UNIVERSIDAD COMPLUTENSE MADRID

Time-based One-time Passwords (TOTP)

Gestión de la Información en la Web
Enrique Martín - emartinm@ucm.es
Grados de la Fac. Informática

Debilidad de las contraseñas

- Autenticar usuarios únicamente por su contraseña no es muy seguro:
  - Los usuarios **repiten** mucho sus claves entre varias aplicaciones web.
  - Las contraseñas suelen **débiles**: cortas y fáciles de adivinar. Ej: `password`, `1234`, `asdf1234`.
  - Las contraseñas no se **cambian** con **frecuencia**.
- Cualquier atacante que conozca la clave (no importa cómo: MITM, ingeniería social, ataque a otra web) podrá acceder a la cuenta sin límites.

Segundo factor de autenticación

- Para mitigar esta situación se utiliza una **autenticación** basada en **varios factores**.
- El caso más común son **2 factores**:
  - **Contraseña**: escogida por el usuario *(algo que sabe)*
  - ***One-Time password:*** generada o transmitida a un dispositivo del usuario *(algo que tiene)*. Ej: SMS al móvil, HOTP o **TOTP en el móvil**, dispositivos dedicados.

Segundo factor de autenticación

- Si la contraseña principal es comprometida pero el atacante no tiene acceso al dispositivo no podrá acceder a la cuenta → carece de la OTP actual.
- *Idealmente*, los sistemas deben avisar al usuario cada vez que un acceso sea denegado por introducir una OTP incorrecta → **posible acceso ilegítimo debido a una contraseña principal comprometida**.

Tipos de OTP

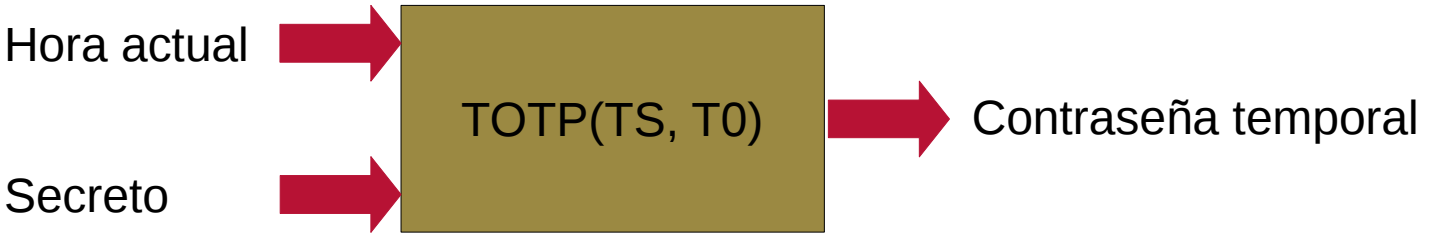
- Las contraseñas de un solo uso se pueden generar de 2 formas:
  - Basadas en **secuencia**: generan la OTP en base a un **secreto** y un **contador** que se incrementa con cada contraseña generada. Ej: HMAC-based OTP.
  - Basadas en **tiempo**: generan la OTP en base a un **secreto** y la hora actual. Tienen una corta duración (usualmente 30 segundos). Ej: Time-based OTP.

Segundo factor de autenticación

- Incluso aunque un atacante descubra la contraseña y la última OTP utilizada, el sistema será seguro:
  - Secuencia: La próxima OTP a usar siempre será diferente de la última utilizada, y no es posible calcularla a partir de esta.
  - Tiempo: La siguiente OTP a usar será igual a la última se estamos en el mismo periodo de tiempo → ese será el tiempo máximo de vulnerabilidad (usualmente 30 s)

TOTP

- Para calcular el TOTP actual hacen falta 2 valores: la **hora actual** y el **secreto**:



- También existen 2 parámetros que es necesario considerar:
  - (**TS**) Periodo: 30 segundos
  - (**T0**) Fecha inicial: 1 de enero 1970 (tiempo UNIX)

TOTP

- El cálculo de la clave temporal se basa en funciones hash criptográficas, y es perfectamente conocido: https://en.wikipedia.org/wiki/Time-based_One-time_Password_Algorithm
- La **robustez** del sistema proviene de que es **impracticable** obtener el secreto a partir de una o varias claves generadas.

