Tequila

Managing your Tequila server.

Claude Lecommandeur

EPFL - KIS

claude.lecommandeur@epfl.ch

Table of Contents

Overview

The Tequila server is written in Perl. Hence, it inherits the simplicity and sheer beauty of this language. Since Tequila does a little bit of crypto, it need a few Perl modules, all easily obtainable on CPAN.

Managing a Tequila server is not difficult at all. This is just a CGI script after all. For performance reason it is advised to use Apache mod_perl to execute it, but it is not necessary, ordinary CGI will do.

In all this document we will suppose that The Tequila server is installed in the /var/www/tequila/ directory, and we are using mod_perl in /var/www/tequila/perl/. If it is not the case for you, adjust the file names to fit your configuration.

Generally, the Tequila server administration is just editing text files, except for policies files where you can use the Tape tool.

If you are running a stand alone server, without partner servers, and you don't use the resources feature, there is not much to do, just maintaining your authentication and attributes connectors.

General principles

There are 3 actors in the Tequila comedy. A people with a browser (**User**), tries to access a protected Web application (**Application**). The Client uses the service of a Tequila server (**Tequila**) to authenticate the User and get some information about him. The scenario is the following :

1. **User** access an URL on **Application**.
2. **Application** tries to find a key value either in the URL or in a cookie.
3. If there is no key, **Application** calls an URL on **Tequila** with the 'createrequest' command. It sends along the description of the request. If **Application** is a resource (see below), it just sends its resource name. In the other case, it sends all the characteristics of the request. There is many of them, the most important are the URL where to redirect the user after authentication (urlaccess), the attributes values the client wants to know (request), the service name (service), etc...
4. **Tequila** stores the attributes of the request and responds with a random hexadecimal request key.
5. **Application** reads this key and redirects **User'**s browser to **Tequila** URL plus the request key.
6. **Tequila** recognize the key, fetch the corresponding authentication request and does what it is meant to do, authenticate **User**, fetch attributes values about him, check if constraints are met, and if everything is OK, redirects **User** to 'urlaccess' plus the request key as a parameter. Optionally Server deposit a specially crafted cookie it can recognize later in **User**'s browser.
7. **Application** will see **User** comes back, but this time, it finds the request key it was waiting for.
8. It first tries to see it there is a session already opened for this key.
9. If there is, **User** is given access to the services of **Application**.
10. But just after **User** returns from **Tequila**, there is no such session. So, **Application** calls another URL on **Tequila** with the command 'fetchattributes', along with the request key.
11. **Tequila** sends back the values of the attributes relative to **User** (username, name, etc...).
12. **Application** creates a Session for **User** with all the attributes values stored in it for future use.
13. **Application** also set a cookie in **User'**s browser with a session key.

Setting up resources

Resources are clients specially trusted by the server. Everything from the resource is known to the server. This include the URL where to redirect the user after authentication (urlaccess), the attributes values the client wants to know, the service name, etc...

When the resource create a new request with the 'createrequest' command, it justs sends its resource name. Server already knows everything about it.

In addition, Server will fully authenticate Client with SSL. For this, Client presents its SSL certificate, and Server checks it corresponds to what is described in the resource description. Namely, Server will check that the subject of the certificate matches some regular expression, that the certificate issuer organization matches some other regexp, and that the host emitting the create request is one of the allowed host.

Each resource has its description in a file in /var/www/tequila/Tequila/Resources/. The following keywords are allowed :

Description: *resource_description*

The description that will be displayed to the user. Try to be explicit.

SubjectMatch: *regexp*

The regular expression the subject of the SSL certificate presented by the application should match. For example :

```
SubjectMatch:   ^testresource Tequila resource$
```

IssuerOrgMatch:

The regular expression the organization of the issuer of the SSL certificate presented by the application should match. For example :

```
IssuerOrgMatch: ^EPFL$
```

Allowedhosts: *host1 host2 ...*

The list of hosts that are allowed to use the resource.

```
Allowedhosts:    host1.epfl.ch host2.epfl.ch
```

Service: *service_name*

The string at the top of login screen. Try to be even more explicit.

Contact: *resource_owner_email*

Try to keep up-to-date.

Request: *attr1 attr2 ...*

List of attributes the application is willing to get from the server.

