# Monitorización

- Actualizar agente Beszel (No necesario actualmente)
- Monitorización simple de SAI (UPS) con Python

# Actualizar agente Beszel (No necesario actualmente)

---

# Script de actualización para Beszel

Este script te ahorra dolores de cabeza: automatiza la actualización del agente **Beszel** en tu servidor. Olvídate de actualizar a mano: con esto, siempre estarás usando la última versión disponible.

---

# El script completo

> Repositorio en Gitea

---

# ¿Qué hace exactamente este script?

- Se mueve al directorio donde tienes Beszel.
- Para el servicio para no liarla mientras actualiza.
- Descarga el binario nuevo adaptado a tu sistema.
- Sustituye el antiguo y le da permisos para que pueda ejecutarse.
- Verifica que todo haya salido bien.
- Arranca de nuevo el servicio.
- Te va avisando de cada paso, para que sepas que no ha explotado nada.

---

# Paso a paso, con lupa

## 1. Configuración de variables

Define dónde está Beszel y cómo se llama su servicio:

```
DIR="/ruta/a/tu/directorio/Beszel"
SERVICE="beszel-agent.service"
```

## 2. Cambio de directorio

Se mete en el directorio. Si falla, te avisa y corta la ejecución:

```
cd "$DIR" || { echo "No se pudo cambiar al directorio $DIR"; exit 1; }
```

## 3. Parar el servicio

Detiene el agente para actualizar sin problemas:

```
sudo systemctl stop "$SERVICE"
```

## 4. Descargar y reemplazar el binario

Usa `curl` para traer la última versión, extrae el ejecutable y lo deja listo:

```
curl -sL "https://github.com/henrygd/beszel/releases/latest/download/beszel-agent_$(uname -s)_$(uname -m | sed 's/x86_64/amd64/' | sed 's/armv7l/arm/' | sed 's/aarch64/arm64/').tar.gz" | tar -xz -O beszel-agent | tee ./beszel-agent >/dev/null && chmod +x beszel-agent
```

## 5. Comprobación de la descarga

Se asegura de que el archivo esté donde debe:

```
if [ -f "$DIR/beszel-agent" ]; then
  echo "El binario se actualizó correctamente."
else
  echo "Error: El binario no se descargó correctamente."
  exit 1
fi
```

## 6. Reiniciar el servicio

Levanta de nuevo el servicio para que trabaje con la versión nueva:

```
sudo systemctl start "$SERVICE"
```

## 7. Mensaje final

Te confirma que todo ha ido bien:

```
echo "Actualización completada."
```

---

# Cómo usarlo

1. Guarda el script como `Update_Beszel.sh`.
2. Dale permisos de ejecución:

   ```
   chmod +x Update_Beszel.sh
   ```

3. Ejecútalo cuando quieras actualizar:

   ```
   ./Update_Beszel.sh
   ```

---

# Ventajas de este script

- **Automatización total:** No tienes que ir a GitHub a mano.
- **Velocidad:** Lo actualiza todo en segundos.
- **Compatibilidad inteligente:** Detecta tu sistema y arquitectura solo.
- **Seguridad:** Para el servicio antes de tocar nada, como debe ser.

---

# Nota importante

Desde las últimas versiones, **Beszel** ya incluye una función de autoactualización integrada.

¿Quiere decir que este script ya no sirve? No exactamente: **puedes seguir usándolo si prefieres forzar la actualización manualmente o asegurarte de tener el último binario en situaciones especiales**.

# Monitorización simple de SAI (UPS) con Python

---

Este script permite consultar el estado de tu SAI (*Uninterruptible Power Supply*) usando `upsc`, mostrando la información en consola con colores, alertas y detalles básicos del sistema. Es una forma ligera y práctica de saber si todo va bien con tu SAI, sin necesidad de dashboards pesados.

El código está disponible aquí: CheckSAI.sh en Gitea

---

## Características

- Consulta parámetros clave del SAI: carga, voltaje, frecuencia, carga de salida...
- Muestra alertas si algún valor está fuera del rango normal.
- Incluye información del sistema (hostname, procesador, plataforma).
- Salida en colores para facilitar la lectura.
- Ayuda integrada con `-h` o `--help`.

---

## Requisitos

Antes de usar el script, necesitas:

- Python 3.x instalado.
- El paquete `colorama`:

```
pip install colorama
```

- Tener el comando `upsc`, que forma parte de **NUT (Network UPS Tools)**.

---

## Adaptaciones necesarias

Antes de ejecutar el script, asegúrate de editar:

- **Nombre del SAI (`ups_name`)**: por defecto está como `nutdev1`. Sustitúyelo por el nombre real de tu SAI.
- **Dirección y puerto del servicio**: si no usas `localhost:4500`, edita el endpoint en la función que lanza `upsc` para reflejar la IP/puerto correctos.

---

# ¿Qué hace exactamente?

1. Ejecuta `upsc` para obtener los datos en bruto del SAI.
2. Extrae y organiza esa información en un formato legible.
3. Comprueba si los valores críticos están dentro de los rangos normales.
4. Muestra alertas si algo está fuera de lo previsto.
5. Añade información sobre el sistema donde se ejecuta.

---

# Alternativa visual

Si prefieres una interfaz gráfica sencilla en el navegador, puedes probar con PeaNUT, también basado en NUT.