

# What is Open Service Access?

Open Service Access (AKA OSA) is a Web gateway designed as a security element of your network, dedicated to Web resources protection and publishing:

* Let web developers be focused on the core business of their services and let OSA take care of security and protection of those resources,
* Let web services user access to all your services form different network works in a single manner and doesn’t bother providers with those issues

# What does Open Service Access?

OpenServicesAccess can be view as a web reverse proxy.

It can:

* Publish backends on different networks (default configuration allow up 2)
* Ensure encryption (https)
* Verify authentication with basic authentication or with a form based mode
* Check authorizations
* Apply global quotas (per second, day and month) for a backend
* Apply user quotas (per second, day and month) for a backend
* Forward consumer identity to provider
* Forward publishing endpoint to provider
* Provide advance service usage logging to administrators.
* Offers simple GUI to adminstrators
* Be used as SSO for web sites

Even if OSA have an administrator GUI to manage the system, like it’s a SOA component, it’s SOA compliant: all administration facilities are available through REST web services.

# How does Open Services Access works?

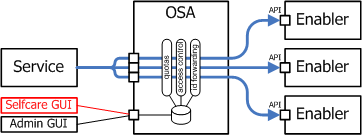
Open Services Access is a completely open source solution based on Apache/PHP/MySQL.

OSA works as a reverse proxy to and protect your resources, REST, SOAP, HTML, Web Sockets. To do so, you have to provide information to expose the resources, throught the AppliaceManagerAdmin. During the installation, be carefull to configure properly the admin password: it is needed to connect to the administration portal at the following url: https://''ServerName''/ApplianceManagerAdmin

It is composed of 2 modules working together:

* RunTimeAppliance: which is in charge to manage access to web services at runtime. It is mainly based on Apache reverse proxy enhanced with new facilities,
* ApplianceManager: which is the administration application, including GUI and REST web services.

It comes with two parts: a mediation platform based on apache2, and a management application, as illustrated in the following picture:



# Authorization model

Each resource deployed in the Open Service Access platform can be protected by an authorization. To do this, a user and authorization management has been implemented in the platform. If a service needs authorization, when using it, a prompt asks for a user. This user has to be a member of a group. A service is attached to a group.

Group

Service

User

belongs to

accepts

\*

1

\*

\*

# Submitting a new service

This section gives you the different steps to submit a new service.

1. Create a new group: First thing to do is to declare a new group. This group will be attached to the new service. You can of course use an existing group, but this would mean to share the user list with different services.
2. Fill the group with consumers: If you want to allow a new consumer to use the service, you have to create a user ('''Users''' menu, '''Add''' section). Then, edit the user and add it to the group previously created.
3. Publish a service: the publication of a service is rather simple, and necessitates very few information, although advanced usage can be configured as well. As seen in the following picture, three parameters only are enough: a service name (the id), an alias, and of course the provider URL. By default, the checkbox ‘IsPublished’ is checked, wich means that the service is usable. For example, if my OSA is installed on the server ‘ServerName’ if I want to publish an SMS service which is located at the URL <http://168.192.1.1/sms-oneapi> (this is a local IP), I may enter this:
   * + Service name: SMS
     + Frontend alias: SMS-V1
     + Backend URL: <http://168.192.1.1/sms-oneapi>

Then save. At this moment, I can use the URL http:// ServerName:81/SMS-V1 to consume the service.

# Advanced parameters

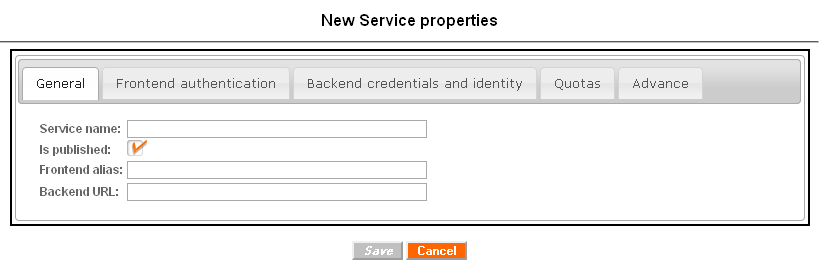
The previous section gives the minimum steps to be able to publish a service. The tabs on the picture give a view of the possibilities that are allowed to the administrator.

Frontend Authentication enables you to select a group for protection of the service. If Enable user authentication is checked, the service consumer will have to use credentials with http basic authentication to consume the service.

Backend credentials and identity can be necessary if the backend service needs authentication itself. Credentials can be provided at this place.

Quotas: On this tab, the administrator can enter the maximum number of requests allowed to consume the service. The quotas can be global or per user. Three numbers are asked: usage by second, day and month.

Last tab allows user to personalize apache directive to put more information in the http headers.



# Have a look to a typical use case…

Let’s imagine that you have in your internal network some high value added web services and you want to give under control, access to all those services:

* Internally to your own developers,
* Externally to partners.

Of course, you want to avoid to re-implement or re-study for each one security enforcement and network connectivity solutions……

Let’s imagine that your network topology looks like:

It’s were OSA take place!

By deploying OSA in your interco DMZ you are now able to:

* Give access to your services to external consumers through HTTPS
* Give access to your services to your developers through HTTP for debug
* Protected your services
* Manage network connectivity issues to private production network for only a single infrastructure (OSA)

Your network may become: