Actualización a SimpleSAMLphp 1.10 de IdPs y SPs

1.0

Sixto Martin

December 14, 2012

ÍNDICE GENERAL

En esta documentación se explica como actualizar una instancia (ya sea un IdP o un SP) desde la anterior versión de simpleSAMLphp (1.8.0) a la versión de finales de 2012 (1.10.0)

CAPÍTULO

ONE

INTRODUCCIÓN

El pasado Junio del 2011 tuvieron lugar las actualizaciones de las instancias de simpleSAMLphp de los IdPs y SPs a la versión 1.8.0. Desde entonces han sido muchas las mejoras, las nuevas funcionalidades y los bugs corregidos.

Es importante destacar una serie de fallos de seguridad que han sido corregidos desde entonces:

- Parche contra ataques XSS (introducido en la versión 1.8.2) [r3009]
- Parche contra ataques al cifrado PKCS 1.5 de las aserciones (introducidos en las versiones 1.9.1 y 1.9.2) [r3132]

1.1 ¿Es obligatorio el uso de la última versión de simpleSAMLphp 1.10 en CONFIA?

Resumen: En IdPs es obligatorio. En SPs es MUY recomendable.

1.1.1 IdPs

El uso de la última versión de simpleSAMLphp para los IdPs es obligatorio en el entorno de CONFIA. Esto se debe a que los IdPs estarán conectados al SP shibboleth 2.4.3 utilizado para federar el software ILIAS y dicho software es incompatible con versiones anteriores a la 1.10 de simpleSAMLphp. (Para este curso es necesario actualizar el SP shibboleth a la 2.4.3 o posteriores debido a que las anteriores versiones tienen problemas de seguridad)

1.1.2 SPs

Desde CONFIA recomendamos encarecidamente que siempre se tengan las instancias de los SP actualizadas a la última versión, sin embargo, somos conscientes de que a pesar de que simpleSAMLphp funciona con versiones de php 5.2 o 5.3 no siempre ocurre lo mismo con versiones del software al que se federa y ello puede acarrearnos un trabajo extra.

Pongamos un ejemplo real de un conflicto, tenemos el siguiente entorno:

- PHP 5.1
- Moodle 1.9, que no es compatible con PHP 5.3
- SimpleSAMLphp 1.5

Si queremos utilizar la ultima versión de simpleSAMLphp, la 1.10, deberemos de actualizar el php a la versión 5.2 y no a la 5.3 porque sino nos dejaría de funcionar el Moodle. Pero nos encontramos con que cuando vamos a actualizar el sistema operativo, la versión de php oficial que soporta es la 5.3 y no la 5.2. Por lo que habrá que hacer un trabajo extra para conseguir un entorno con php 5.2 en el que tanto Moodle 1.9 como simpleSAMLphp funcionen.

CAPÍTULO

TWO

GUÍA DE ACTUALIZACIÓN DEL IDP A LA VERSION 1.10.0

2.1 Pasos

Vamos a actualizar la versión del IdP para que utilice simplesamlphp 1.10.0

2.1.1 Fuentes

Supongamos que tenemos la siguiente estructura:

```
# directorio de simplesamlphp
/var/www/simplesamlphp
```

Descargamos del subversion la nueva versión de simplesamlphp

```
cd /var/www
svn co http://simplesamlphp.googlecode.com/svn/tags/simplesamlphp-1.10.0 simplesamlphp1.10
```

Desde la versión 1.8 se han añadido numerosas funcionalidades y mejoras y corregidos fallos de seguridad. Podemos ver un resumen en el changelog de simpleSAMLphp. Destacamos la corrección de ataques XSS en la 1.8.2, evitar ataques contra el cifrado PKCS 1.5 de las aserciones en las versiones 1.9.1 y 1.9.2. También destacar que los SP de shibboleth 2.4.3 requiren un IdP de simpleSAMLphp 1.10.

