Grouper web services act as

There is another "Act as" for grouper, which is audited. This enhancement is for Grouper messaging to web services and is not audited (more of a backdoor).

Send the act as

To get this to work, an HTTP header is sent with the request. You need to send the source ID. And you need to send the id or identifier. Note, this is a layer on top of the normal actAs and does not conflict, both can be used. Note these header values should be BASE64 encoded with UTF-8.

```
X-Grouper-actAsSourceId: base64(sourceId)

X-Grouper-actAsSubjectId: base64(subjectId)
-or-
X-Grouper-actAsSubjectIdentifier: base64(subjectIdentifier)
```

e.g. for jdbc and test.subject.0

```
X-Grouper-actAsSourceId: amRiYw==
X-Grouper-actAsSubjectId: dGVzdC5zdWJqZWN0LjA=
```

Grouper client options

```
  --grouperActAsSubjectId=subjId
  --grouperActAsSubjectIdentifier=subjIdent
  --grouperActAsSubjectSource=sourceId
```

Configure the WS to accept a grouper act as

in grouper-ws.properties

```
# similar syntax as ws.act.as.group but for the grouper actas (e.g. for grouper messaging to WS bridge)
# Web service users who are in the following group can use the actAs field to act as someone else
# You can put multiple groups separated by commas.  e.g. a:b:c, e:f:g
# You can put a single entry as the group the calling user has to be in, and the grouper the actAs has to be in
# separated by 4 colons
# e.g. if the configured values is:       a:b:c, e:f:d :::: r:e:w, x:e:w
# then if the calling user is in a:b:c or x:e:w, then the actAs can be anyone
# if not, then if the calling user is in e:f:d, then the actAs must be in r:e:w.  If multiple rules, then
# if one passes, then it is a success, if they all fail, then fail.
ws.grouper.act.as.group =
```