Automatización de tareas en Windows con Gorilla

Un modelo cliente-servidor sobre HTTP/HTTPS para desplegar Scripts y Software de forma desatendida y escalable

luisvela@us.es

¿Qué es Gorilla?

Gorilla se define en su proyecto GitHub como:

Munki-like Application Management for Windows

Munki es una herramienta software diseñada por Walt Disney para el entorno macOS que se utiliza para desplegar software y/o scripts en las máquinas clientes de forma desatendida.

Y esto mismo es lo que hace Gorilla, pero en Windows. Es una herramienta Open-Source con la misma idea pero en otro entorno.
Tiene licencia **Apache License 2.0**

luisvela@us.es

¿Qué podemos hacer con Gorilla?

- Realizar instalaciones de software en caliente y de forma desatendida con pocas líneas de código y sólo en los equipos deseados (escalabilidad).
- Solucionar errores que hayamos descubierto en nuestra imagen mediante scripts en Powershell sin tener que volver a clonar o ir uno a uno.

Ejemplos reales:

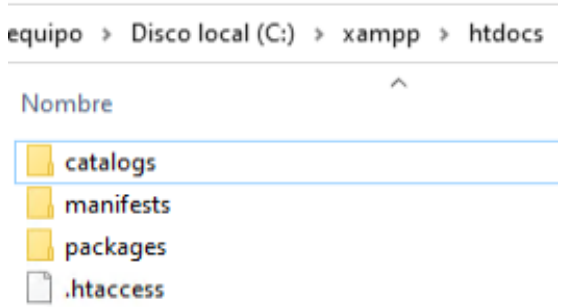
- Ejecutar un script con alguna configuración específica sólo en ciertos equipos (los de los profesores).
- Nos traen una impresora nueva, y reconfiguramos el sistema para que borre la antigua y añada la actual (instalando los nuevos drivers, predeterminando, cambiando IP si es necesario, etc.). Y sólo en cierta aula.
- Cambiar el fondo de pantalla de todos los equipos por uno nuevo en cualquier momento.
- Crear/habilitar una cuenta de usuario para un examen en una fecha determinada.
- Crear cierta tarea programada en el equipo para que realice trabajos con una periodicidad específica.

luisvela@us.es

¿Qué estructura tiene Gorilla?

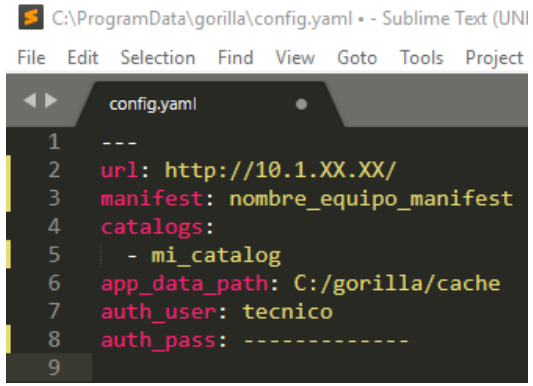
- SERVIDOR: servidor web con la siguiente estructura (y nada más)

```
[web root]
├── manifests
│   ├── *.yaml
├── catalogs
│   ├── *.yaml
└── packages
    ├── *.nupkg
    ├── *.msi
    ├── *.exe
    └── *.ps1
```



- CLIENTE: un archivo de configuración que lo vincula al servidor (y nada más)

```
---
url: https://YourWebServer/gorilla/
manifest: example
catalogs:
  - alpha
  - beta
app_data_path: C:/gorilla/cache
auth_user: GorillaRepoUser
auth_pass: pizzaisyummy
tls_auth: true
tls_client_cert: c:/certs/client.pem
tls_client_key: c:/certs/client.key
tls_server_cert: c:/certs/server.pem
```



luisvela@us.es

¿Qué necesito saber para empezar con Gorilla?

No se requieren grandes conocimientos específicos. Puede hacerlo cualquier persona que sepa crear/interpretar un script básico.

- **Sistemas:** Montar un servidor Web
- **Scripting:** Manejarse en línea de comandos (Powershell y CMD).

¿Qué seré capaz de hacer con Gorilla?

A partir de un Windows 10 base (con unas configuraciones mínimas) y un catálogo de aplicaciones y scripts en mi servidor de gorilla seré capaz de crear una imagen casi completa de aulas (con aplicaciones en su última versión, actualizaciones de sistema, cuentas de usuario, impresoras, y mucho más).

Y si se te olvida alguna configuración o encuentras cualquier error tras clonar el aula, ¡no te preocupes! en caliente puedes solucionarlo.

luisvela@us.es

¿Qué haríamos en el curso?