Varias cosas que tenemos que tener en cuenta:

- Desde la versión 1.6.0 de simplesamlphp se requiere una versión de php > 5.2.0 , recomendamos la 5.2.12. (Al actualizar la versión de php tener en cuenta que el resto del software que utilizáis sea compatible con dicha versión, se dió el caso de que moodle 1.9 no era compatible con ciertas versiones de php 5.3.X.
- Desde la versión 1.6.0 de simplesamlphp se utilizan archivos json para las traducciones lo cual significa que si tenemos módulos propios que hacen uso de diccionarios deberemos construir los archivos json correspondientes.
- Ha habido cambios en el archivo de configuración de simplesamlphp (`config.php`). Algunas de las variables del fichero `config.php` han cambiado. Por ahora los cambios son compatibles hacia atrás pero conviene tener en cuenta:
- Desde la 1.7 se tiene la variable *'session.cookie.lifetime'* para especificar cuando deben caducar las cookies.
- Para obligar que los accesos sean HTTPS desde la 1.7 hacemos uso de: *'session.cookie.secure'* y *'session.phpsession.httponly'*.
- La variable *'session.phpsession.limitedpath'* ha dejado de utilizarse desde la 1.7 y ahora se hace uso de *'session.cookie.path'*.

- La variable *'session.handler'* desde la 1.7 ha sido substituida por *'store.type'* y se ha añadido el backend sql para poder guardar las sesiones en base de datos. En el caso de establecer el *'store.type'* al valor 'sql' hay que establecer el valor de las variables: *'store.sql.dsn'*, *'store.sql.username'*, *'store.sql.password'* y *'store.sql.prefix'*.
- Desde la 1.8 se dispone de la variable *'proxy'* para poder especificar un proxy. (Ojo porque antiguamente los que necesitaban salir con un proxy utilizaban otra variable llamada *'saml_proxy'* proveniente de un parche implementado en CONFIA)

2.1.2 Configuración

Deberemos de copiar las configuraciones del antiguo simplesamlphp, los metadatos y el certificado.

```
# Ojo cuando se copien archivos simbólicos que puede haber problemas
cp -a /var/www/simplesamlphp/cert/* /var/www/simplesamlphp1.10/cert
cp -R /var/www/simplesamlphp/metadata/* /var/www/simplesamlphp1.10/metadata
cp /var/www/simplesamlphp/config/* /var/www/simplesamlphp1.10/config
```

Habilitamos los permisos de los directorios metadata y log para que el apache pueda escribir sobre dichos directorios (mirar permisos del antiguo simplesamlphp y ponerle los mismos). Ejemplo:

```
# En maquinas debian usar www-data:www-data en lugar de apache:apache
chown -R apache:apache /var/www/simplesamlphp/log
chown -R apache:apache /var/www/simplesamlphp/metadata
```

Editamos el fichero de configuración de simplesamlphp (`/var/www/simplesamlphp1.10/config/config.php`) y comprobamos el valor de los siguientes parámetros:

```
'session.cookie.lifetime' => 0,         // Si queremos que no caduque la cookie
'session.cookie.secure' => TRUE,        // 'TRUE' si queremos 'cifrar las cookies' (solo se cifran si
'session.phpsession.httponly' => FALSE, // Con valor a 'TRUE' evita que APIs non-HTTP como por ejempl

'store.type' => 'phpsession',    // o 'memcache' o 'sql', dependiendo de como queremos guardar la ses
                                 // (antiguamente el parametro se llamaba 'session.handler')
```

Nota: Se ha comprobado que se producen errores si en el config.php se especifican las variables *'session.cookie.path'* y *'session.phpsession.limitedpath'* a la vez. La variable *'session.phpsession.limitedpath'* está deprecada desde la versión 1.8 y debe ser eliminada si queremos hacer uso de la variable *'session.cookie.path'*.

