

Implementación de un Sistema de Contraseñas de uso único para la web

PROTECCIÓN DOBLE

Añadimos seguridad a un sitio web con un sistema de contraseñas de un sólo uso.

POR JAMES A. BARKLEY

La autenticación basada en dos factores es un sistema en el que se utilizan dos componentes en combinación para autenticar al usuario. Dos factores, al contrario que uno, proporcionará un nivel de seguridad superior en el proceso de autenticación. Los factores combinados podrían ser:

- Algo que conozca el usuario (contraseña o pin)
- Algo que posea el usuario (tarjeta inteligente, certificado PKI, RSA SecurID)
- Algo propio del usuario (huella dactilar, secuencia de ADN)

La primera opción es la más sencilla. Las contraseñas se usan con bastante frecuencia. La tercera opción se basa en la biométrica y no es buena para utilizarla en la web. "Algo que posea el usuario" es el segundo mejor factor para la autenticación. Aunque la mayoría de los sistemas de autenticación de dos factores basados en la web disponibles hoy en día implican el uso de algún token hardware, como el RSA SecurID. La distribución de estos elementos a los usuarios no es algo costoso. Una empresa podría permitirse el coste para unos 1000 usuarios, pero un buen blog podría encontrarse fácilmente con 30000 usuarios nuevos de la noche a la mañana. Requerirles que tengan que adquirir un token hardware por su propia cuenta es pedir demasiado para la mayoría de los mismos. Además, estos tokens tienen que sincronizarse con algún software especial en el servidor, el cual a menudo posee una licencia propietaria.

Una alternativa más barata y más escalable para la autenticación basada en dos factores en la web consiste en la utilización de un sistema de contraseñas de uso único (OTP). El número 45 de Linux Magazine, edición en castellano, hace una presentación de las OTPs

[1] centrada principalmente en la autenticación local; sin embargo, tareas como comprobar una cuenta bancaria desde una red de poca confianza piden a gritos el uso de algún sistema de autenticación basado en dos factores, siendo el sistema OTP a menudo una solución práctica. En este artículo se describe cómo añadir la seguridad de un sistema OTP a un sitio web.

OTP en la Web

El RFC 2289 [2] define un sistema OTP derivado de la tecnología Bellcore S/KEY (RFC 1760). Si se implementa correctamente proporciona un sistema de autenticación de dos factores a bajo coste para su uso en la web. Imagínese que un técnico con privilegios administrativos para un sitio web entra en una página administrativa que genera una lista del tamaño de una tarjeta de crédito de 30 pares de números/claves. A continuación se le hace llegar la lista al usuario. Esta lista de claves se convierte en algo que posee el usuario, el segundo factor, y como nunca se transmite electrónicamente, se añade un elemento más de confianza en su nivel de seguridad. Si el sitio web no tiene en cuenta las transmisiones electrónicas dentro de su dominio de confianza, el administrador podría enviar por fax o incluso por correo electrónico la lista al usuario. Desde un café en Amsterdam, por ejemplo, el usuario podría introducir un nombre de usuario y una contraseña convencionales. Si esta autenticación inicial se realiza con éxito, entonces el servidor solicita la introducción de la OTP correspondiente. Tras este proceso, la OTP se invalida, con lo que se garantiza que no podrá volver a usarse en un ataque en el futuro. En la siguiente autenticación el usuario tendrá que utilizar la siguiente OTP de la lista.

Así se fuerza a que se autentique por medio de dos mecanismos distintos. La autenticación basada en dos factores proporciona una alternativa mucho más segura para el proceso de autenticación en la web. Este escenario básico nos lleva a un sin fin de posibilidades. Por ejemplo, un usuario podría asociar un número de teléfono móvil a su cuenta; luego, cuando vaya a autenticarse, el sistema podría enviarle una OTP como mensaje de texto. O podría generar contraseñas OTP desde un programa ejecutándose en una PDA. Una ventaja añadida de este escenario es que la implementación puede proporcionarle a una OTP un componente temporal de modo que caduque transcurridos 60 segundos, como SecurID RSA, aunque requeriría que el usuario sincronizase la aplicación de la PDA con el servidor.

