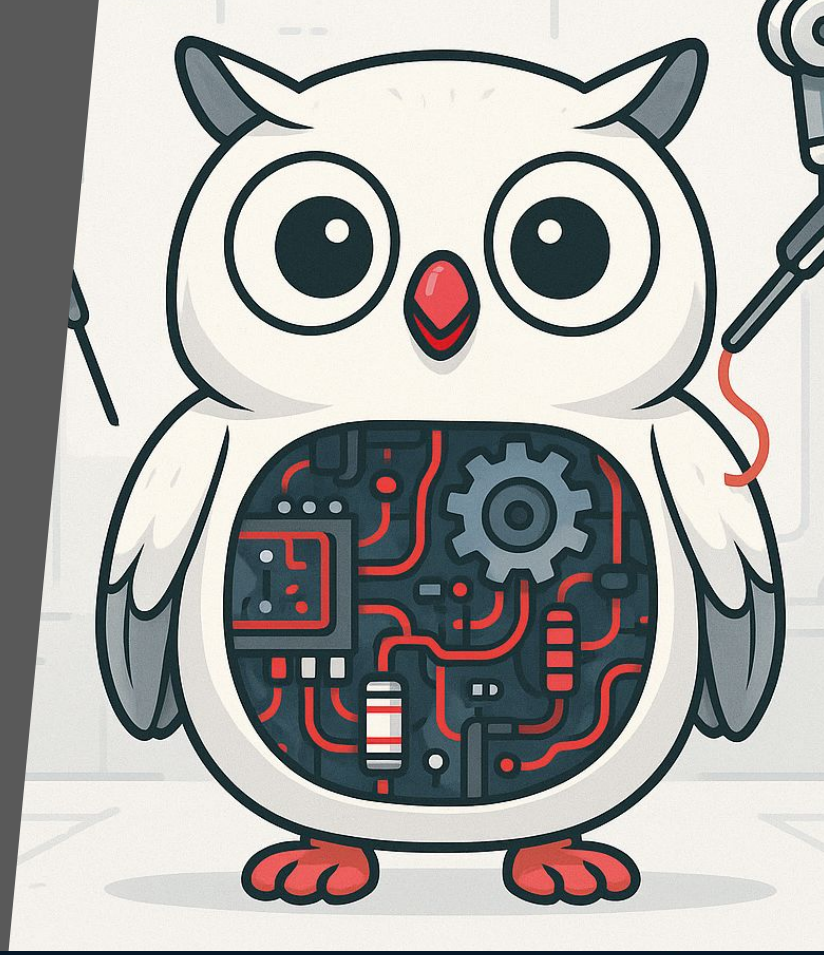
Vulseek Internals

Desafíos, Arquitectura y Aprendizajes

Fabian Martinez Portantier
CTO & Co-Founder

Vulseek
By Securetia



H4CK3D Security Conference

Securetia

UP Universidad de Palermo

Introducción

Objetivo

“Compartir algunas de las cosas que aprendimos al construir Vulseek”

¿Qué es Vulseek?

“SaaS que te permite detectar, priorizar y gestionar vulnerabilidades”

Flujo de trabajo resumido

- Cargás tus activos (al menos tu dominio)
- Identifica activos relacionados (hostnames, direcciones IP, etc)
- Identifica vulnerabilidades
- Prioriza las vulnerabilidades (usando CVSS, EPSS, KEV)
- Identifica qué vulnerabilidades se remediaron

(Por simplicidad dejamos varias cosas afuera)

Promesas

Simple: Brinda información de forma clara y concisa

Contínua: Escanea tu infraestructura los 365 días del año

Rápida: Entrega los primeros resultados en minutos

Timeline muy resumido

Nace Securetia!

Securetia

Presentamos Vulseek!

Vulseek
By Securetia

Seleccionados en múltiples programas de escalabilidad

STARTUP OLÉ

Buenos Aires Ciudad

Ministerio de Desarrollo Productivo Argentina

aws startups

Certificamos ISO 9001!