Autenticación con TOTP (I)

- Para incorporar TOTP como segundo factor de autenticación en una aplicación web es necesario un primer paso de **configuración**:
  1) La aplicación genera un **secreto** y lo almacena en el perfil del usuario.
  2) La aplicación muestra el secreto al usuario y este lo registra en su aplicación TOTP (Google Authenticator, Latch, etc.). Este paso se suele agilizar mediante códigos **QR**.

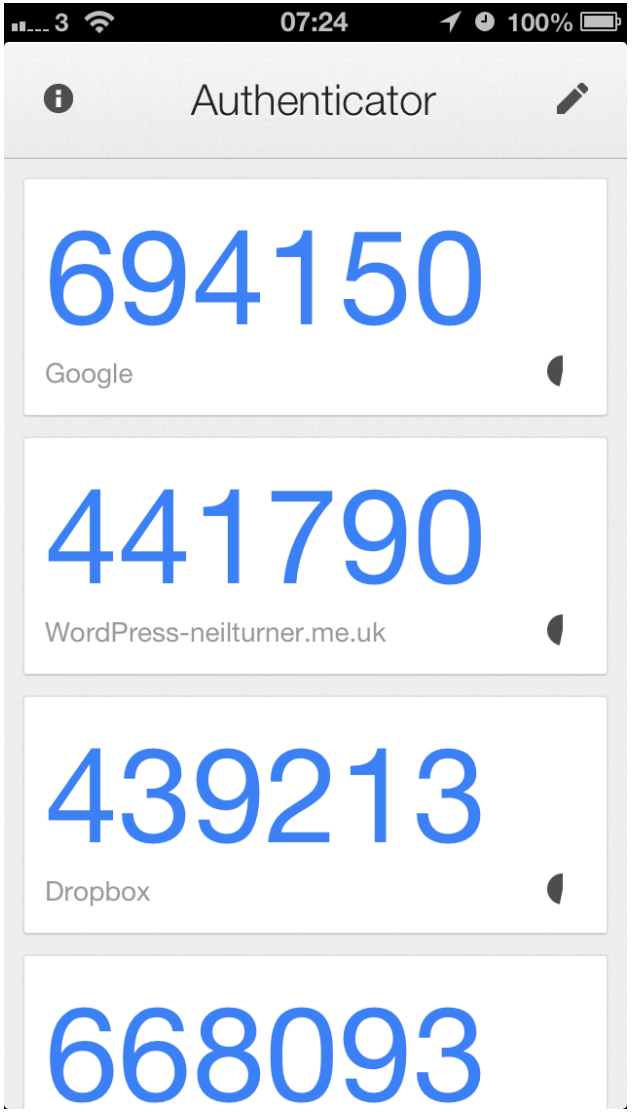
Registro en TOTP

Fuente: https://www.netiq.com/documentation/netiqaccessmanager4/identityserverhelp/data/b17z8cfo.html

Autenticación con TOTP (II)

- Cuando un usuario quiere acceder al sistema, la aplicación debe realizar más pasos:
  1) Solicita usuario y contraseña principal.
  2) Si tiene éxito, pedir la TOTP.
  3) A partir del secreto del usuario y la hora, generar un TOTP interno → comparar con el TOTP introducido por el usuario.
  4) *(Buenas prácticas)* En caso de fallo de TOTP, notificar al correo del usuario.

TOTP: vista del usuario



Fuente: https://www.cyberriskopportunities.com/how-to-use-google-authenticator/

Aspectos de TOTP

- A diferencia de la contraseña, el secreto para TOTP solo se comparte una vez, al configurar la cuenta.
- Sin embargo, cualquiera que conozca el secreto podrá generar todos los TOTPs.
- Tanto la aplicación web como el dispositivo del usuario que genera TOTPs deben estar sincronizados → *Network Time Protocol*
  - Como el periodo suelen ser 30 segundos, pequeñas diferencias no afectan a la usabilidad.

TOTP en Python

- Es muy sencillo usar TOTP en Python mediante la biblioteca **onetimepass**:
  - En los laboratorios ya está instalada.
  - En los demás equipos se instala fácilmente con **pip.**
  - https://github.com/tadeck/onetimepass
- **Generar** un TOTP con la hora actual y `secret`:
  `get_totp(secret)`
- **Verificar** si `t` es un TOTP valido para la hora actual y `secret`:
  `valid_totp(token=t, secret=secret)`