Herramientas OTP

Existen una gran variedad de librerías OTP para SSH, consola y autenticaciones de red, con multitud de librerías para herramientas más exóticas como SquirrelMail y PalmPilots, pero encontrar librerías de código abierto para autenticación OTP en un entorno web es difícil. La librería OTPAuth PHP [3] es un sistema OTP compatible con el estándar RFC 2289 que ha sido probada y se distribuye bajo licencia GPL. OTPAuth, que utiliza el algoritmo de hash SHA1, ha sido utilizada con éxito en un sitio web con varios cientos de usuarios durante dos años. Otra librería PHP, otp, se encuentra disponible en SourceForge [4]. Los desarrolladores de otp esperan tener una demo ejecutándose pronto.

Existen diversas herramientas Java que ayudan en la tarea de construir y validar OTPs [5], pero no conozco una librería web completa (por ejemplo, una que integre una

implementación completa dentro de una aplicación j2ee).

La librería Google AuthSub [6] permite la autenticación con aplicaciones Google por medio de tokens de seguridad. Aunque AuthSub no cumple estrictamente el RFC 2289, proporciona seguridad, permitiendo la utilización de tokens de un único uso para la autenticación con las aplicaciones Google. Sería interesante ver si Google va a continuar desarrollando esta solución o si migrará completamente a Oauth. Otros paquetes de software proporcionan soluciones OTP personalizadas exclusivamente para uso con ellos mismos, como el plugin para el CMS Joomla [7].

En este artículo se describe cómo instalar un sistema OTP con la librería de código abierto OTPauth. Las otras herramientas funcionan de manera similar. Si está interesado en explorar una de las alternativas, véase la documentación en el sitio web del proyecto correspondiente.

Un Vistazo a OTP

Imagínese un banco que desee poner en marcha una buena política de seguridad pero que no desee espantar a la mitad de sus

clientes insistiendo en un sistema de autenticación de dos factores universal. El banco necesita un sistema que soporte OTP sin que dañe el modelo de negocio y sin tener que forzar su uso a todos sus clientes.

La solución debe proporcionar un medio por el cual un usuario visite una página de preferencia y especifique que el programa requiera una autenticación basada en dos factores en el proceso de autenticación para acceder a la cuenta desde un ordenador diferente al que se está utilizando en ese momento. El usuario entonces genera una lista personal de OTPs del tamaño de una tarjeta de crédito de 30 pares número/clave desde la página de preferencias. La próxima vez que acceda a su cuenta desde una localización de poca confianza, digamos, desde la casa de un amigo, se le solicitará que proporcione una OTP junto con su nombre de usuario y su contraseña convencional.

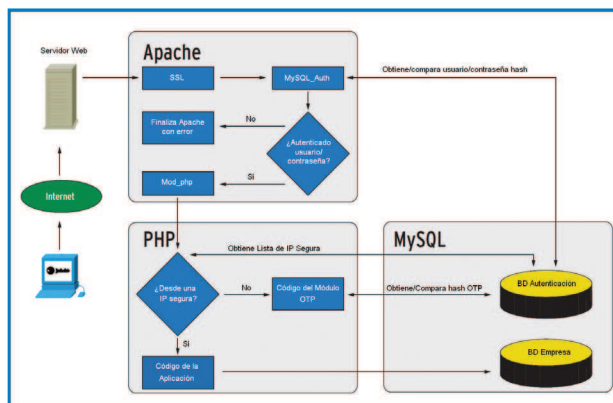


Figura 1: Escenario de autenticación web OTP.

El primer paso consiste en presentar una autenticación básica con el nombre de usuario y la contraseña. Existen buenas librerías y estándares para proporcionar una autenticación básica, ya sea por medio de ficheros *.htaccess* con Apache, validación por medio de una base de datos en la capa de aplicación o dejar que Apache valide al usuario con *mod_auth_mysql*. Como se desea tener controles de seguridad en múltiples capas, utilizaremos en este artículo la arquitectura mostrada en la Figura 1.