ISO 9001

20 personas 100 empresas



2014 — 2021 — 2024 — 2025

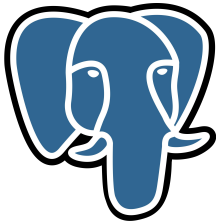
Tecnologías Clave

- Containers
- Python (back)
- Javascript (front)
- PostgreSQL (db)
- AWS (cloud)
- Anthropic (ia)













Servicios de AWS


ALB


Batch


Bedrock


CloudWatch


ECR


ECS


Fargate


RDS


S3



Route 53


SQS


SNS


SES


WAF

Desafíos

Desafíos

1. Tiempo hasta obtener resultados
2. Entregar resultados consistentes

Para próximas charlas o artículos:

- ¿Cómo elegir un stack tecnológico?
- ¿Qué tan crítico es un EndOfLife?
- ¿Cómo seguirle el ritmo a los avances de la IA?

Desafío 1: Tiempo hasta obtener resultados

¿Cuánto tiempo pasa desde que cargás un activo hasta que empezás a ver resultados?

Desafío 1: Tiempo hasta obtener resultados

- Sabemos que tiene que ser rápido (promesa)
- Pero... ¿Qué tan rápido?
- Hay que escanear continuamente (promesa)
- Acá viene el primer trade-off

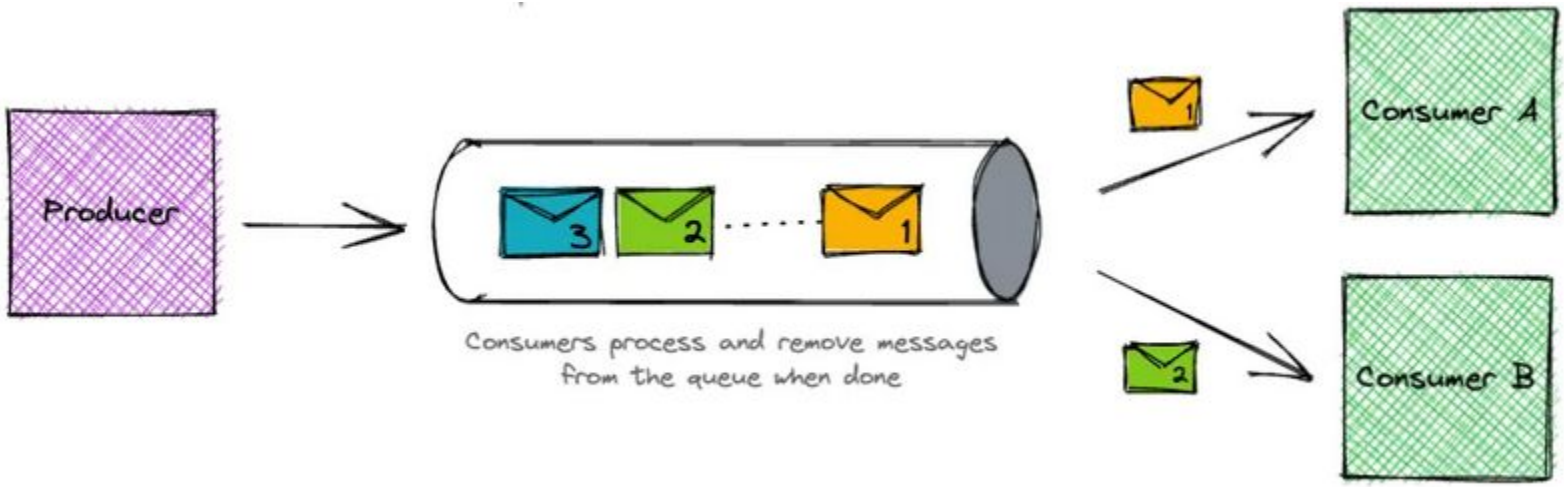
Desafío 1: Tiempo hasta obtener resultados

Tiene que ser una solución:

1. **Que de respuesta rápida** (es el objetivo)
2. **Fácil de implementar y mantener** (somos pocos)
3. **Escalable** (es un SaaS)

Desafío 1: Tiempo hasta obtener resultados

"Usamos colas priorizadas para que los escaneos se ejecuten de forma controlada y escalable"



Desafío 1: Tiempo hasta obtener resultados

1. **Que de respuesta rápida:**
   a. Se priorizan los escaneos nuevos
   b. Se ordenan los pasos para optimizar al máximo
   c. Se entregan resultados parciales
2. **Fácil de implementar:**
   a. Resuelto en AWS con SNS y SQS (pero hay alternativas)
   b. Boto3 (librería de AWS oficial en python)
3. **Escalable:**
   a. Parametrización
   b. Autoscaling

Desafío 1: Tiempo hasta obtener resultados

Gracias a esto, vemos resultados en menos de 5 minutos

(Podría ser en menos, pero, trade-off)

Desafío 1: Tiempo hasta obtener resultados



Misión Cumplida!