Es importante indicar los datos de la organización y el contacto. Debemos editar el archivo con los metadatos de nuestro IdP y rellenar convenientemente los siguientes atributos en el fichero `/var/www/simplesamlphp1.10/metadata/saml20-idp-hosted.php`:

```
'OrganizationName' => array(
    'en' => 'IdP-Homeless CONFIA',
    'es' => 'IdP-Homeless CONFIA',
),
'OrganizationDisplayName' => array(
    'en' => 'IdP-Homeless of CONFIA',
    'es' => 'IdP-Homeless de CONFIA',
),
'OrganizationURL' => array(
    'en' => 'http://confia.aupa.info/',
    'es' => 'http://confia.aupa.info/',
),
```

Los datos de contacto del administrador del Proveedor de Identidad se especifican en el archivo `/var/www/simplesamlphp1.10/config/config.php` en los parámetros (*'technicalcontact_name'*,

'technicalcontact_email'). Es importante que estos datos sean correctos y que correspondan a los encargados de los proveedores de identidad ya que serán distribuidos como parte de los metadatos de la entidad.

En el marco de CONFIA en el WAYF se tiene habilitado el módulo attributevalidator que valida que todos los datos que se envíen del usuario desde el IdP son correctos y son todos los que deben de ser. En caso contrario se le muestra una pantalla de error al usuario, donde puede reportar del problema al administrador de su proveedor de identidad. Por defecto se enviará un mail al correo de contacto que aparece en los metadatos de ese IdP.

Si el encargado del las fuentes de los datos es distinto que el responsable del IdP se debe de notificar de este hecho al administrador de CONFIA para que el registre la dirección de de contacto adicional en el módulo y los correos de reporte de errores lleguen a ambos (Al administrador del IdP para que sea consciente del problema y al encargado de las fuentes de datos para que lo solucione).

Nota: Existe la posibilidad de configurar también en el IdP el filtro de attributevalidator para que si los datos no son válidos ni siquiera salgan del IdP.

2.1.3 Módulos

Ahora lo que nos queda es comprobar que módulos estaban habilitados en el anterior simplesamlphp que debemos habilitar en este. Si el directorio estaba subversionado rápidamente podremos observarlo ejecutando el comando:

```
cd /var/www/simplesamlphp
svn st
```

y viendo que temas son específicos de nuestra instancia y cuales están activados.

Como mínimo deberemos habilitar los modulos *metarefresh* y *cron*:

```
touch /var/www/simplesamlphp1.10/modules/metarefresh/enable
touch /var/www/simplesamlphp1.10/modules/cron/enable
```

Es posible que tengamos que hacer también uso del módulo courses y attributecollector para lo cual simplemente deberemos de copiarlo del antiguo simplesamlphp al nuevo.

```
cp -R /var/www/simplesamlphp/modules/courses /var/www/simplesamlphp1.10/modules/
cp -R /var/www/simplesamlphp/modules/attributecollector /var/www/simplesamlphp1.10/modules/
```

Lo mismo se haría con los módulos para temas, fuentes de autenticación y demás módulos específicos que pudiéramos estar utilizando.

2.1.4 Parches

Tenemos que aplicar una serie de parches a esta versión de simpleSAMLphp, los descargamos del repositorio en una carpeta:

```
svn co https://confia.aupa.info/svn/confia/trunk/ssp/updates/
```

Creamos un directorio *patches* en el directorio temporal y copiamos ahí los parches correspondientes a la versión de simplesamlphp1.10:

```
mkdir /tmp/patches
cp updates/simplesamlphp1.10/patches/*.diff /tmp/patches
```

Aplicamos todos los parches sobre el simplesamlphp1.10

```
cd /var/www/simplesamlphp1.10
for patch in  /tmp/patches/*.diff; do patch -p0 < $patch; done
```

2.1.5 Sustitución

Una vez realizado todo lo anterior ya podemos sustituir la nueva instancia por la antigua:

```
mv /var/www/simplesamlphp/ /var/www/simplesamlphp_old/
mv /var/www/simplesamlphp1.10 /var/www/simplesamlphp/
```

Y comprobaríamos si todo funciona correctamente. Si no funciona siempre podremos volver a la anterior versión deshaciendo el anterior renombrado de directorios

Puede que algún filtro o algún módulo especifico que se implementara ajeno a CONFIA no funcione con la nueva versión de simplesamlphp por lo que habrá que hacer un testeo exhaustivo de que todo funciona correctamente.