Allows: *allows_filter*

Used to lift default server restriction (defined with the Restrict directive). For example, if you are configured to interface with Shibboleth, but it is not visible by default, a resource that wants to have it by default will say :

```
Alows: userclass=shibboleth
```

Language: *language*

Default language for the user interface.

Urlacces: *url*

Where to redirect the user after successful authentication.

Setting up Tequila partners

Partners are Tequila independent servers that fully trust themselves. A Tequila server manage a set of users and a set of secured services. If two servers are partner, users managed by one server can access services managed by the other.

To achieve this, each Tequila server can act as a client of all others servers. It does a complete transaction with the target server sot it has a proof of the user's identity and the values of asked attributes, it then forwards these values to the actual client.

Servers must trust themselves and be sure to speak to the right server. Therefore, it is mandatory to use encrypted and authenticated dialog between partners. Each partner must know information on the partner certificate and trust the certification authority (CA) that signed it.

Each trusted partner is described in a file in the /var/www/tequila/Tequila/Partners/ directory. The format of this file is the following :

ShortName: *short_string*

Short name the partner will referred with. It is displayed to end users, at authentication and when setting attributes release policy.

LongName: *long_string*

Same, but more descriptive.

Contact: *email_address*

Host: *host_of_the_partner*

FQDN of the partner. eg. tequila.acme.com

Domain: *domain_of_the_partner*

Internet domain. eg. acme.com

URL: *url_of_the_partner_server*

eg. https://tequila.acme.com/tequila

SubjectMatch: *regexp*

The subject in the partner SSL certificate must match this regular expression.

IssuerOrgMatch: *regexp*

The organization of the partner SSL certificate must match this regular expression.

Dealing with SSL certificates

The whole trust infrastructure inside Tequila is built on top of SSL.

The Shibboleth interface

Shibboleth interface

Servers farm

Files

Binaries

The default Tequila distribution comes with 2 binaries :

```
/var/www/tequila/perl/tequila
```

The main binary. Will be called by client applications and client users.

```
/var/www/tequila/perl/tape
```

Tape stands for Tequila Attributes Policy Editor. It is used to manage the way the Tequila server releases user attributes,

There is also a set of so-called ‘connectors’, there is 2 types of connectors, Authentication connectors, and Data connectors. Data connectors are in charge of retrieving user attributes values. Standard connectors are :

```
/var/www/tequila/perl/Tequila/AuthConnector.pm
/var/www/tequila/perl/Tequila/DataConnector.pm
/var/www/tequila/perl/Tequila/LdapAuthConnector.pm
/var/www/tequila/perl/Tequila/LdapDataConnector.pm
/var/www/tequila/perl/Tequila/TestAuthConnector.pm
/var/www/tequila/perl/Tequila/TestDataConnector.pm
```

Configuration files

Always present files :

```
/var/www/tequila/Tequila/Tequila.conf
/var/www/tequila/Tequila/AttributesTranslations.conf
/var/www/tequila/Tequila/Messages.conf
/var/www/tequila/Tequila/Policies.conf
```

Sometimes present, depending on the configuration used. These files are used for configuring Connectors ans Plugins :

```
/var/www/tequila/Tequila/Shibboleth.conf
/var/www/tequila/Tequila/LdapAuthConnector.conf
/var/www/tequila/Tequila/LdapDataConnector.conf
/var/www/tequila/Tequila/AuthDigest.conf
```

Directories :

```
/var/www/tequila/Tequila/Partners
/var/www/tequila/Tequila/Resources
```

```
/var/www/tequila/Tequila/UsersPolicies
/var/www/tequila/Tequila/Requests
/var/www/tequila/Tequila/ServerSessions
/var/www/tequila/Tequila/ssl
```

We will now examine these files in detail.

Tequila.conf

This is the main configuration file.

Each line has the form :

keyword: *value* [*value value* etc...]

Comments begin with a sharp character.

Possible keywords are :DataConnector:

Organization: *your_organization_name*

Your organization name or acronym. eg. EPFL. Mandatory.

