HOW TO use Ansible Toolkit in a centralized way

Requirements

- A Debian 8 (Jessie) or Debian 9 (Stretch) server with ROOT access
- SSL Credentials for HTTPS and LDAPS
- Flags, favicons and logos of the IdP
- IdP VM has to add to ‘debian’ user “authorized_keys” the SSH Public Key of the Ansible Master

Simple flow to install and configure a Shibboleth IdP with IDM

1. Become ROOT:
   - `sudo su -`

2. Upgrade system and install Ansible:
   - `apt-get update ; apt-get upgrade -y`
   - `apt-get install python-dev`
   - `cd /usr/local/src`
   - `wget https://bootstrap.pypa.io/get-pip.py`
   - `python get-pip.py --prefix=/usr/local`
   - `pip install --upgrade --force-reinstall ansible pyopenssl`

3. Retrieve GIT repository of the project:
   - `apt-get install git vim default-jdk --no-install-recommends`
   - `cd /opt ; git clone https://github.com/GEANT/ansible-shibboleth.git`
   - `cd /opt/ansible-shibboleth ; git clone https://github.com/GEANT/ansible-shibboleth-inventories.git inventories`
   - `cd /opt/ansible-shibboleth ; git clone https://github.com/GEANT/ans-idpcloud-utility.git scripts`

4. (Optional) Create your `.vault_pass.txt` that contains the encryption password (this is needed ONLY when you use Ansible Vault):
   - `cd /opt/ansible-shibboleth`
   - `openssl rand -base64 64 > .vault_pass.txt`

5. Download the Identity Provider source (needed to generate Shibboleth Metadata ‘credentials’):
   - `cd /usr/local/src`
   - `wget https://shibboleth.net/downloads/identity-provider/3.3.3/shibboleth-identity-provider-3.3.3.tar.gz`
   - `tar xzf /usr/local/src/shibboleth-identity-provider-3.3.3.tar.gz`
   - `rm -f /usr/local/src/shibboleth-identity-provider-3.3.3.tar.gz`

6. Generate the IdP requirements:
   - `pip install python-utils requests validators`
   - `export JAVA_HOME=/usr/lib/jvm/default-java`
   - `cd /opt/ansible-shibboleth/scripts/createIdp`
   - `python createIdp.py <FQDN> --everything`

   where <FQDN> is the Full Qualified Domain Name of your new IdP server (eg: idp.example.org)

   OPTIONAL: The default language for scrypt questions is "English". If you want to customize the language of the questions you have to change the '**en-GB**' inside "**createIdp/utils/ymlUtils.py**" to your preferred language and adapt '**utils/langUtils.py**' senteces to your needs.

7. Insert the IdP's SSL Certificate renamed into "`<FQDN>.crt`", the IdP's SSL Certificate Key renamed into "`<FQDN>.key`" and the Certification Authority certificate renamed into "`ca.crt`" inside the directory "`/opt/ansible-shibboleth/inventories/files/<FQDN>/common/ssl/`". (be sure to replace <FQDN> value with **F**ull **Q**ualified **D**omain **N**ame of the IdP)

8. (Optional) Encrypt the IdP configuration file with Ansible Vault (this is needed ONLY when you use Ansible Vault):
   - `cd /opt/ansible-shibboleth`
   - `ansible-vault encrypt inventories/production/host_vars/localhost.yml --vault-password-file .vault_pass.txt`

The ansible recipes use the languages provided by the "`idp_metadata`" dictionary.
You HAVE TO LEAVE the default language "en" and add all other languages that your IdP will support and for which you have provided the needed files ('XX' is the ISO 639-1 code of your language):

1. Logos:
   a. `inventories/files/FQDN/idp/styles/en/logo.png`
   b. `inventories/files/FQDN/idp/styles/XX/logo.png`

2. Favicon:
   a. `inventories/files/FQDN/idp/styles/en/favicon.png`
   b. `inventories/files/FQDN/idp/styles/XX/favicon.png`

3. Messages:
   a. `roles/idp/files/messages/messages_XX.properties`
   b. `roles/idp/templates/messages/custom_XX.properties.j2`
   c. `roles/idp/templates/messages/cookie_policy_XX.properties.j2`

4. Adapt Attribute Definition Default Dictionary to your needs (if you don't use a custom "**attribute-resolver.xml**" file):
   a. `roles/idp/vars/attr-defs-dict-java7.yml`
   b. `roles/idp/vars/attr-defs-dict-java8.yml`

