Getting Started

This document describes how to install the *SAP HANA XSA Runtime* and how to deploy XSA applications

Contents

Installation and Startup

You have 4 installation options, choose one of them:

Installation Option	Description
Monsoon Readymade (XSA+HANA)	The most convenient setup option. You'll get a fully installed and running HANA+XSA system in a Linux Monsoon VM.
Monsoon Readymade (XSA standalone)	Installs a standalone XSA system in a Monsoon image and connects it to a custom HANA. Also convenient, able to run in a Monsoon sandbox.
HANA+XSA using hdblcm	Run the HANA installer 'hdblcm' to install HANA along with XSA on your favorite Linux box.
Manual XSA installation	Manually install and start a standalone XSA runtime on your local machine and connect it to a running HANA.

Command-Line Interface

After successful installation, the XSA Runtime can be operated using a command line client. It's named `xs` and located in `<XSA_ROOT>/bin`. It's commands are similar to the CloudFoundry `cf` client, for example:

Display all available commands:

```
$ xs help
```

Get detailed help:

```
$ xs help <command>
```

Version information:

```
$ xs -v
```

Find download links for the XS client at the sidebar on the right-hand side of this page.

Userful: There's also a conventient bash TAB completion script for xs commands!

Bug Reports
• Bug tracker: XSA Runtime • Mailing list: xs2-users

Download: Weekly Release	
Version:	v0.1642
Build-Date:	October 31, 2016
HANA:	SPS 11.2 - SPS 13
Changes:	Recent Changes
XSA Runtime:	Windows x64 Linux x64 Darwin x64 Linux ppc64 Linux ppc64le

Login

<SID>adm login

When logged in at your HANA system with the <sid>adm user in an SSH session, you can use

```
$ xs-admin-login
```

to log in as XSA admin user conveniently. This script will grab the right API URL, the credentials and care about the certificate. Afterwards, you can use the `xs` client to operate the XS Controller.

External login

To log in from outside the HANA system (or as non-admin), use the `xs login` command:

```
$ xs login -a <API_URL>
```

You are asked for credentials interactively. The default depends on the installation scenario you chose above:

	Readymade Full	Readymade Standalone	Local Installation	hdblcm
API_URL	`https://<fully_qualified_host>:30030`			`https://<your-hana-host>:3<instance-nr>30`
USERNAME	XSA_ADMIN			XSA_ADMIN
PASSWORD	Toor1234			Toor1234
ORG	myorg			orgname
SPACE	PROD			PROD

See also Login with HANA users in order to see how other HANA users can be enabled to access the XS Controller.The actual credentials are chosen during installation. So if the default do not work, ask the admin who installed the XSA system.

SSL Certificate

In case you use a `https` API URL, you have to

either provide the right certificate

```
$ xs login --cacert <pem file>
```

(In a HANA system, the <pem file> to use is located in `/hana/shared/<SID>/xs/controller_data/controller/ssl-pub/router/default.root.crt.pem`. Copy this one to your machine)

or, for development purposes, skip ssl validation:

```
$ xs login --skip-ssl-validation
```

Adding further users

From the start, there exists the admin user `XSA_ADMIN.` To create additional users, you need to

XSA Client:	Windows x64 Linux x64 Darwin x64 Linux ppc64 Linux ppc64le

Download: Customer Release

Version:	v1.0.34
Build-Date:	August 29, 2016
HANA:	SPS 11 - SPS 12.1
Changes:	XSA Release Notes

create HANA users and assign XS roles to them (see Login with HANA users and HANA SQL Guide for a full description). Therefore, ssh into the HANA system with the <sid>adm user:

Adding further 'admin' users

To create a new admin user called `NEW_ADMIN`, use the following command line:

```
$ /usr/sap/XSA/HDB00/exe/hdbsql -i 00 -n localhost:30015 -u
SYSTEM -p Toor1234 "CREATE USER NEW_ADMIN PASSWORD
\"Welcome1\" SET PARAMETER XS_RC_XS_CONTROLLER_ADMIN =
'XS_CONTROLLER_ADMIN'"

$ /usr/sap/XSA/HDB00/exe/hdbsql -i 00 -n localhost:30015 -u
NEW_ADMIN -p "Welcome1" ALTER USER NEW_ADMIN PASSWORD "Toor123
4"
```

The first statement creates the user. Note, that you need the credentials for the HANA user `SYSTEM` and the example assumes you're HANA has instance number 00 and you system ID is XSA. The second statement does the necessary password change on first login.

Adding further restricted users:

To create a restricted user `NEW_USER`, use the following command line:

```
$ /usr/sap/XSA/HDB00/exe/hdbsql -i 00 -n localhost:30015 -u
SYSTEM -p Toor1234 "CREATE USER NEW_USER PASSWORD \"Welcome1\"
SET PARAMETER XS_RC_XS_CONTROLLER_USER = 'XS_CONTROLLER_USER'"

$ /usr/sap/XSA/HDB00/exe/hdbsql -i 00 -n localhost:30015 -u
NEW_USER -p "Welcome1" ALTER USER NEW_USER PASSWORD "Toor1234"
```

Then assign the actual rights of the user by logging in as an XSA admin user (e.g. `XSA_ADMIN`) and assigning XSA roles via xs cli:

```
$ /usr/sap/XSA/HDB00/exe/hdbsql -i 00 -n localhost:30015 -u
NEW_USER -p "Welcome1" ALTER USER NEW_USER PASSWORD "Toor1234"

$ xs set-space-role NEW_USER myorg SAP SpaceDeveloper
```

The second statement sets `SpaceDeveloper` rights for `NEW_USER` in organization `myorg` and space `SAP`.

