Malas Compañías

Fabian Martinez Portantier

Co-Founder, Securetia

OWASP LATAM Tour 2017 - Argentina

Caso Real: Penetration Test

No se va a divulgar el nombre real del cliente

El cual maneja información confidencial

Caso Real: Penetration Test

Infraestructura con dos aplicaciones web:

-> Sitio web principal de la empresa

-> Aplicación para carga de reclamos

Sitio web Principal

-> Ubuntu GNU/Linux

-> Apache HTTPd

-> MySQL

-> Wordpress

Sitio web Principal

-> ~~Ubuntu GNU/Linux~~

- Actualizado

-> Apache HTTPd

-> MySQL

-> wordpress

Sitio web Principal

-> ~~Ubuntu GNU/Linux~~

- Actualizado

-> ~~Apache HTTPd~~

- Actualizado

-> MySQL

-> wordpress

Sitio web Principal

-> ~~Ubuntu GNU/Linux~~

- Actualizado

-> ~~Apache HTTPd~~

- Actualizado

-> ~~MySQL~~

- Sólo accesible desde localhost

-> wordpress

Sitio web Principal

-> ~~Ubuntu GNU/Linux~~

- Actualizado

-> ~~Apache HTTPd~~

- Actualizado

-> ~~MySQL~~

- Actualizado y sólo accesible desde localhost

-> ~~wordpress~~

- Desactualizado, sin nada grave

Aplicación de Reclamos

-> Ubuntu GNU/Linux

-> Apache HTTPd

-> PHP 5.x

* Aplicación legacy, sin mantenimiento
* Evidentemente, desarrollada muy rápido

Aplicación de Reclamos

Vulnerabilidad en el módulo de file upload

-> Permitía subir cualquier tipo de archivo

-> Permitía acceder a los archivos subidos

Aplicación de Reclamos

Subir cualquier archivo + Accederlo = web Shell

Remote Hell - PHP Shell

-> Shell PHP propia

-> 54 líneas de código

-> Soporta GET y POST

-> Permite autenticar las peticiones

-> Servidor: rhell.php

-> Cliente: rh (Bash)

Remote Hell - PHP Shell - ¿why?

-> Curiosidad

-> No querer meter nada "raro" en el server

Remote Hell - PHP Shell

```
# enviar config.php remoto a nuestro equipo

rh cat config.php | grep PASSWORD > pass.txt

https://github.com/portantier/rhell
```

Aplicación de Reclamos

-> Shell con permisos de www-data (apache)

-> Base de usuarios en formato YAML

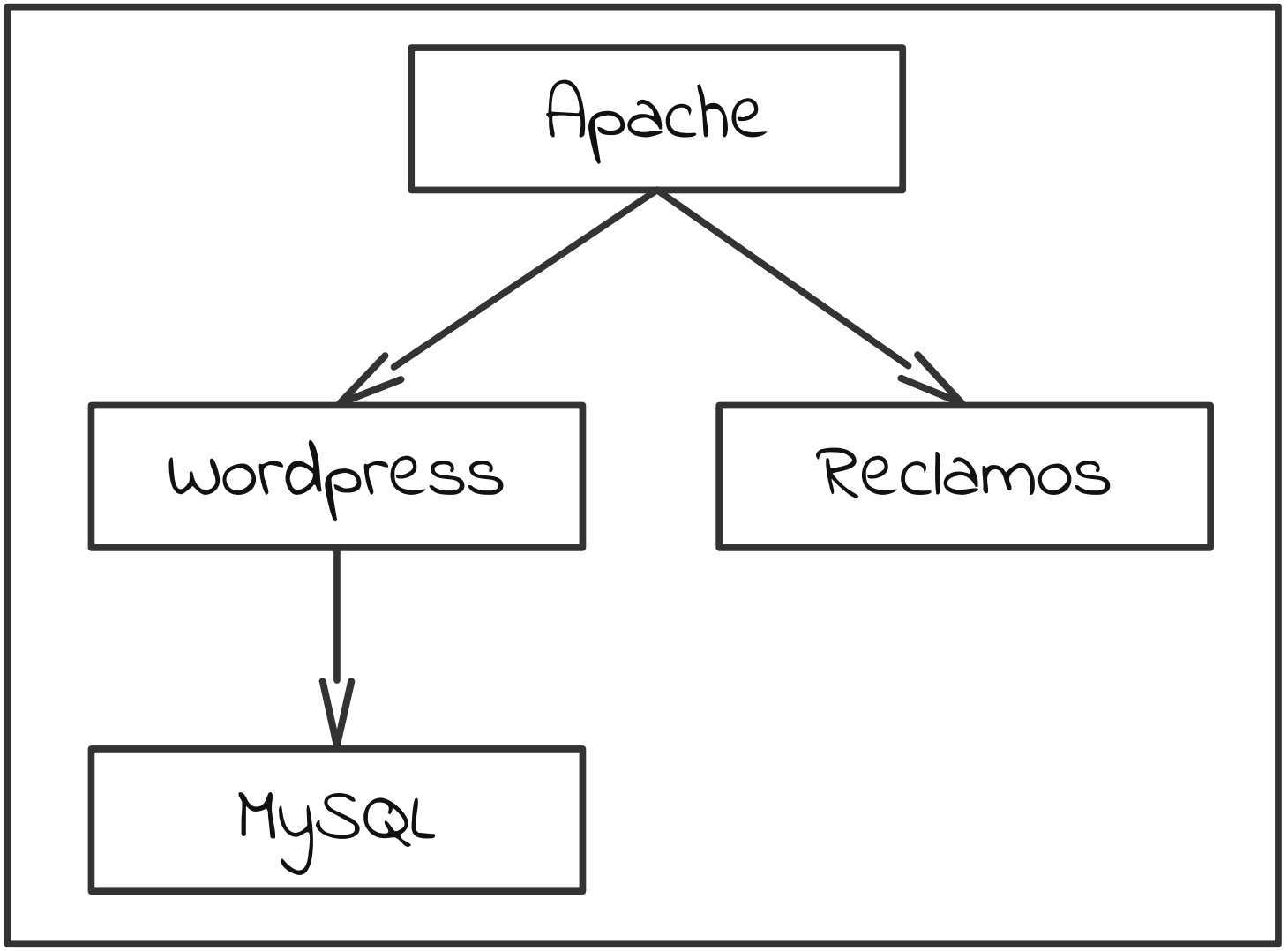
-> Usuarios con password por defecto (123456)

(la aplicación NO permitía cambiar password)

Aplicación de Reclamos

La aplicación, en si misma, no contenía ningún dato de interés, pero...

Todo en el mismo servidor !



Elevación de Privilegios

1) Shell con acceso al filesystem
2) Acceso a wp-config.php (password MySQL)
3) Login a la base de datos
4) Extracción de datos confidenciales
5) Usuarios y hashes de wordpress
6) Cracking -> Passwords de wordpress
7) Reutilización de Passwords -> más accesos

Solución: Separación de Ambientes

1) Servidores separados fisicamente

2) Servidores virtuales

3) Contenedores (docker)

Solución: Análisis de MIME Type

1) Definir formatos de archivo permitidos

2) Verificar tipo de archivo subido

3) Bloquear o eliminar archivos no permitidos

Solución: Apache mod_app_armour

1) Cada virtual host con permisos diferentes

2) Definición de permisos granulares

Solución: Apache mod_security

1) Reglas para bloquear archivos con código

2) Detección de exfiltración de datos

Muchas Gracias!

¿Preguntas?