5. Adapt the Attribute Filters for COCO and RS Entity Categories to your needs:
   a. `roles/idp/files/conf/attribute-filter-v3-rs.xml`

    b. `roles/idp/files/conf/attribute-filter-v3-coco.xml`

6. Execute this command to run Ansible on the production inventory set and to install and configure the IdP:
    a. WITH Ansible Vault:
        i. `cd /opt/ansible-shibboleth`
        ii. `ansible-playbook site.yml -i inventories/production/production.ini --vault-password-file .vault_pass.txt`
    b. WITHOUT Ansible Vault:
        i. `cd /opt/ansible-shibboleth`
        ii. `ansible-playbook site.yml -i inventories/production/production.ini`

Documentation

This section describes the behaviour of the different roles used by the Ansible toolkit.

Ansible roles available

Each role has a “`default/main.yml`” file where are configured the default values of the variables.
If no specific value will be set on the “`host_vars/<IDP_FQDN>.yml`”, roles will use the default values.

common

This role will use the variables defined under the following host_vars/<IDP_FQDN>.yml sections:
- ### Machine Variables
- ### Common Variables
- ### NTP Variables

...and perform the following actions on the machine:
1. Configure /etc/hosts with the correct FQDN
2. Configure the ‘/etc/resolv.conf’
3. Disable “manage_etc_hosts” cloud-init configuration if exists
4. Install common packages
5. Set the preferred mirror to retrieve packages
6. Configure and start NTP service
7. Set Timezone
8. Set up the server certificate used by HTTPS and LDAPS
9. Manage SWAP through a swapfile under the root (“/”) directory

apache

This role will use the variables defined under the following host_vars/<IDP_FQDN>.yml sections:
- ### Machine Variables
- ### Apache Variables

...and perform the following actions on the machine:
1. Install needed Apache2 packages
2. Add apache “*www-data*” user to "*ssl-cert*" group
3. Enable “Alias”, "SSL", ”Include”, “Negotiation”, "Headers" and "Proxy HTTP" apache modules

4. Configure and enables HTTPS with improvements about strength of Apache web server so that to obtain “**A+**” degree on: https://www.ssllabs.com/ssltest/
5. Redirect all HTTP url to HTTPS
6. Enable Apache2 localized error pages

jdk

This role will use the variables defined under the following host_vars/<IDP_FQDN>.yml sections:

- ### Common Variables (only common[‘mirror’] variable)
- ### JDK Variables

...and perform the following actions on the machine:

1. Install Open JDK (7 or 8)
2. Install Oracle JDK with JCE (7 or 8)

jetty

This role will use the variables defined under the following host_vars/<IDP_FQDN>.yml sections:

- ### JDK Variables

...and perform the following actions on the machine:

1. Install and configure Jetty Servlet Container for Java7 or Java8

openldap

This role will use the variables defined under the following host_vars/<IDP_FQDN>.yml sections:

- ### Machine Variables
- ### LDAP Variables
- ### IDP Variables (for ‘idpuser’ used only for binding and search other LDAP users)

...and perform the following actions on the machine:

1. Install and configure OpenLDAP Directory Service (package version)
2. Configure Rsyslog to redirect OpenLDAP logs into /var/log/slapd/slapd.log
3. Configure LDAP TLS (STARTTLS) with the same Certificate and Key used by Apache. Useful to update both at the same time.
4. Configure openLDAP olcLogLevel to ‘256’
5. Configure openLDAP olcSizeLimit to ‘unlimited’
6. Configure openLDAP ACL
7. Enable MemberOf and PPolicy modules
8. Add 'people' and 'groups' branches on OpenLDAP
9. Add 'idpuser' to openLDAP to perform search
10. Add 'testuser' used to perform fake login needed to Check_MK probes
11. Add a default Password Policy (see ‘add_password_policy_entry.yml’ for details)
12. Put an ldap backup script in the '/etc/cron.hourly' directory to run backup once an hour. It maintains only the last backup on the IdP.
13. Set 'slapd' log rotation
14. Restore openLDAP if requested

mysql

This role will use the variables defined under the following host_vars/<IDP_FQDN>.yml sections:

- ### Machine Variables

- ### MySQL Variables
- ### IDP Variables (for 'idpuser' used only for binding and search other LDAP users)

...and perform the following actions on the machine:

1. Install and Configure the SQL Server, MySQL (Debian Jessie) or MariaDB (Debian Stretch)
2. Put a MySQL backup script in the '/etc/cron.hourly' directory to run backup once an hour. It maintains only the last backup on the IdP.