Listado 1: Ejemplo Base de Datos Autenticación

```
CREATE TABLE user (user_id int(11) NOT NULL AUTO_INCREMENT, user_name text NOT NULL, user_pw varchar(32) NOT NULL DEFAULT '', realname varchar(32) NOT NULL DEFAULT '', STATUS char(1) NOT NULL DEFAULT 'A', add_date int(11) NOT NULL DEFAULT '0', confirm_hash varchar(32) DEFAULT NULL, phone_number varchar(20) NOT NULL DEFAULT '', last_pw_change int(11) NOT NULL DEFAULT '0', otp_enabled tinyint(1) NOT NULL DEFAULT '0', PRIMARY KEY (user_id),) TYPE=MyISAM;

CREATE TABLE session (user_id int(11) NOT NULL DEFAULT '0', session_hash char(32) NOT NULL DEFAULT '', ip_addr char(15) NOT NULL DEFAULT '', otp_auth tinyint(1) NOT NULL DEFAULT '0', time int(11) NOT NULL DEFAULT '0', locked tinyint(1) NOT NULL DEFAULT '0', PRIMARY KEY (session_hash),) TYPE=MyISAM;

CREATE TABLE otp (user_id int(11) NOT NULL DEFAULT '0', sequence int(11) NOT NULL DEFAULT '0', otp char(60) NOT NULL DEFAULT '', PRIMARY KEY (session_hash),) TYPE=MyISAM;
```

Listado 2: Añadiendo mod_auth_mysql a http.conf

```
01 #carga librería objeto compartido mod_auth_mysql
02 LoadModule mysql_auth_module modules/mod_auth_mysql.so
03
04 #le dice a mod_auth_mysql como conectar a la base de datos de autenticación, los parámetros sone:<br>#Auth_MySQL_Info hostname user password
05 Auth_MySQL_Info localhost auth_db_user myP@ssw0rd<br>
06
07 # redirige inicios de sesión fallidos a página de rechazo
08 ErrorDocument 401 /chapter14/rejection.html
09
10 #define tablas y columnas MySQL para autenticación
11 AuthName "My Web Site"
12 AuthType Basic
13 Auth_MySQL_DB auth_db
14 Auth_MySQL_Encryption_Types MySQL
15 Auth_MySQL_Password_Table user
16 Auth_MySQL_Username_Field user_name
17 Auth_MySQL_Password_Field user_pw
18
19 require valid-user
```

La base de datos para la autenticación se almacena separada de la base de datos con la información del modelo de negocio, conteniendo el nombre de usuario, la contraseña y la información OTP. El Listado 1 muestra las sentencias *CREATE* para MySQL que contienen toda la información necesaria para la base de datos para la autenticación, con *mod_auth_mysql* para comprobar el nombre del usuario y la contraseña. Primero hay que descargar *mod_auth_mysql* u obtenerlo del gestor de paquetes de la distribución [8]. Una vez que se tenga instalado, hay que configurarlo añadiéndole las líneas del Listado 2 al fichero *httpd.conf*. Recuerde que hay que personalizar los parámetros para que coincidan con los de su sitio web.

Ahora ya se puede añadir el código a la capa de la aplicación para comprobar la autenticación OTP. El usuario no llegará nunca a la aplicación sin haber introducido correctamente su nombre de usuario y su contraseña, pero una vez hecho esto, habrá que asegurarse de que se encuentre disponible la autenticación basada en dos factores. Primero, la aplicación debe determinar si el usuario ha activado la autenticación OTP para su cuenta. Si este es el caso, tiene que comparar la dirección IP actual y/o el nombre del equipo para ver si se encuentra en la lista de los sitios de confianza de la cuenta del usuario. Si el usuario no tiene activada la opción OTP o está accediendo desde un sitio de confianza, la aplicación permitirá el acceso a las páginas solicitadas. El código de ejemplo que aparece



Figura 2: Entrando con una contraseña de uso único.

en el Listado 3 podría incluirse (*include*) al comienzo de la página de su aplicación (en este caso, se ejecutaría al comienzo) o bien colocarlo dentro de una función.

