One-time Passwords (OTP)

Dynamic Password Generator

Oregon State University
College of Engineering

- System periodically creates and displays new password
  - E.g., RSA SecurID, Google Authenticator App,

RSA SecurID

- User enters current password


Google Authenticator


FreeOTP

One-Time Passwords

Oregon State University
College of Engineering

- Password that can be used exactly *once*
  - After use, it is immediately invalidated
- Challenge-response mechanism
  - Challenge is one of a number of authentications; response is password for that particular number
- Problems
  - Synchronization of user (prover), system/server (verifier)
  - Generation of good random passwords
  - Password distribution problem

S/Key

Oregon State University
College of Engineering

- One-time password scheme based on idea of Lamport

- *h* one-way hash function (SHA-256, for example)

- User chooses initial seed *k*
- Server calculates:

$$h(k) = k_1,\ h(k_1) = k_2,\ \ldots,\ h(k_n) = k_{n+1}$$

  - Needs only save k_{n+1}

- Passwords are reverse order:

$$p_1 = k_n,\ p_2 = k_{n-1},\ \ldots,\ p_{n-1} = k_2,\ p_n = k_1$$

S/Key

Central Ideas:

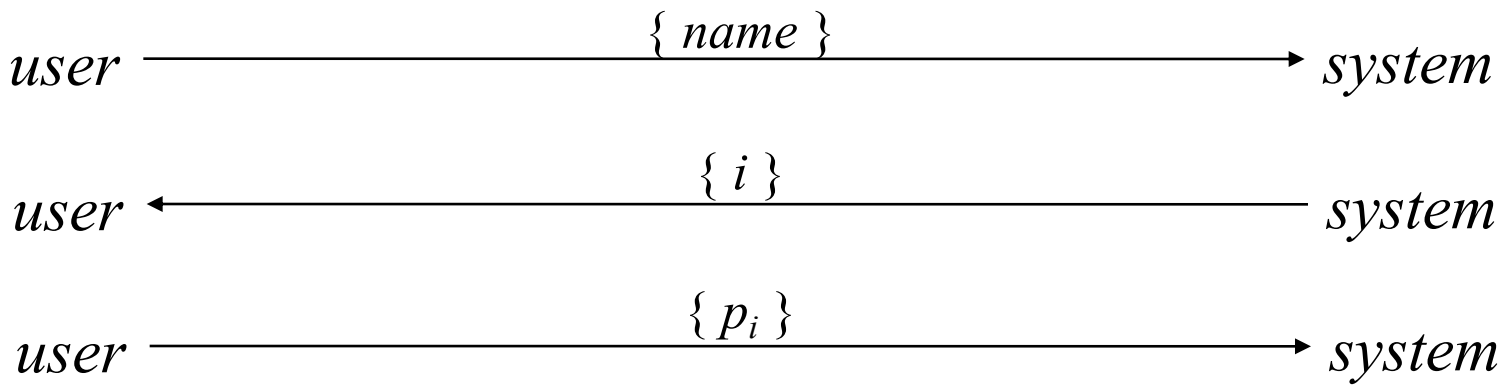
- Given last pwd p, observer cannot predict p’ s.t. h(p’) = p,
  - i.e., cannot predict next password.

- Server remembers last pwd p, and when p’ is offered, validates h(p’) = p

S/Key Protocol

Oregon State University
College of Engineering

System stores maximum number of authentications n, number of next authentication i, last correctly supplied password p_{i-1}.



System computes $h(p_i) = h(k_{n-i+1}) = k_{n-i+2} = p_{i-1}$. If match with what is stored, system replaces p_{i-1} with p_i and increments i.

HOTP

- HMAC-based One-Time Password (HOTP) algorithm
  - IETF RFC 4226, December 2005
- Server and user pre-establish a shared secret K, and a beginning counter value C
- HOPT One-time password value is computed as follows
  - HOTP(K,C) = Truncate(HMAC-SHA-1(K,C))
  - HOTP Password = HOTP(K,C) mod 10^d where d is password length (typically 6 - 8 digits)
  - Truncate() extracts 31-bits but NOT the last 31 bits. 31 bits starting at index *i+1* where *i* is last four digits of the MAC value.
- Counter is updated after every successful login

TOTP

Oregon State University
College of Engineering

- Time-based One-Time Password (TOTP) algorithm
  - Extension of HOTP
  - IETF RFC 6238, May 2011

- Server and user pre-establish a shared secret K

- Counter value is obtained using current Unix time
  - C_T = floor([(T – T_0)/Time-step])
  - T is current Unix Time, T_0 defaults to 0. Time-step defaults to 30 seconds

- To account for network delays and synchronization issues, validator will also check with C_T+1 and C_T-1

Summary

Oregon State University
College of Engineering

- Dynamic or One-Time Passwords provide better security against password loss etc.
- S/KEY's use is reducing as a main form of authentication
- OTPs (e.g., HOTP, TOTP) are gaining use as a 2nd factor of authentication