Sobre el reporte del bug #56883 en Apache 2.4 y otros conflictos sobre Ubuntu 14.04

*— Eugenia Bahit agradece a **Hugo** (@huguidugui) por la **revisión ortográfica** de este artículo*

Recorrido sobre los problemas más frecuentes hallados en la versión 2.4 de Apache compilada por Ubuntu 14.04 y sus posibles soluciones.

El pasado jueves 21 de agosto estaba dictando clases, cuando casi literalmente «explotó», producto de un ataque de nervios generado por los innumerables «martes 13» que vengo atravesando con la versión 14.04 LTS de Ubuntu.

Algunos de mis alumnos decidieron probar esta nueva versión cuando yo aún no terminé de investigarla, pues los cambios a los que me he tenido que enfrentar, muy poco me han agradado y eso, ha retrasado mis *testeos*. Entre Ubuntu 14.04 y Apache 2.4 los cambios implementados, en mi opinión profesional, no hacen «uno bueno» y traen más problemas que soluciones:

- que si el *Web Site* no se almacena en `/var/www`;
- que si el *VirtualHost* no posee la extensión `.conf`;
- y ahora, que si se activa `MultiViews` falla `ModRewrite`.

¿Por qué si no va a /var/www da un Forbiden?

Porque en el archivo de configuración de Apache que viene en la compilación para Ubuntu 14.04 (localizado en `/etc/apache2/apache2.conf`), por defecto, se encuentran denegadas todas las solicitudes:

```
<Directory />
    ...
    Require all denied
</Directory>
```

Pero son concedidas al directorio `/var/www`:

```
<Directory /var/www/>
    ...
    Require all granted
</Directory>
```

¿Es esto una cuestión de seguridad?
Sí lo es, no me explico cómo es posible que por defecto se permita entonces un **listado del directorio raíz**, **una de las peores prácticas de seguridad**:

```
<Directory /var/www/>
    Options Indexes FollowSymLinks
    ...
</Directory>
```

Si se quiere conservar /var/www como lugar de almacenaje para los *Web Sites*, bastará con agregar un nuevo `<Directory>` y así solucionar «ese maldito *Forbiden*». Sino, solo modificar la ruta `/var/www` por la que se desee:

```
<Directory /nuevo/directorio/>
    Options Indexes FollowSymLinks
    AllowOverride None
    Require all granted
</Directory>
```

Vale aclarar que **prohibir el listado de directorios será una buena práctica**. Pero **¡ojo!** que ahora Apache requiere la asignación del signo + cuando se emplee además un signo - y de no hacerlo, fallará:

```
Either all Options must start with + or -, or no Option may
```

Para que no falle, **siempre que se agregue un -** (desactivar) a una opción, **se deberá agregar un +** (activar) a las opciones que no se estén desactivando:

```
<Directory /nuevo/directorio/>
    Options -Indexes +FollowSymLinks
    ...
</Directory>
```

¿Por qué el VirtualHost debe llevar la extensión .conf?

Sinceramente, esta me resulta una de las políticas menos justificables de la historia. **La «extensión» de un archivo** no es más que parte del nombre del archivo y **no representa en absoluto el tipo MIME del archivo**, como el sistema operativo privativo de las *ventanitas* ha querido imponer.

Entonces ¿para qué agregar un `.conf` a los archivos? Todo archivo que esté dentro de `sites-enabled` debería ser cargado como *VirtualHost* porque ¿quién colocaría allí un archivo que no fuese un *VirtualHost* habilitado?

Sin embargo, `apache2.conf` en el paquete compilado por Ubuntu 14.04, decide establecer que «solo» los archivos con extensión .conf sean cargados como sitios habilitados:

```
# Include the virtual host configurations:
IncludeOptional sites-enabled/*.conf
```

¡Qué chiste! **Quien colocase en `sites-enabled` un archivo que no pertenezca a un *VirtualHost* habilitado, estaría siendo desordenado e incoherente**. Es por ello, que una buena práctica sería cargar «todos» los archivos que estuviesen en ese directorio:

```
IncludeOptional sites-enabled/*
```

De esa forma, **la extensión de los *VirtualHost* ya no sería un problema**.