Server: *server_fqdn*

The FQDN of the server. eg. tequila.epfl.ch. Mandatory

Domain: *your_domain_name*

The local internet domain name. eg. epfl.ch. Mandatory

ManagerEmail: *email_of_the_server_manager*

The email address of the person managing this server. Mandatory

ManagerUsername: *username_of_the_server_manager*

The username (the Tequila username) of the manager. It is used by the Tape tool to configure the server attribute release policy.

SessionManager: *session_manager_name*

Since version 2.0 Tequila can use a session manager that is in charge with managing sessions. Default is no sessions manager, that means, store sessons in local files.

UseCookies: *on|off|optional*

Should we set browser cookies to store already authenticated users. If ‘on’, always set cookies, if ‘off’ never set cookies, if ‘optional’ the user decides itself if she want cookies.

CookiePolicy: *session|persistent*

Policy of the cookie set by the Tequila server.

SessionDuration: *duration_in_hours*

Duration during which the server cookie will be valid. Default is 12 hours.

AcceptCertificates: *on|off*

If value is 'on', tequila will propose SSL client authentication to its clients. In this case remember to have

tequilac linked to tequila in your server cgi-bin.

SSLCertificateFile: *filename*

Where your server SSL certificate file is. Used only when your server trusts another Tequila server and needs to authenticate to it.

SSLKeyFile: *filename*

Where your server SSL key file is. Used only when your server trusts another Tequila server and needs to authenticate to it.

UserClassAttribute: *attribute_name*

This attribute will have a special meaning within Tequila. Default value is 'userclass' . Some of the values of this attributes are 'unknown', 'loginfail', 'noaccess' or 'shibboleth'. See below for detailed explanation.

DefaultCharset: *charset*

Default character set for clients. Default is 'utf8',

Restrict: *attribute_filter*

Impose a server wide restriction on attributes. Users with attributes not matching this restriction will be ignored (eg. Will be unknown users).

```
Restrict: userclass=EPFL|EHE
```

AllowsAnonymous: *domains_allowed_to_use_us*

List of Internet domains allowed to authenticate with this server. Domains are separated with spaces. They are justs the firsts characters of Internet address. Incoming clients will be allowed only it their address begins with one of these domains. Value 'all' means everybody and his sister is allowed.

DefaultIdentities: *one|any*

Tequila can handle usernames that correspond to multiple identities. If 'one' is specified, when a user authenticate she will be asked which identities she chooses to use. In case of 'any', she won't be asked, any identity will be used, generally the first one.

UserCanOverridePolicy: *on|off*

Can users define their own attribute release policy.

FixedPolicy: *list_of_attributes_with_fixed_policy*

In case UserCanOverridePolicy is set, this is a list of attributes which the user can't override server wide policy.

AlwaysConfirmUser: *on|off*

When a user is authenticated vie a cookie previously set by us, should we asked for login confirmation. Default is 'off'.

AllowsUnknownUsers: *on|off*

Tequila has the ability to allow unknown users login if the client application asks for it. In this case the pair (username, password) is always accepted if username is unknown. The actual authentication os left to the

client application. This feature that can be dangerous when misused can be enables here. Default of off.

SoftwareKeyboard: *on|off*

The login interface ca present an on screen keyboard for users that suspect a key logger to be active on their computer. Default is on.

AuthConnector: *authconnector_name*

The name of the authentication connector module. This is the name of the Perl package that implements the AuthConnector interface. At startup, the Tequila server loads Tequila::*authconnector_name.* So you should arrange for this package to be in the @INC path.

DataConnector: *dataconnector_name*

The name of an authentication connector module. This is the name of the Perl package that implements the DataConnector interface. At startup, the Tequila server loads Tequila::*dataconnector_name.* So you should arrange for this package to be in the @INC path. You can have as many DataConnector line as you want.

LoadPlugin: *plugin_name*

The name of a plugin module. This is the name of the Perl package that implements the Plugin. At startup, the Tequila server loads Tequila::plugin*_name.* So you should arrange for this package to be in the @INC path. You can have as many LoadPlugin line as you want.

DoWAYF: *on|off*

This is part of the Shibboleth interface. If you are running it, this directs Tequila to implement itself the WAYF role.