Listado 3: Lógica PHP OTP

```

01 <?php
02
03 ...
04 ...
05
06 //obtener id usuario de global
  establecido Apache
07 //o mecanismo similar
08 $uid = user_getid();
09
10 //establece sesión de usuario
11 $session =
  user_getsession($uid);
  //intenta obtener sesión de
  BdeD
12 if (!$session) {
13 $session =
  user_create_session($uid);
  //crea entrada en tabla de
  sesiones
14 }
15
16 //comprueba si el usuario ya
  está autenticado
17 //Esto evita condición de
  carrera especificada por RFC
  2289
18 while ($session['locked']) {
19 /* sigue intentándolo hasta
  que el bloqueo se libera o
  ocurre un timeout */
20 $session =
  user_getsession($uid);
21 if
  (spinlock_timeout_reached()) {
22 header("Location:
  http://www.example.com/retry.p
  hp");
23 exit;
24 }
25 }
26
27 //bloquear cuenta mientras se
  autentica
28 set_session_lock($uid);
  //establece flag de bloqueo en
  tabla de sesiones
29
30 //comprueba si otp auth ha
  sido habilitado cuenta
31 //user_getotppauth() ejecuta
  consulta en base de datos
  performs y devuelve
32 //flag otp_enabled flag de la
  tabla user
33 $otp_auth_enabled =
  user_getotppauth($uid);
34
35 if ($otp_auth_enabled) {
36 if ($session['otp_auth']) {
37 /* éxito, usuario ya se ha
  autenticado con otp */
38 } else {
39 /* el usuario ha iniciado
  sesión, pero no ha sido
  autenticado con otp */
40
41 //untrusted_host() compara IP
  de la sesión actual
42 //con lista confianza de el
  usuario especificado
43 if (trusted_host($uid)) {
44 /* el usuario viene de una
  dirección que no requiere
  autenticación OTP */
45 } else {
46 /* el usuario debe
  autenticarse con otp */
47 header("Location:
  http://www.example.com/otp_aut
  h.php");
48 exit;
49 }
50 }
51 }
52
53 //en todos los casos, menos
  los de otp requerido
54 //el usuario acaba aquí
55 //liberar bloqueo y proceder a
  código de página
56 unset_session_lock($uid);
57
58 ...
59 ...
60
61 ?>

```

Listado 4: Ejemplo de Página de Autenticación OTP

```

01 <?php
02 /* LICENSED UNDER THE GPL */
03 # si han hecho clic en el
   botón de inicio de sesión
04 if ($login) {
05 $success =
   valid_otp($form_challenge_respon
   se, 06$user);
06 if ($success) {
07 /* actualizar sesión/estado de
   autenticación y redirigir a
   los recursos del sistema y
   salir */
08 header("Location:
   http://www.example.com/". $page
   );
09 exit();
10 }
11 }
12
13 $sequence = get_otp_seq($uid);
14 if ($sequence == -1) {
15 /* mostrar mensaje de error y
   salir; */
16 }
17
18 print "
19 <p>
20 <FORM ACTION=\""$PHP_SELF\"
   METHOD=\"POST\">
21 <p>
22 <INPUT TYPE=\"TEXT\"
   NAME=\"user\" VALUE=\""$user\">
23 <p>
24 Enter One-Time Password for
   Challenge number
25 <B>$sequence</B>:
26 <br><INPUT TYPE=\"TEXT\"
   NAME=\"form_challenge_response
   \"
   VALUE=\""$form_challenge_respon
   se\" SIZE=\"31\">
27 <p>
28 <INPUT TYPE=\"SUBMIT\"
   NAME=\"LOGIN\"
   VALUE=\"Login\">
29 </FORM>
30 <P>
31 ";
32 ?>

```

Cuando el usuario se haya redirigido al sitio *otp_auth*, la librería OTP se encargará del proceso de autenticación basado en la OTP. El Listado 4 muestra una página básica que presenta al usuario la petición de una OTP y su correspondiente validación. Esta página es intencionadamente escueta, ya

que mi aplicación aún no está convencida de que esta persona sea la auténtica. Sin proporcionar ninguna de las librerías normales o JavaScript, ni incluso el aspecto habitual del sitio web, hay que limitar la información ante un posible ataque. El uso de una buena librería OTP simplifica la

lógica de esta aplicación a una cantidad de código trivial con funciones como *valid_otp()* y *get_otp_seq()*. El código del Listado 4 muestra una pantalla similar a la que aparece en la Figura 2.

Por último, no hay que olvidar suministrar a los usuarios herramientas que habiliten el

Estimación de Riesgos y Vectores de Ataque

El mayor riesgo en un sistema de seguridad sería implementar una arquitectura con errores de programación. El riesgo técnico consiste en ser "hackeado"; el riesgo de negocio consiste en ser demandado. Para protegernos, se pueden escribir extensos casos de pruebas y hacer que un tercero revise el software, o se puede elegir alguna implementación existente que esté probada o que se acepte que sea correcta.