Desafío 2: Entregar resultados consistentes

Nos vamos a encontrar con:

A. Diferencias entre un scanner y otro (versión, OS, etc)
B. Problemas en la red (mal funcionamiento)
C. Problemas en la red (demasiado buen funcionamiento)
D. Filtros dinámicos (Firewall/IPS/WAF)

Desafío 2: Entregar resultados consistentes

$ sudo nmap -Av scanme.nmap.org

Desde mi casa:
Nmap done: 1 IP address (1 host up) scanned **in 37.59 seconds**

Desde uno de nuestros scanners:
Nmap done: 1 IP address (1 host up) scanned **in 17.68 seconds**

Relación: 2.12

Desafío 2: Entregar resultados consistentes

$ ping -4 -c 1000 -i 0.01 scanme.nmap.org

Desde mi casa:
1000 packets transmitted, 997 received, 0.3% packet loss, time 13385ms
rtt min/avg/max/mdev = 214.608/**223.485**/247.649/2.942 ms

Desde uno de nuestros scanners:
1000 packets transmitted, 1000 received, 0% packet loss, time 15947ms
rtt min/avg/max/mdev = 0.523/**0.689**/1.373/0.083 ms

Desafío 2: Entregar resultados consistentes

$ sudo nmap **-T2** -Av scanme.nmap.org

Desde mi casa:
Nmap done: 1 IP address (1 host up) scanned **in 477.36 seconds**

Desde uno de nuestros scanners:
Nmap done: 1 IP address (1 host up) scanned **in 441.31 seconds**

Relación: 1.08

Desafío 2: Entregar resultados consistentes

$ ncat -4v scanme.nmap.org 17
Ncat: Version 7.94SVN (https://nmap.org/ncat)

Desde mi casa:
Ncat: Connection refused.
Para Nmap: **closed**

Desde uno de nuestros scanners:
Ncat: Connection failed: TIMEOUT
Para Nmap: **filtered**

<-- - - Estos filtros pueden ser dinámicos!

Desafío 2: Entregar resultados consistentes

Desde mi casa:
T3: Aggressive OS guesses: Linux 2.6.32 or 3.10 (93%)
T2: Aggressive OS guesses: Linux 2.6.32 or 3.10 (93%)

Desde uno de nuestros scanners:
T3: OS details: Linux 4.15 - 5.8
T2: Aggressive OS guesses: Linux 5.0 - 5.5 (99%)

Desafío 2: Entregar resultados consistentes

Anécdota de puertos abiertos.

(Mensaje del Fabian del presente para que el Fabian del futuro no se olvide de contar esta anécdota)

Desafío 2: Entregar resultados consistentes

Mitigamos las variaciones con:

- Consistencia de software y hardware de origen
- Consistencia de velocidad de red de origen
- Consistencia de dirección IP de origen
- Parametrización de escaneos
- Todo esto para QA, Stage y Prod

Desafío 2: Entregar resultados consistentes



¿Misión Cumplida? (Por ahora sí)

Aprendizajes

Pensamiento Present-Proof y Future-Proof

- SYLPA ahora y SYLPA a futuro

SYLPA = Sirva Y Lo Podamos Administrar

Pensamiento Present-Proof y Future-Proof

- SYLPA ahora y SYLPA a futuro
- Algo nunca es demasiado simple

SYLPA = Sirva Y Lo Podamos Administrar

Pensamiento Present-Proof y Future-Proof

- SYLPA ahora y SYLPA a futuro
- Algo nunca es demasiado simple
- Aceptar que los unknown unknowns existen

SYLPA = Sirva Y Lo Podamos Administrar

Muchas Gracias!