8. AUTO MANIFEST

Hasta el momento, en cada cliente hemos tenido que configurar el fichero config.yaml de forma manual, cada uno con su propio manifest.

¿Tengo que hacer esto a mano en cada uno de los clientes? -> NO

Podemos automatizar esta tarea, dando una configuración genérica y después especializando dicha configuración en cada cliente localmente.

8.1 Configuración genérica del archivo config.yaml

```
---
url: https://gorillaserver/
manifest: default_manifest    -> OJO A ESTA LÍNEA
catalogs:
  - mi_catalogo
app_data_path: C:/gorilla/cache
#auth_user: tecnico
#auth_pass: XXXXXXXXXX
#tls_auth: true
#tls_client_cert: C:\gorilla\cert.pem
#tls_client_key: C:\gorilla\key.pem
#tls_server_cert: C:\gorilla\server.pem
```

8.2 Manifest genérico “default_manifest”

```
---
name: default_manifest
managed_installs:
  - auto_manifest
managed_uninstalls:
managed_updates:
included_manifests:
catalogs:
  - mi_catalogo
```

8.3 Tarea “auto_manifest”

```
# Obtenemos la dirección IP del cliente
$data = Get-NetIPAddress -AddressFamily IPv4 -IPAddress "10.1.XX.*"
$ipAddress = $data.IPAddress
$manifest = "default_manifest"

# Obtenemos el manifest que corresponde a la IP obtenida
switch ( $ipAddress )
{
    "10.1.XX.AA" { $manifest = 'cliente1_manifest'    }
    "10.1.XX.BB" { $manifest = 'cliente2_manifest'    }
    "10.1.XX.CC" { $manifest = 'cliente3_manifest'    }
    "10.1.XX.DD" { $manifest = 'cliente4_manifest'    }
}


# Actualizamos el manifest en el archivo de configuracion
$config_file = "C:\ProgramData\gorilla\config.yaml"
# Obtenemos el contenido del archivo y modificamos el manifest en el archivo
$config_file_replace = (Get-Content -Path $config_file) -Replace
"\w*_manifest", $manifest
# Sustituimos en el fichero original
Set-Content -Path $config_file -Value $config_file_replace
```

Expresiones regulares en powershell:

acerca de las expresiones regulares - PowerShell | Microsoft Docs