idp

This role will use the variables defined under the following host_vars/<IDP_FQDN>.yml sections:

- ### Machine Variables
- ### Rsyslog Variables
- ### IDP Variables
- ### IDP Metadata Variables
- ### Federation Metadata URL
- ### Statistics Web App

...and perform the following actions on the machine:

1. Install a Shibboleth IdP (tested with v3.3.2) by checking the checksum to ensure that the package downloaded is the right one.
2. Enable command line to see the IdP Status (but JAVA_HOME variable has to be set before)
3. Create the persistent identifiers 'shibboleth' database
4. Load fake RS SP Metadata in the '/metadata' directory
5. Load fake CoCo SP Metadata in the '/metadata' directory
6. Allow to provide for the IdP the following customized files:
    a. `saml-nameid.properties`
    b. `global.xml`
    c. `idp.properties`
    d. `ldap.properties`
    e. `services.properties`
    f. `services.xml`
    g. `attribute-resolver.xml` (that will be renamed by Ansible into "`attribute-resolver-v3-custom.xml`" to maintain the original one)
    h. `attribute-filter.xml` (that will be renamed by Ansible into "`attribute-filter-custom.xml`" to maintain the original one)
    i. `authn/ldap-authn-config.xml`
    j. `authn/password-authn-config.xml`
    k. `c14n/subject-c14n.xml`
    l. `logback.xml`
    m. `attribute-filter-v3-coco.xml`
    n. `attribute-filter-v3-coco.xml`
    o. `audit.xml`
    p. `mvc-beans.xml`
    q. `trustCertificates.crt`
7. Download Federation's Metadata files
8. Configure Jetty to load the IdP war file
9. Configure Styles and Languages of Shibboleth IdP
10. Configure Shibboleth IdP /statistics application
11. Restore Shibboleth IdP if requested

12. Configure Rsyslog on Shibboleth IdP

phpldapadmin

This role will use the variables defined under the following host_vars/<IDP_FQDN>.yml sections:

- ### Machine Variables
- ### phpLDAPadmin Variables

...and perform the following actions on the machine:

1. Install and configure a Web Gui for the IDM part with phpLDAPadmin
2. Change the location '/phpldapadmin' into '/idm' to prevent random login attempts
3. Load logo and apache2 configuration files of phpLDAPadmin
4. Configure the user that has access to the IDM

sys-update

This role doesn't use variables

...and perform the following actions on the machine:

1. Update all packages to last version.

The inventories have different environments (production, development & test):

- `inventories/development/development.ini`
- `inventories/production/production.ini`
- `inventories/test/test.ini`

The inventory INI file decides which type of IdP configure by inserting its FQDN under the right label:

● `[Debian-IdP-without-IdM]`	(IdP without Identity Management)
● `[Debian-IdP-with-IdM]`	(IdP with Identity Management handled with phpLDAPadmin)
● `[Debian-IdP-with-IdM-GARR]`	(IdP with Identity Management improved by GARR)

The "`site.yml`" file contains the ansible playbook used to instance IdPs:

- `shib-idp-servers.yml`:

  This playbook will deploy a new Shibboleth IdP without the Identity Management part.
  IdPs that will be inserted under "`[Debian-IdP-without-IdM]`" will run this playbook.

- `shib-idp-idm-servers.yml`

  This playbook will deploy a new Shibboleth IdP with the Identity Management part handled by phpLDAPadmin.
  IdPs that will be inserted under "`[Debian-IdP-with-IdM]`" will run this playbook.

- `shib-idp-idm-servers-garr.yml`

  This playbook will deploy a new Shibboleth IdP with improvements made by GARR.
  IdPs that will be inserted under "`[Debian-IdP-with-IdM-GARR]`" will run this playbook.

Each playbook contains:

- `hosts` (a label represent who will be involved by Ansible)
- `remote_user` (the username of the user who have SSH access on the IdP server)
- `roles` (what will be applied to the IdP server)

Each “role” will contains:

- `defaults/main.yml`: where the default values of role’s variables are defined
- `tasks/main.yml`: where the starting point of the role’s operations is defined.
- `templates/*.j2`: where useful Jinja2 templates for the role are defined
- `vars/Debian.yml`: where specific (Debian architecture) variables’ value are defined

Each "`vars/`" directories contains (at least, for each role):

- `Debian.yml` (will contains all variables debian-oriented)
- `RedHat.yml` (will contains all variables redhat-oriented)

The "`host_vars/`" directory contains one `FQDN.yml` file for each server and it contains specific variables for the host into the specific environment. (These files have to be encrypted (you can do this with Ansible Vault) if shared on GitHub or somewhere other)

The "`roles/idp/vars/attr-defs-dict-java7.yml`" and "`roles/idp/vars/attr-defs-dict-java8.yml` contain all the attribute definitions supported by default on an IdP for Java 7 or 8. If you need to limit or change the default Attribute Definitions provided, you have to implement your "`idp_attrDef`" dictionary on the IdP "*FQDN.yml*" file.