2.2 Actualización del certificado en el IdP

Desde la versión 1.7 se da soporte a que simplesamlphp trabaje a la vez con 2 certificados, para evitar que durante el periodo de propagación del nuevo certificado por la federación se deje de dar servicio.

El proceso sería el siguiente:

Primero editamos el archivo con los metadatos de nuestro IdP, que será `metadata/saml20-idp-hosted.php`, añadiendole 2 nuevos atributos:

```
'new_privatekey' => 'new.pem',
'new_certificate' => 'new.crt',
```

donde *'new.pem'* y *'new_crt'* corresponden a los ficheros alojados en el directorio *cert* que poseen el nuevo certificado y la nueva key.

Una vez realizado esto, el IdP pasará a exportar 2 metadatos válidos. Los metadatos la próxima vez que el gestor de metadatos se conecten a ellos serán leidos, almacenados y posteriormente distribuidos al resto de elementos de la federación.

En un plazo de 4 horas podremos estar seguro de que los elementos han sido distribuidos a todos los elementos del sistema por lo que podremos volver a configurar los datos de nuestro IdP para que ahora únicamente utilice el nuevo certificado y la nueva key. Para ello volvemos a editar los metadatos de nuestro IdP, alojados en `metadata/saml20-idp-hosted.php`, y eliminaremos la entrada que añadimos antes:

```
'new_privatekey' => 'new.pem',
'new_certificate' => 'new.crt',
```

Y cambiaremos la entrada correspondiente con los antiguos certificado y clave:

```
'privatekey' => 'old.pem',
'certificate' => 'old.crt',
```

por los nuevos:

```
'privatekey' => 'new.pem',
'certificate' => 'new.crt',
```

Fuente original de la documentación del cambio de certificado

CAPÍTULO

THREE

GUÍA DE ACTUALIZACIÓN DEL SP A LA VERSION 1.10.0

3.1 Pasos

Vamos a actualizar la versión del SP para que utilice simplesamlphp 1.10.0

3.1.1 Fuentes

Supongamos que tenemos la siguiente estructura:

```
# directorio de simplesamlphp
/var/www/simplesamlphp
```

Descargamos del subversion la nueva versión de simplesamlphp

```
cd /var/www
svn co http://simplesamlphp.googlecode.com/svn/tags/simplesamlphp-1.10.0 simplesamlphp1.10
```

Desde la versión 1.8 se han añadido numerosas funcionalidades y mejoras y corregidos fallos de seguridad. Podemos ver un resumen en el changelog de simpleSAMLphp. Destacamos la corrección de ataques XSS en la 1.8.2, evitar ataques contra el cifrado PKCS 1.5 de las aserciones en las versiones 1.9.1 y 1.9.2. También destacar que los SP de shibboleth 2.4.3 requiren un IdP de simpleSAMLphp 1.10.

Varias cosas que tenemos que tener en cuenta:

- Desde la versión 1.6.0 de simplesamlphp se requiere una versión de php > 5.2.0 , recomendamos la 5.2.12. (Al actualizar la versión de php tener en cuenta que el resto del software que utilizáis sea compatible con dicha versión, se dió el caso de que moodle 1.9 no era compatible con ciertas versiones de php 5.3.X.
- Desde la versión 1.6.0 de simplesamlphp se utilizan archivos json para las traducciones lo cual significa que si tenemos módulos propios que hacen uso de diccionarios deberemos construir los archivos json correspondientes.
- Ha habido cambios en el archivo de configuración de simplesamlphp (`config.php`). Algunas de las variables del fichero `config.php` han cambiado. Por ahora los cambios son compatibles hacia atrás pero conviene tener en cuenta:
- Desde la 1.7 se tiene la variable *'session.cookie.lifetime'* para especificar cuando deben caducar las cookies.
- Para obligar que los accesos sean HTTPS desde la 1.7 hacemos uso de: *'session.cookie.secure'* y *'session.phpsession.httponly'*.
- La variable *'session.phpsession.limitedpath'* ha dejado de utilizarse desde la 1.7 y ahora se hace uso de *'session.cookie.path'*.