A nivel arquitectónico, la programación segura requiere diversas consideraciones, muchas de ellas únicas para la infraestructura existente del sistema. Las dos más importantes son: 1) separar la base de datos para la autenticación de la base(s) de datos de la empresa y 2) separar la autenticación usuario/contraseña de la autenticación OTP tanto como sea posible. Es deseable que cada elemento de la autenticación sea claro y lo más individual posible. Si se opera con el nombre del usuario y la contraseña de autenticación en la misma librería de código como la autenticación OTP, se corre el riesgo de contaminar el proceso completo de

autenticación con errores de código cruzados.

Otra vulnerabilidad técnica en cualquier sistema OTP es el ataque por medio de la técnica conocida como "man-in-the-middle" (MITM). Si un hacker puede enmascararse como un sitio web de cara al usuario mientras que se enmascara como un usuario de cara al sitio web, podrá suplantar al usuario. Incluso con SSL, el ataque MITM simplemente tiene que realizar conexiones SSL independientes y decidir qué es lo que tiene que enviarle al usuario y qué lo que tiene que enviarle al sitio web. La monitorización de la red puede impedir este tipo de ataques: Como el atacante continúa operando, surgen patrones de errores. El RFC 2289 permite variantes de algoritmos de hashing en la generación de la OTP y en las funciones de autorización. Hay que asegurarse de que se utiliza un algoritmo hash fuerte, ya que un intruso que gane acceso a la base de datos de autenticación y utilice una función de hash poco segura como MD5, podría generar infinitas listas de elementos. Al menos, se debe utilizar

SHA1, aunque no sea de alta seguridad. SHA256 es la mejor solución, pero no está disponible en muchos lenguajes, así que habrá que confiar en una librería externa o bien programarlo directamente. El RFC también menciona una excepción interesante (véase el cuadro titulado "Excepción") para asegurarse de que la librería cumple completamente el estándar.

Otro asunto es la ingeniería social. Tristemente, la mayoría de las soluciones a este problema no son técnicas. La mayor parte de los sistemas son susceptibles, incluyendo los sistemas de autenticación comerciales basados en tokens hardware, así que ya existen modelos para establecer identidades. La formación y las políticas de servicios podrían ser de ayuda. En la práctica, la mayoría de los sitios con bastantes requerimientos de seguridad para garantizar el uso de OTP ya poseen planes de formación anuales sobre técnicas de seguridad. Esta formación también debería prohibir el almacenamiento de listas de OTP con nombres de usuarios y contraseñas regulares.

Listado 5: Hoja de Cálculo para Generar Listas OTP

```

01 $otp_list = generator($uid);      08
02 /*                                09
03 simular ser un fichero excel y    10 print "<TABLE BORDER=1>";
dejar que excel o oo.calc           11 print "<TH>Sequence
poner el html en el formato de     number</TH><TH>Password</TH>";
hoja de cálculo correcto
04 */
05 header("Content-Type:            12 while (list($key, $val) =
application/vnd.ms-excel");        each($otp_list)) {
06 header("Expires: 0");            13 print
07 header("Cache-Control:           <TR><TD>$key</TD><TD>$val</TD
must-revalidate, post-check=0,    ></TR>";
pre-check=0");                    14 }
15 print "</TABLE>";
    
```

uso de OTPs en sus cuentas, generar una lista de OTPs y gestionar las listas de confianza. El Listado 5 muestra una página simple que genera OTPs en formato de hoja de cálculo, pero hay que asegurarse de que cuando un usuario active el uso de OTPs en una cuenta, no se haya desconectado antes de generar primero su lista OTP.

