Control Accesos Proxy

Vigente en version: v2.3.2

Documentacion completa de API

`api/Relojes/enrolador` POST (sin uso)

Retorna {"status": (bool)}

Campos:

- (int)nodo

Accion: Setea un reloj activo como enrolador, y quita cualquier otro que este marcado como tal.
Se asume que solo puede existir un reloj enrolador.
NOTA: En el codigo existe un comentario que parece ser un bug.
`puede que no quede marcado ninguno, en cuyo caso se hara busqueda siempre hasta reiniciar el proceso`

`api/Relojes/{(int)nodo}/estado` GET (sin uso)

Retorna JSON

- [(bool), (string)"yyyy-MM-dd HH:mm:ss"]

Accion: Retorna el estado de conexion para un nodo en particular.

`api/Relojes/estado` GET

Retorna JSON

- {"(number)":[(bool), (string)"yyyy-MM-dd HH:mm:ss"]}

Campos:

Accion: En JSON si el primer valor no es 0, entonces no es un array, es un objeto con atributo numerico, pasado como string.
El valor de cada atributo es un array cuyo primer item es el estado del reloj como booleano, y el segundo la ultima fecha activo.

`api/Relojes/agregar` POST

Retorna JSON

- {"status": (bool)}

Campos:

- (string)ip
- (string)dns
- (int)puerto
- (bool)enrolador
- (int)id
- (int)nodo
- (string|null)ultima_marcacion: 'Y-m-d H:i:s'
- (bool)habilitado (no se usa, peligro)

Accion: Crea nueva instancia de Reloj con los datos pasados. Si el Reloj ya estaba agregado, no hace nada y devuelve "false". Si no existia lo agrega y devuelve "true".

Para la conexion utiliza "{dns}:{puerto}"
Timeout 35000 (35 segundos)
Luego intenta conectar 3 veces cada 10000 (10 segundos)

api/Relojes/recargar **POST**

Retorna

- {"status": (bool)}

Campos:

- (string)ip
- (string)dns
- (int)puerto
- (bool)enrolador
- (int)id
- (int)nodo
- (string|null)ultima_marcacion: 'Y-m-d H:i:s'
- (bool)habilitado (no se usa, peligro)

Accion: Busca la instancia de Reloj por nodo, e intenta eliminarla (y su link de comunicacion). Luego intenta agregarlo de nuevo igual que en `api/Relojes/agregar`
NOTA: Proceso buggeado, elimina pero no levanta, al parecer el proceso sigue corriendo pero sin conexion.
Esta funcionalidad deberia ser revisada.

api/Relojes/{(int)nodo}/borrar_marcaciones **GET**

Retorna JSON

- {"status": true}

Campos:

Accion:
Busca la instancia de Reloj por nodo, y envia la orden de limpiar registros internos (marcaciones offline).

api/Relojes/{(int)nodo} **DELETE (sin uso)**

Retorna JSON

- {"status": true}

Campos:

Accion:
Busca la instancia de Reloj por nodo, y lo desconecta para luego eliminar la instancia.

api/Relojes/{(int)nodo}/habilitar **POST (sin uso)**

Retorna JSON

- {"status": true}

Campos:

Accion:
Busca la instancia de Reloj por Nodo y la desconecta junto a su link, para volver a levantarlo y reconectar.
Al parecer la reconeccion se intenta infinitamente.

api/Relojes/{(int)nodo}/deshabilitar **POST (sin uso)**

Retorna JSON

- {"status": true}

Campos:

Accion:
Busca la instancia de Reloj por Nodo y marca como Deshabilitada pero no la borra, ni desconecta.

NOTA: Esta funcionalidad deberia ser revisada.

api/Templates/accessId/{(int)documento} GET

Retorna JSON

- [{"accessId": (int)DNI,"index": (int),"data": (string)}, {"accessId": (int)DNI,"index": (int),"data": (string)}]

Campos:

Accion: Sirve para obtener los "template" (huellas) desde la instancia del reloj enrolador, por usuario (dni).
El index, obedece a los valores 0 y 1, respecto al dedo de la mano (huella digital) escaneado.

api/Templates/baja DELETE (En uso - Deprecado)

Retorna JSON

- null, false, {"status": true}

Campos:

- (int)accessId

Accion: Identico a api/Templates/accessId/{(int)documento}. Endpoint que en realidad se deberia usar.
NOTA: Se deberia dejar de utilizar y eliminar y/o hacer una llamada interna al otro metodo para facilitar mantenimiento.

api/Templates/accessId/{(int)documento} DELETE (Sin uso)

Retorna JSON

- null, false, {"status": true}

Campos:

Accion: Trabaja unicamente con la intancia de Reloj del tipo Enrolador.
Elimina los "templates" (huellas) del equipo para el empleado especificado por DNI.
Elimina al DNI del usuario del reloj enrolador.

api/Templates/distribuir/{(int)documento} POST

Retorna

- null, false, {"status": true}

Campos:

- (json)templates = [{"persona":{"documento":(int)}, "index":(int), "data":(string)}]
- (json)nodes = ["(int)id_nodo"]

Accion: Realiza la baja de de los templates (huellas digitales en BASE64) enviadas correspondiente al documento pasado por URL, y del empleado para todos los nodos pasados en el array, y luego vuelve a dar de alta tanto al usuario como al template (huellas digitales).
El valor `"index":(int)` suele ser 0 o 1 para identificar que son dedos de la mano distintos (o distintas manos).
`"data":(string)` es un sting en base64.
Sirve para que cada Nodo (Reloj) tenga el registro y huella de la persona que se quiere dar acceso autorizado.
NOTA: Tiene un comportamiento inconsistente respecto al manejo de documentos DNI, por un lado el codigo pareciera estar preparado para recibir distintas personas, pero a fin de cuentas solo utiliza el pasado por URL.
Esta funcionalidad deberia ser revisada.
La performance del PHP para operar con esto, es una bosta (con amor <3).

api/Templates/distribuirTarjeta/{nroTarjeta} POST

Retorna

- null, false, {"status": true}

Campos:

- (json)nodes = ["(int)id_nodo"]

Accion: Elimina el registro `{nroTarjeta}` de todos los nodos pasados, y lo vuelve a cargar.
Se asume que el proceso de eliminar es para evitar duplicidad o colision en caso de existir.

`api/Templates/distribuirTarjetaDesenrolar/{nroTarjeta}` **POST**

Retorna JSON

- null, false, {"status": true} || {"status": false, "message":(string)}

Campos:

- (json)nodes = ["(int)id_nodo"]

Accion: Elimina el registro `{nroTarjeta}` de todos los nodos pasados

`api/users` **POST (Inexistente)**

Retorna

- null, false, (object)->status

Campos:

- accessId (int)dni_documento

NOTA: Este metodo se deja documentado porque se detecta uso en la aplicacion de PHP pero el servicio CAP no lo posee.