Use the xs help to get more information:

```
$ xs help set-space-role

$ xs help set-org-role

$ xs help unset-space-role

$ xs help unset-org-role
```

Navigate through orgs and spaces

Print out your current location:

```
$ xs target
```

Print out available orgs and spaces:

XSA **Runtime:**	Windows x64 Linux x64 Darwin x64 Linux ppc64
XSA **Client:**	Windows x64 Linux x64 Darwin x64 Linux ppc64

```
$ xs orgs

$ xs spaces
```

Change org and space:

```
$ xs target -o <ORG> -s <SPACE>
```

Create orgs and spaces:

```
$ xs create-org
$ xs create-space
```

Deploy an Application

Like in CloudFoundry, applications are uploaded and started using the `push` command:

```
$ cd <APP_DIRECTORY>
$ xs push <app_name>
```

Check if your application is running using the command:

```
$ xs apps
```

This command will give an overview about all running apps the via which URLs they can be reached, for example:

```
Found apps:

name   requested state   instances   memory   disk
urls
--------------------------------------------------
-------------------------
myapp  STARTED           1/1         1.00GB
<unlimited>   [YOURHOST]:51000
```

Now, you should be able to use your app at http://[YOURHOST]:51000

Sample Applications

Find instructions to deploy the XSA Samples below:

node.js Hello World

XSJS Hello World

You can also deploy the sample application using the deploy service. For more information, see Deploying Sample Applications.

Scaling Application Instances

If you need more than 1 instance of an application, you can start more instances using:

```
$ xs scale <app_name> -i <number_of_instances>
```

The XSA OnPremise Runtime will then load-balance between the started instances.

Stopping Applications

All instances of an application can be stopped at once using:

```
$ xs stop <app_name>
```

You can re-start the application anytime using:

```
$ xs start <app_name>
```

In order to delete the application and it's resources permanently, use:

```
$ xs delete <app_name>
```

Creating a Buildpack

In order to use applications, you need to upload appropriate buildpacks using:

```
$ xs create-buildpack <buildpack_name>
<buildpack_directory> <buildpack_number>
```

Here you'll find a list of predeployed buildpacks.

Shutdown

In order to shut down the XSA Runtime on-premise and all running apps gently

```
$ xs shutdown
```

or press 'Enter' in the XS Controller console.

Note: DO NOT kill the XS Controller forcibly because it's not guaranteed that all app and platform processes get killed in this case.

Troubleshooting

Long Error Messages

If anything goes wrong and the information contained in the error message is not enough to tackle the problem, you may want to run the failing command with the `-verbose` parameter to get a more detailed error message:

```
$ xs <command> -verbose
```

Request Tracing

In order to see what's going on on the controller when you execute a command, you can enable tracing for a command:

```
$ xs <command> --trace
```

Log Files

Print the stdout / stderr output of your application, useful to analyse startup or runtime problems:

```
$ xs logs <app_name> --all
```

npm network/proxy errors running xs-init

If you used and older version of XSA you had you configure a proxy for npm by adding a .npmrc file to your user home directory. This is no longer necessary with the current XSA version and have to be removed.

Default Ports

If don't specify any ports at startup, the XSA OnPremise Runtime uses the following default ports:

Port	Component	Controller option	Execution Agent option
30030	Controller service port	--port	
9998	Controller system port	--sys-port	
40301-41000	EA application ports	--port-ranges	--port-ranges
8081	EA service port		--port
40000-40300	Router application ports in case port based routing is enabled	--router-port-range	
80	Router port in case domain name based routing is enabled	--router-port	

Distributed Setup

In order to start the *XSA Controller* and the *Execution Agent* on different machines, use

host1:

```
$ <XSA_ROOT>/bin/xs-controller --path
<xsa-work-directory>
```

host2:

```
$ <XSA_ROOT>/bin/xs-ea --path <xsa-work-directory>
--controller-host <host1> --controller-port 30030
```

You can start more Execution Agents on other hosts as well.

Deploy Service

You can also use the deploy service to deploy applications packaged in Multi Target Application

(MTA) archives. For more information, see Deploying Sample Applications.

Help!

There's a mailing list to which you can register to ask questions regarding XSA:

Mailing list: xs2.users@listserv.sap.corp

Register here: https://listserv.sap.corp/mailman/listinfo/xs2.users

```
.wiki-content h1 {
  padding: 2px;
  margin-top: 40px;
  margin-bottom: 30px;
  font-size: 30px;
  border-bottom-width: 1px;
  border-bottom-style:
solid;
  border-bottom-color:
#d5d5d5;
  font-weight: bold;
}
```