Máquina Servidor:

- Montar un servidor web con una instancia de Multipass que contenga la estructura de carpetas de gorilla (catalogs, manifests y packages).
- Crear varios manifests
- Definir un catalog y crear un script básico. Por ejemplo, que cree una carpeta en el escritorio y deje una marca para que podamos ver que ya se ha ejecutado (aunque borrásemos la carpeta)

Máquina/s Cliente/s:

- Definir en cada cliente su archivo de configuración (a qué manifest del servidor lo vamos a asociar)
- Desplegar tareas en estos
- Observar logs en tiempo real o tras la ejecución
- Crear nuestros propios logs (muy útil)
- Controlar tareas que sólo queremos que se ejecuten una vez
- Procedimientos para depurar errores
- Herencia de manifests

luisvela@us.es

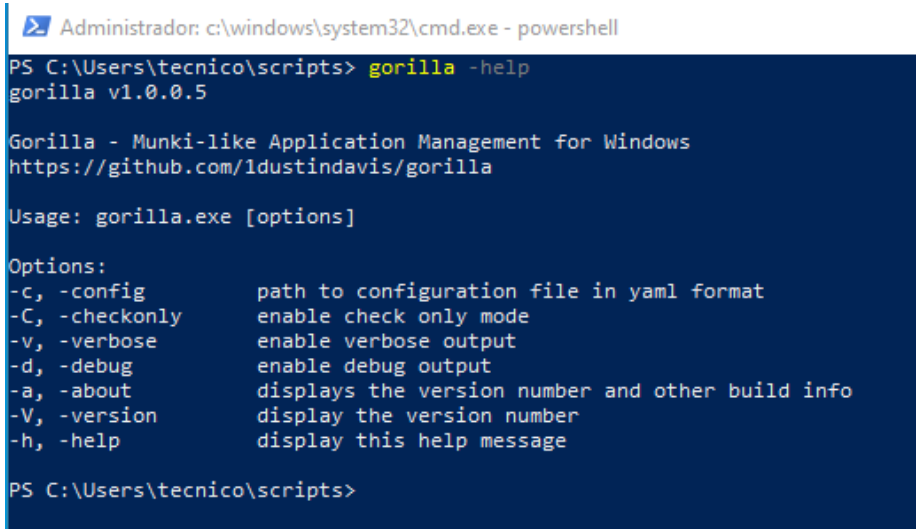
Desplegar manifests en el cliente con Gorilla

¿Cómo conectan cliente-servidor?
Con el binario de Gorilla. Siempre desde el cliente hacia el servidor.

¿Cada cuanto tiempo conectan?
Con la periodicidad que yo quiera. Debemos crear un script que indique al cliente cuándo debe conectarse.

¿Y si quiero ejecutar cierta tarea de un manifest sólo una vez?
Si ejecutamos gorilla de forma periódica, necesitaremos que ciertas tareas sean Idempotentes. Un procedimiento de trabajo con scripts soluciona esto.

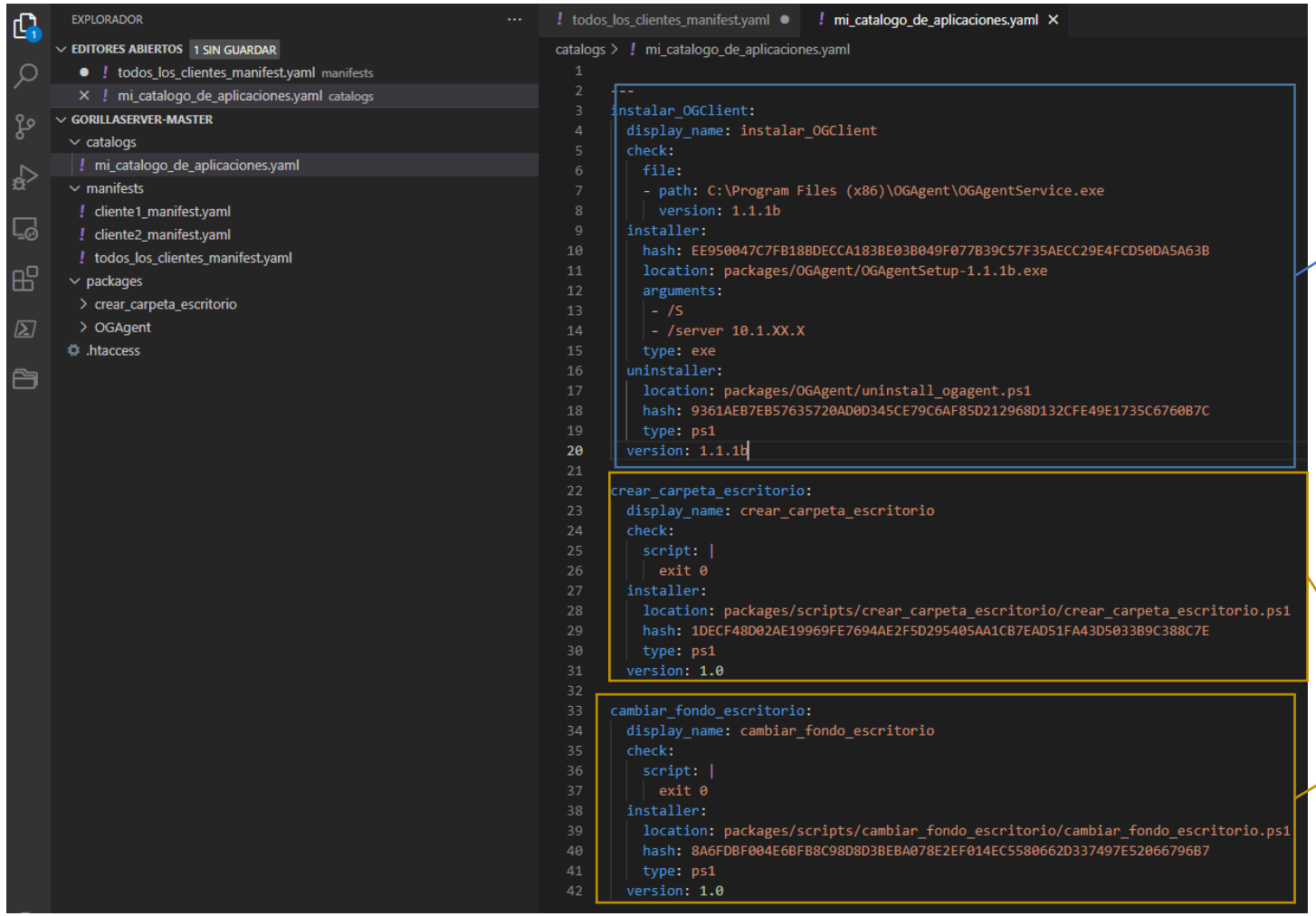
¿Cómo puedo depurar errores si no se ejecuta como espero?
Conociendo la forma de trabajo de Gorilla. También los archivos de log y su significado. Además, el editor de Powershell también nos puede ayudar bastante.