Ya no se trata de una mala decisión, se trata de un bug...

Y esta vez no es responsabilidad de Ubuntu. Sucede que en la versión 2.4 de Apache y al menos hasta la 2.4.10 **la activación de la opción `MultiViews` genera un fallo en `ModRewrite` cuando la URI contiene parte del nombre de un archivo existente**.

> ***MultiViews*** *es una opción que permite lo que se denomina «negociación de contenido» a través del módulo* ***Negotiation***

Cuando se solicite a través de la URI un archivo inexistente, `ModNegotiation` permitirá a Apache efectuar una búsqueda por patrón y servir aquel archivo que primero coincida con dicha búsqueda. De esta forma, si se solicitase el archivo `/foo` en la URI, se serviría el primer archivo cuyo nombre coincidiera con la palabra `foo`, pudiendo tratarse de `foo.php`, `foo.png`, etc. Pero siempre será el archivo que primero coincida en la búsqueda.

Cuando se trabaja con el módulo `Rewrite` activado y se implementa un sistema de URLs virtuales (conocidas entre los usuarios y profesionales del *marketing* como «*friendly URLs*») toda solicitud efectuada a través de la URI, es primero manejada por `ModRewrite` y solo cuando esto no sea posible, tomaría el mando `ModNegotiation`, pues para eso ha sido creado: para tratar de «solucionar el problema» cuando todo lo

demás falle. De esta forma, si se posee una regla de reescritura como la siguiente:

```
RewriteRule ^ whatever.php
```

Toda solicitud (absolutamente «toda») debería ser redirigida por `ModRewrite` al archivo `whatever.php` y esto, es lo que sucedía hasta la aparición de la versión 2.4 de Apache.

En esta nueva versión, cuando `/archivo-que-no-existe` es solicitado, `whatever.php` es servido. Sin embargo, al solicitar `/si-existo` el archivo `si-existo.php` es servido al igual que la petición literal `/otro-archivo-que-existe.php` se sirve de forma directa. Un error garrafal que si bien no compromete la seguridad de forma directa, podría ponerla en riesgo considerando que la implementación de reglas de reescritura suele utilizarse para enmascarar las verdaderas *URIs*.

Esto mismo es lo que reporté como *bug* a Apache:
https://issues.apache.org/bugzilla/show_bug.cgi?id=56883

Al momento de escribir este artículo, el reporte es muy reciente y por lo tanto, no se pueden esgrimir conjeturas de ningún tipo sobre el tratamiento que se le haya dado al mismo.

Dado que a diferencia de los dos casos anteriores, aquí se trata de un *bug*, **no existe solución posible hasta que no se implemente una a nivel del *core* de Apache**. No obstante, **puede implementarse una medida provisoria** para impedir que el *bug* se refleje en nuestros *hosts*. Lo que debe hacerse en **desactivar la opción `MultiViews`**. Lamentablemente, la desactivación de esta opción, **impedirá la negociación de contenido** y frente a reglas tales como:

```
RewriteRule !(^static) whatever.php
```

La solicitud de `/static/img/foo` **«NO»** sería negociada como `/static/img/foo.png` provocando entonces, un error 404 Not Found. Vale aclarar que **al desactivar la opción `MultiViews`** en un directorio, **no será posible activarla en ninguno de sus subdirectorios**, de manera tal que la activación en un caso como el siguiente, sería absolutamente ignorada por Apache:

```
<Directory /foo/>
    Options -Indexes +FollowSymLinks -MultiViews
</Directory>

<Directory /foo/bar>
    # La activación de la opción MultiViews aquí, sería ignorada
    # ya que bar es subdirectorio de foo
    Options +MultiViews
</Directory>
```