Example :

```
#
# EPFL Tequila server Configuration.
#
Organization: EPFL
#
Server: tequila.epfl.ch
#
Domain: epfl.ch
#
ServerManager: claude.lecommandeur@epfl.ch
#
AuthConnector: LdapAuthConnector
#
DataConnector: LdapDataConnector
#
Restrict: categorie=EPFL|EHE
#
UserClassAttribute:     categorie
SoftwareKeyboard:       on
UserPolicy:             on
ConfirmAllAttrs:        off
AlwaysConfirmUser:      off
#AllowsUnknownUsers:    on
Allowsanonymous:        128.178.
```

```
LoadPlugin:             AuthDigest
#
```

rc4key

This file contains the main secret key for the server. Take great care of it. It should be readable by your https server and no one else. The RC4 key is generally 16 characters long.

Messages.conf

The messages translations file. Line have the form :

```
keyword.language1:        string_in_language1
keyword.language2:        string_in_language2
keyword.language3:        string_in_language3
```

For example :

```
title.fr:               Login pour le service
title.en:               Login for the service
title.de:               Login für den Dienst
```

You should modify this file only to add or modify translations. Some are not that perfect.

AttributesTranslations.conf

This file contains the user attributes names translations. There is 3 kind of entries :

SupportedLanguages: *lg1 lg2 ...*

Each language name is the ISO 639 name of the language, for example, fr, en, de, ...

DefaultLanguage: *lgcode*

Default language for the user interface when there is no other way to determine it (via HTTP_ACCEPT_LANGUAGE for example). Default is ‘en’.

Attribute: *name Name Nom Nachname*

Attribute names translations in the supported languages. You should have one Attribute line per attribute. If an attribute has no translation, its name is displayed in the user interface.

LdapAuthConnector.conf

Configuration of the LDAP authentication connector. The format of the lines is :

URL: *ldap_url*

The LDAP URL has the form :

ldaps://ldapserver/search_base?filter

Example :

```
#
# Configuration of the LDAP authentication connector for Tequila.
#
URL: ldaps://ldap.epfl.ch/o=epfl,c=ch?sub
URL: ldaps://ldap.epfl.ch/o=ehe,c=ch?sub
URL: ldaps://ldap.epfl.ch/o=epfl-guests,c=ch?sub
URL: ldaps://ldap.epfl.ch/o=epfl-old,c=ch?sub
#
```

You can use as many URL lines as you want and you can use ldap protocol instead of ldaps.

LdapDataConnector.conf

Configuration of the LDAP data connector. The format of the lines is :

keyword: *value [value value etc...]*

The possible keywords are :

URL : *ldap_url*

Same as URL lines in LdapAuthConnector.conf.

Supports : *list_of_attributes*

List of attributes supported by the connector. Use the Tequila name, not the LDAP name. See Mapping.

Mapping : *tequila_attribute ldap_attribute*

Mapping of names between Tequila names and LDAP names. Use as many Mapping lines as necessary.

Example :

```
#
# LDAP Data connector configuration for Tequila at EPFL.
#
URL: ldap://ldap.epfl.ch/o=epfl,c=ch?sub
URL: ldap://ldap.epfl.ch/o=ehe,c=ch?sub
URL: ldap://ldap.epfl.ch/o=epfl-guests,c=ch?sub
URL: ldap://ldap.epfl.ch/o=epfl-old,c=ch?sub
#
#
Supports: name firstname email title unit office phone \
          username uniqueid unixid groupid
#
Mapping: name         sn
Mapping: firstname    givenname
Mapping: email        mail
Mapping: title        title
Mapping: unit         ou
Mapping: office       roomNumber
Mapping: phone        phone
Mapping: username     uid
Mapping: uniqueid     uniqueIdentifier
Mapping: unixid       uidnumber
Mapping: groupid      gidnumber
#
rc4key
```

Just the key itself. About 8 characters is enough. Necessary only if you want to use the cookie option.

privkey

The X509 private key of the server. Necessary only if you are using the RSA option. You can generate the private key with :

```
openssl req -new -nodes -keyout tequila.key -out tequila.csr
```

And then self-sign it with :

```
openssl x509 -req -in tequila.csr -signkey tequila.key -out tequila.crt
```

crts/.crt*

The X509 certificates of the servers in the Tequila network. Necessary only for servers using the RSA option. The name of the file is : *orgname.crt*