The default mirror site is "`https://mi.mirror.garr.it/mirrors/debian/`". If you want to change it, add the variable "mirror" on your `inventories/#_environment_#/host_vars/FQDN.yml`.

The openLDAP logs will be stored on "`/var/log/slapd/`" directory.

The recipes can configure an IdP to be monitored through Check_MK. To be able to add the IdP hosts on the check_mk centralized server, it is needed create an automation user on check_mk server and provide its username and secret as requested by FQDN.yml To reach this, it is needed to configure `check_mk` dictionary on your `FQDN.yml` file.

The recipes offer the possibility to configure the IdP to send its logs to a Rsyslog Server through RELP protocol. To use this feature fill the rsyslog server `ip` and `port` parameters on your `FQDN.yml` file.

The recipes offer the possibility to configure the IdP to send its mysql and ldap backups to a Backups Server through RSYNC. The best way found to offer this feature on each IdP is to share a pair of SSH credentials, authorized by backups server, on all IdP and consent them to write their own backups on a specific directory named with their FQDN. The SSH credentials (SSH-CERT and SSH-KEY) HAVE TO BE generate and placed into `roles/rsync/files/ssh` renamed as README says. To use this feature fill the backups server `ip` and `remote_path` parameters on your `FQDN.yml` file, where `remote_path` is the directory on the backups server where every IdP will create their own directory and put their mysql/ldap backups.

Restore Procedures

Databases Restore

1. Retrieve database backup files from `/var/local/backups` on the IdP:
2. Put the backups file (for shibboleth and statistics database) into:
   - `inventories/files/FDQN/idp/mysql-restore/shibboleth_db.sql.gz`
   - `inventories/files/FDQN/idp/mysql-restore/statistics_db.sql.gz`
3. Set the IDP configuration variable `idp_db_restore` to `"true"` on its `host_vars` file
4. Run again the playbook

LDAP Restore

1. Retrieve LDAP backup files from `/var/local/backups` on the IdP:
2. Put the LDAP backup into:
   - `inventories/files/FDQN/openldap/restore/ldap-users.ldif.gz`
3. Set the IDP configuration variable `idp_ldap_restore` to `"true"` on its `host_vars` file
4. Run again the playbook

Useful Commands

```
--- development.ini ---
[Debian-IdP-with-IdM]
ansible-slave-2.test.garr.it

[Debian-IdP-without-IdM]
ansible-slave-1.example.garr.it
-----------------------
```

1. Test that the connection with the server(s) is working:
   - `ansible all -m ping -i /opt/ansible-shibboleth/inventories/#_environment_#/#_environment_#.ini -u debian` ("`debian`" is the user used to perform the SSH connection with the client to synchronize)
2. Get the facts from the server(s):
   - `ansible GROUP_NAME_or_HOST_NAME -m setup -i /opt/ansible-shibboleth/inventories/#_environment_#/#_environment_#.ini -u debian`
3. Examples:
   - without encrypted files: `ansible GROUP_NAME_or_HOST_NAME -m setup -i /opt/ansible-shibboleth/inventories/#_environment_#/#_environment_#.ini -u debian`
   - with encrypted files: `ansible GROUP_NAME_or_HOST_NAME -m setup -i /opt/ansible-shibboleth/inventories/#_environment_#/#_environment_#.ini -u debian --vault-password-file .vault_pass.txt`
4. ("`.vault_pass.txt`" is the file you have created that contains the encryption password)
5. Encrypt files:
   - `ansible-vault encrypt inventories/#_environment_#/host_vars/#_full.qualified.domain.name_#.yml --vault-password-file .vault_pass.txt`
6. Decrypt Encrypted files:

    - `ansible-vault decrypt inventories/#_environment_#/host_vars/#_full.qualified.domain.name_#.yml --vault-password-file .vault_pass.txt`
7. View Encrypted files:
    - `ansible-vault view inventories/#_environment_#/host_vars/#_full.qualified.domain.name_#.yml --vault-password-file .vault_pass.txt`

Authors

Original Author and Development Lead

- Marco Malavolti (marco.malavolti@gmail.com)