- La variable *'session.handler'* desde la 1.7 ha sido substituida por *'store.type'* y se ha añadido el backend sql para poder guardar las sesiones en base de datos. En el caso de establecer el *'store.type'* al valor 'sql' hay que establecer el valor de las variables: *'store.sql.dsn'*, *'store.sql.username'*, *'store.sql.password'* y *'store.sql.prefix'*.
- Desde la 1.8 se dispone de la variable *'proxy'* para poder especificar un proxy. (Ojo porque antiguamente los que necesitaban salir con un proxy utilizaban otra variable llamada *'saml_proxy'* proveniente de un parche implementado en CONFIA).

3.1.2 Configuración

Deberemos de copiar las configuraciones del antiguo simplesamlphp, los metadatos y el certificado

```
# Ojo cuando se copien archivos simbólicos que puede haber problemas
cp -a /var/www/simplesamlphp/cert/* /var/www/simplesamlphp1.10/cert
cp -R /var/www/simplesamlphp/metadata/* /var/www/simplesamlphp1.10/metadata
cp /var/www/simplesamlphp/config/* /var/www/simplesamlphp1.10/config
```

Habilitamos los permisos de los directorios metadata y log para que el apache pueda escribir sobre dichos directorios (mirar permisos del antiguo simplesamlphp y ponerle los mismos). Ejemplo:

```
# En máquinas debian usar www-data:www-data en lugar de apache:apache
chown -R apache:apache /var/www/simplesamlphp/log
chown -R apache:apache /var/www/simplesamlphp/metadata
```

Editamos el fichero de configuración de simplesamlphp (`/var/www/simplesamlphp1.10/config/config.php`) y comprobamos el valor de los siguientes parámetros:

```
'session.cookie.lifetime' => 0,         // Si queremos que no caduque la cookie
'session.cookie.secure' => TRUE,        // 'TRUE' si queremos 'cifrar las cookies' (solo se cifran si
'session.phpsession.httponly' => FALSE, // Con valor a 'TRUE' evita que APIs non-HTTP como por ejempl

'store.type' => 'phpsession',    // o 'memcache' o 'sql', dependiendo de como queremos guardar la ses
                                 // (antiguamente el parametro se llamaba 'session.handler')
```

Nota: Se ha comprobado que se producen errores si en el config.php se especifican las variables *'session.cookie.path'* y *'session.phpsession.limitedpath'* a la vez. La variable *'session.phpsession.limitedpath'* está deprecada desde la versión 1.8 y debe ser eliminada si queremos hacer uso de la variable *'session.cookie.path'*.

Es importante indicar los datos de la organización y el contacto. Debemos editar el archivo con los metadatos de nuestro IdP y rellenar convenientemente los siguientes atributos en el fichero `/var/www/simplesamlphp1.10/config/authsources.php`:

```
'OrganizationName' => array(
    'en' => 'Test-SP CONFIA',
    'es' => 'SP-Pruebas CONFIA',
),
'OrganizationDisplayName' => array(
    'en' => 'Test-SP of CONFIA',
    'es' => 'SP de Pruebas de CONFIA',
),
'OrganizationURL' => array(
    'en' => 'http://confia.aupa.info/',
    'es' => 'http://confia.aupa.info/',
),
```

Los datos del contacto del administrador del Proveedor de Servicio se especifican en el `/var/www/simplesamlphp1.10/config/config.php` (*'technicalcontact_name'*, *'technicalcontact_email'*). Es importante que sean correctos y que se correspondan a los encargados de dichas entidades.

