Concepts and Background -- Create Session Key

I have created 4 examples that show how to generate a Session Key in SystemLink, 2 by referencing existing Policies and 2 by passing specific Statements, 2 as python scripts you can run in Visual Studio Code or DOS and 2 as JupyterHub notebooks you can run right there in SystemLink.

**This is what it means for a Swagger route to be labelled as “whitelisted operations”:**

1) The "POST /session-keys" route will only run when authenticated with a whitelisted ApiKey

2) Any whitelisted-ApiKey-only routes will be intercepted by the SystemLink web server and denied automatically

3) Therefore, you can not run this route in a python script that makes requests of "[https://localhost:443/](https://localhost/)"

4) For the same reason, you also can not run this route from Swagger, even though Swagger offers you that option

5) You have to bypass the SystemLink Web Server at port 443 and direct all requests instead directly to port 12100

6) Any "WhitelistedApiKey" field in any SystemLink json Config file will work-- use like a regular ApiKey in the requests "headers"

7) Creating Session Keys this way bypasses all SystemLink's Workspace and Role controls-- USE WITH CARE

8) It should therefore ONLY be done by the SystemLink Administrator, on the SystemLink server computer

In the "Access Control" page in the SystemLink GUI, you see Workspaces, Roles, and Users displayed as parallel options, though there is an implicit hierarchy there. The objects in the API that hold that information are more numerous and differently named-- here's a summary attempting to provide clarity:

**GUI: Workspace\User+Roles\Privileges**

Each "Workspace" is a subset of items (Files, Tags, Systems...) that can be acted upon, with an understandable name ("Default")

Each "User" is a log-in account (or logical group of accounts) that is added with one or more Roles to one or more Workspaces

Each "Role" is a collection of atomic Privileges with an understandable name ("Systems Maintainer")

Each "Privilege" is an atomic action that a user assigned that "Role" can perform inside any "Workspace" assigned to that user ("Upload files")

**API: key\user+policies\statements\actions+workspace+resources+description**

Each "api-key" or "session-key" describes a collection of policies, associated to a user, with an understandable name ("Super User Access")

Each "policy" describes a collection of statements that builds a complete set of permissions, with an understandable name ("Super User")

Each "statement" describes a collection of actions with associated workspace, resource list, and understandable description ("File Maintainer")

Each "action" describes an atomic privilege like you see in the Role permissions list in the GUI (file:Query, file:Download, file:Upload, file:Update)

Only Session Keys expire automatically. A regular ApiKey can only be deleted or disabled.

Only Session Keys require whitelisted authentication. A regular ApiKey can be created with normal web service requests.

The ApiKey definition will always have an associated "userId" property.

The Session Key requires at least "userId" or "orgId" property and will accept both.

The “Super User” Policy is the only Policy with "builtIn" = True.

You can look up the statements and actions of each Role by running the “GET /policy-templates” route with "builtIn" = True.