Hay que estar preparado para tener un mecanismo para reiniciar las listas OTP. Esto podría llevarse a cabo por otros canales, como teléfono, correo electrónico o soporte irc; o por medio de una página automática, pero ya sea de cualquiera de estas formas, el usuario tendrá que probar su identidad con algo más que una simple pregunta de seguridad. Tampoco hay que olvidarse de sus propias medidas de seguri-

A Nivel de Bits

La implementación de la especificación RFC 2289 utilizada en OTPauth fue programada en PHP4, y también funciona en PHP5. Para implementar la especificación de forma correcta es necesario realizar diversas operaciones a nivel de bits. Sin embargo, en el momento de la implementación (y que yo sepa no ha cambiado), las operaciones específicas a nivel de bits no funcionan en PHP4. Así que, operaciones como desplazamientos de bits para enteros de 32 bits sin signo no funcionan. PHP4 suministra el operador, pero simplemente no funciona y no produce ningún error. Por ello, OTPauth proporciona una librería de operaciones matemáticas cuya misión consiste en solventar estos problemas con estas clases de "características" no documentadas del lenguaje.

dad: ningún código de los mostrados anteriormente valida los datos de entrada, por poner un ejemplo.

¿Y un Token?

La especificación RFC 2289 para contraseñas de un sólo uso puede ofrecer autenticación basada en dos factores; sin embargo, nunca será tan segura como una alternativa basada en un token. Por un lado, la mayoría de las

Condición de Carrera

Existe una condición de carrera para los sistemas OTP. Un atacante que esté escuchando con un programa que registre las pulsaciones del teclado mientras un usuario esté autenticándose podría ser capaz de escuchar lo suficiente de la contraseña OTP como para lanzar un ataque por fuerza bruta antes de que el usuario termine de teclearla, permitiendo al atacante autenticarse como el usuario antes de que éste finalice su proceso de autenticación. El RFC 2289 en realidad prevé esta excepción y requiere que se proteja para asegurarse de que la implementación se cumpla de forma completa. La defensa resaltada en la Sección 9.0 del RFC sirve para impedir múltiples sesiones simultáneas. Dicho de otro modo, una vez que un usuario inicie una secuencia de autenticación, cualquier otro intento de autenticación por parte del mismo usuario debe ser bloqueado hasta que el proceso de autenticación de dicho usuario haya finalizado. Teniendo en cuenta esto, se podría producir un ataque por denegación de servicio, así que es necesario algún tipo de temporizador en el proceso de autenticación.

soluciones basadas en un token requieren que se concatene un PIN privado al OTP para crear el segundo factor, aumentando de forma considerable la seguridad. También, las soluciones basadas en un token hardware están diseñadas para evitar su falsificación en el caso de que alguien intentase utilizar ingeniería inversa para generar el algoritmo. Por último, las herramientas basadas en tokens poseen controles de tiempo y cambian cada minuto aproximadamente, con lo que es muy difícil para un atacante obtener una OTP que un usuario no haya ya utilizado. Con soluciones que requieran una lista OTP, un atacante que se haga con una copia de la lista (o que se encuentre una lista perdida en la parada del autobús) poseerá acceso a las siguientes OTPs.

El sistema OTP definido por el RFC 2289 ofrece una solución abierta y escalable para la autenticación web. Incluso es posible integrar un sistema OTP en un teléfono móvil. Las OTP válidas para la web tienen sus propios riesgos y vectores de ataque, por lo que probablemente un sistema OTP para la web nunca llegue a ser tan seguro como una solución basada en hardware como RSA SecurID. A pesar de esto, la combinación de OTP con un esquema de autenticación normal es un candidato excelente para la implementación de un sistema de autenticación basado en dos factores.

RECURSOS

- [1] "Usar y Tirar" por Udo Seidel, Linux Magazine Edición en Castellano, Número 45
- [2] RFC 2289: <http://www.apps.ietf.org/rfc/rfc2289.html>
- [3] OTPauth: <http://code.google.com/p/otpauth/>
- [4] otp: <http://sourceforge.net/projects/otp/>
- [5] OTPs en Java: <http://www.java2s.com/Code/Java/Security/OTPonetimepasswordcalculation.htm>
- [6] Librería AuthSub de Google: <http://code.google.com/apis/accounts/docs/AuthSub.html>
- [7] Plugin de Joomla para OTP: <http://code.google.com/p/joomla-otp-auth/>
- [8] Módulo de Apache mod_auth_mysql: <http://modauthmysql.sourceforge.net/>
- [9] Ataque de tipo phishing a OTPs: http://www.theregister.co.uk/2005/10/12/outlaw_phishing/