luisvela@us.es

Ejemplos:
- Paso 1/3 - Implementando el **catálogo** de aplicaciones del servidor

Aquí defino todo lo PUEDO hacer en mis clientes, lo haga o no



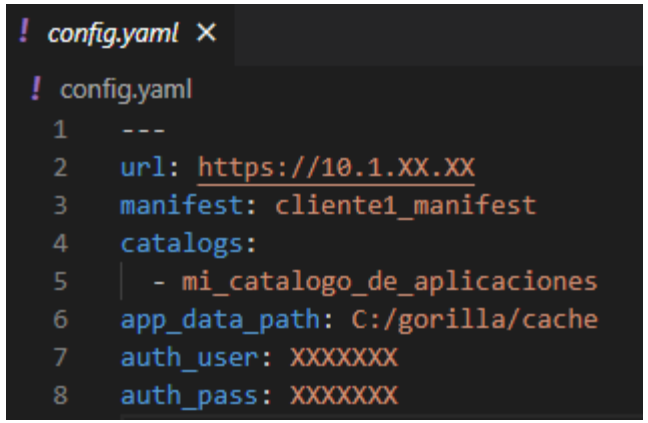
Instalación software (.exe)

Powershell scripts (.ps1)

luisvela@us.es

Ejemplos:
- Paso 2/3 - vinculando un cliente con su manifest

A un cliente concreto, le digo: TÚ vas a hacer lo que diga “cliente1_manifest”



Configuración de un pc cualquiera del aula. Lo estoy asociando a “cliente1_manifest”

¿Tengo que ir equipo a equipo configurando la línea “manifest” de su archivo de configuración “config.yaml”?

La respuesta es NO.
Con un juego de scripts y manifest auxiliares podemos automatizar esta configuración, para que cada equipo obtenga su manifest correspondiente de forma automatizada en función de un dato concreto, como por ejemplo su IP.

luisvela@us.es

Ejemplos:
- Paso 3/3 - implementando **manifests** en clientes

Indico qué tareas de mi catálogo quiero implementar en cada cliente, o en ambos (included_manifests)

cliente1_manifest.yaml

```yaml
---
name: cliente1_manifest
managed_installs:
  - instalar_OGClient
managed_uninstalls:
managed_updates:
included_manifests:
  - todos_los_clientes_manifest
catalogs:
  - mi_catalogo_de_aplicaciones
```

Asociado con cliente1

cliente2_manifest.yaml

```yaml
---
name: cliente2_manifest
managed_installs:
  - crear_carpeta_escritorio
managed_uninstalls:
managed_updates:
included_manifests:
  - todos_los_clientes_manifest
catalogs:
  - mi_catalogo_de_aplicaciones
```

Asociado con cliente2

todos_los_clientes_manifest.yaml

```yaml
---
name: todos_los_clientes_manifest
managed_installs:
  - cambiar_fondo_escritorio
managed_uninstalls:
managed_updates:
included_manifests:
catalogs:
  - mi_catalogo_de_aplicaciones
```

Sin asociación

luisvela@us.es