Building development Apache HTTPD module on Debian 8 manually

The Apache HTTP server is the Apache Software Foundation's web server. See the project's website for more details.

Contents

Apache Moonshot module information
These instructions relate to manually building the Apache Moonshot module (development version). This version provides support for exporting the SAML attributes received from the IDP as environment variables, thus making them available for the web applications (e.g. OpenStack).

All of the instructions below assume that you have root access, and will work as the root user (either directly or using sudo).

1. System Preparation

1.1. Add the Moonshot libraries.

If you have not already done so, you first need to follow the instructions on how to install the Moonshot Libraries on Debian 8.

1.2. Install some prerequisites

Building the Apache mod_auth_kerb module requires you to have several packages already installed on the machine. To install them:

```
$ apt-get install autoconf git apache2-mpm-prefork apache2-prefork-dev gcc
make libkrb5-dev
```

1.3. Build the module

We are now ready to build the Apache module.

1. Get a copy of the code via git:

   ```
   $ git clone https://bitbucket.org/umujisc/mod_auth_kerb.git
   ```

2. Enter the directory that just got created:

```
$ cd mod_auth_kerb
```

3. Run autoreconf:

```
$ autoreconf -fi
```

4. Build the software:

```
$ ./configure && make && make install
```

> **Tip**
> This will install the module to `/usr/local/etc/apache2/mods-available/auth_gssapi.load`. To get the process to put the module somewhere where Apache can find it without any intervention, run the following configure command in step 5:
>
> ```
> $ ./configure --sysconfdir=/etc
> ```

2. Installation Instructions

1. To install the Apache module, issue the following command (or create the appropriate symlinks manually):

```
$ a2enmod auth_gssapi
```

2. This module requires the Apache MPM prefork module to be active.

```
$ a2dismod mpm_event
$ a2enmod mpm_prefork
```

3. Ensure that the certificates referenced in `/etc/radsec.conf` can be read by the Apache user:

```
$ su - --shell=/bin/bash www-data
$ cat path_to_ca.pem
$ cat path_to_client.pem
$ cat path_to_client.key
```

4. Verify that the `KeepAlive` option is enabled in the Apache configuration file `/etc/apache2/apache2.conf`:

```
KeepAlive On
```

5. Restart Apache:

```
$ service apache2 restart
```

3. Configuration Instructions

Shibboleth2 Apache module incompatibility
Please note that this module is currently not compatible with the Shibboleth2 service provider Apache module. When testing or using the Moonshot module, disable the Shibboleth module and restart the webserver before attempting your test. We are attempting to resolve this problem.

3.1. Protecting a location with Moonshot

To protect a particular location on your Apache server, you must configure it with an AuthType of "Negotiate".

Example
To allow anyone with a valid Moonshot account to access `/wherever`, you would do the following:

```
<Location "/wherever">
    AuthType Negotiate
    Require valid-user
</Location>
```

3.2. Exporting GSS API attributes as environment variables

The module includes an option called *GssapiNameAttributes* that allows controlling which GSS API attributes (either SAML or RADIUS) are exported as environment variables. It is used as follows:

```
GssapiNameAttributes ENV_VAR_NAME ATTRIBUTE_NAME
```

This option can be specified multiple times, once for each attribute to expose. The Special value "json" is used to expose all attributes in a json formatted string via the special environment variable GSS_NAME_ATTRS_JSON.

Example:

```
<Location "/wherever">
    AuthType Negotiate
    Require valid-user
    GssapiNameAttributes json
    GssapiNameAttributes RADIUS_USER_NAME
urn:ietf:params:gss:radius-attribute 1
    GssapiNameAttributes EPPN urn:ietf:params:gss:federated-saml-attribute
urn:oasis:names:tc:SAML:2.0:attrname-format:uri
urn:oid:1.3.6.1.4.1.5923.1.1.1.6
</Location>
```

The special environment variable **GSS_NAME_ATTR_ERROR** is set with the GSS API returned error string in case the inquire name function fails to retrieve attributes, and with the string "0 attributes found", if no attributes are set.

In addition to this, in the event of an authentication failure, the module exports an environment variable called **MAG_ERROR** which contains one of the following values:

- *"NO_AUTH_DATA"* when the client did not send any authentication data (usually because the appropriate libraries are not installed on the browser).
- *"UNSUP_AUTH_TYPE"* when the client sent authentication data of an invalid type.
- *"GSS_MECH_ERROR"* when the GSS mechanism failed for some reason (e.g. invalid credentials).

Finally, whenever MAG_ERROR takes a value of "*GSS_MECH_ERROR"*, an additional environment variable named **GSS_ERROR_STR** is sourced. This variable contains the result of the gss_display_status() call and may help web developers to show a more appropriate error page/string to the user.

3.3. Populating REMOTE_USER

Web services often rely on the `REMOTE_USER` Apache environment variable for user information, such as a local user account or a pseudonymous identifier.

To populate `REMOTE_USER`, update the reply from the RP Proxy with the `User-Name` RADIUS attribute in the RP Proxy's `post-auth` section:

```
update reply {
        User-Name := "content"
}
```

3.4. HTTPS Internet Explorer compatibility

For updated best practice with Internet Explorer connections, you should also read Microsoft's HTTPS and Keep-Alive Connections article.