3.1.3 Módulos

Ahora lo que nos queda es comprobar que módulos estaban habilitados en el anterior simplesamlphp que debemos habilitar en este. Si el directorio estaba subversionado rápidamente podremos observarlo ejecutando el comando:

```
cd /var/www/simplesamlphp
svn st
```

y viendo que temas son específicos de nuestra instancia y cuales están activados.

Como mínimo deberemos habilitar *metarefresh*, *cron*:

```
touch /var/www/simplesamlphp1.10/modules/metarefresh/enable
touch /var/www/simplesamlphp1.10/modules/cron/enable
```

Es posible que tengamos que hacer también uso de ciertos módulos de temas, filtros, o conectores. Para ello habrá que añadirlos dentro de la carpeta modules y asegurarnos que están habilitados (existe fichero *enable* o tienen un *default-enable*)

3.1.4 Parches

Tenemos que aplicar una serie de parches a esta versión de simpleSAMLphp, los descargamos del repositorio en una carpeta:

```
svn co https://confia.aupa.info/svn/confia/trunk/ssp/updates/
```

Creamos un directorio 'patches' en el directorio temporal y copiamos ahí los parches correspondientes a la version de simplesamlphp1.10:

```
mkdir /tmp/patches
cp updates/simplesamlphp1.10/patches/*.diff /tmp/patches
```

Aplicamos todos los parches sobre el simplesamlphp1.10

```
cd /var/www/simplesamlphp1.10
for patch in  /tmp/patches/*.diff; do patch -p0 < $patch; done
```

3.1.5 Sustitución

Una vez realizado todo lo anterior ya podemos sustituir la nueva instancia por la antigua:

```
mv /var/www/simplesamlphp/ /var/www/simplesamlphp_old/
mv /var/www/simplesamlphp1.10 /var/www/simplesamlphp/
```

Y comprobaríamos si todo funciona correctamente. Si no funciona siempre podremos volver a la anterior versión deshaciendo el anterior renombrado de directorios

Puede que algún filtro o algún módulo especifico que se implementara ajeno a CONFIA no funcione con la nueva versión de simplesamlphp por lo que habrá que hacer un testeo exhaustivo de que todo funciona correctamente.

3.2 Actualización del certificado en el SP

Desde la versión 1.7 se da soporte a que simplesamlphp trabaje a la vez con 2 certificados, para evitar que durante el periodo de propagación del nuevo certificado por la federación se deje de dar servicio.

El proceso sería el siguiente:

Primero editamos el archivo con los metadatos de nuestro SP, que será config/authsources.php en la declaración de la fuente de autenticación SP:SAML, añadiendole 2 nuevos atributos:

```
'new_privatekey' => 'new.pem',
'new_certificate' => 'new.crt',
```

donde *'new.pem'* y *'new_crt'* corresponden a los ficheros alojados en el directorio *cert* que poseen el nuevo certificado y la nueva key.

Una vez realizado esto, el SP pasará a exportar 2 metadatos válidos. Los metadatos la próxima vez que el gestor de metadatos se conecten a ellos serán leidos, almacenados y posteriormente distribuidos al resto de elementos de la federación.

En un plazo de 4 horas podremos estar seguro de que los elementos han sido distribuidos a todos los elementos del sistema por lo que podremos volver a configurar los datos de nuestro SP para que ahora únicamente utilice el nuevo certificado y la nueva key. Para ello volvemos a editar los metadatos de nuestro SP, alojados en `config/authsources.php`, y eliminaremos la entrada que añadimos antes:

```
'new_privatekey' => 'new.pem',
'new_certificate' => 'new.crt',
```

Y cambiaremos la entrada correspondiente con los antiguos certificado y clave:

```
'privatekey' => 'old.pem',
'certificate' => 'old.crt',
```

por los nuevos:

```
'privatekey' => 'new.pem',
'certificate' => 'new.crt',
```

Fuente original de la documentación del cambio de certificado