Key-Derivation Scheme

Rifaat Shekh-Yusef
IETF 91, SIPCore WG
Honolulu, Hawaii, USA
November 13, 2014

Digest Scheme Issues

- Weak protection of passwords at rest.
- Low entropy passwords.
- Password-hash sent on the wire.
- Dictionary attacks.
- Downgrade attacks.
- Replay attacks.
- And more

Alternatives

- PAKE-based approach
  - e.g JPAKE
- Key-Derivation-based approach
  - RFC5802
    - Salted Challenge Response Authentication Mechanism (SCRAM) SASL and GSS-API Mechanisms

PBKDF2

- **Password-Based Key Derivation Function (PBKDF):**
  - A function used to derive cryptographic keys from a password for the protection of stored data.

- **Parameters:**
  - Password
  - Salt
  - Iteration Count
  - Key Length
  - KDF
    - e.g HMAC-SHA256

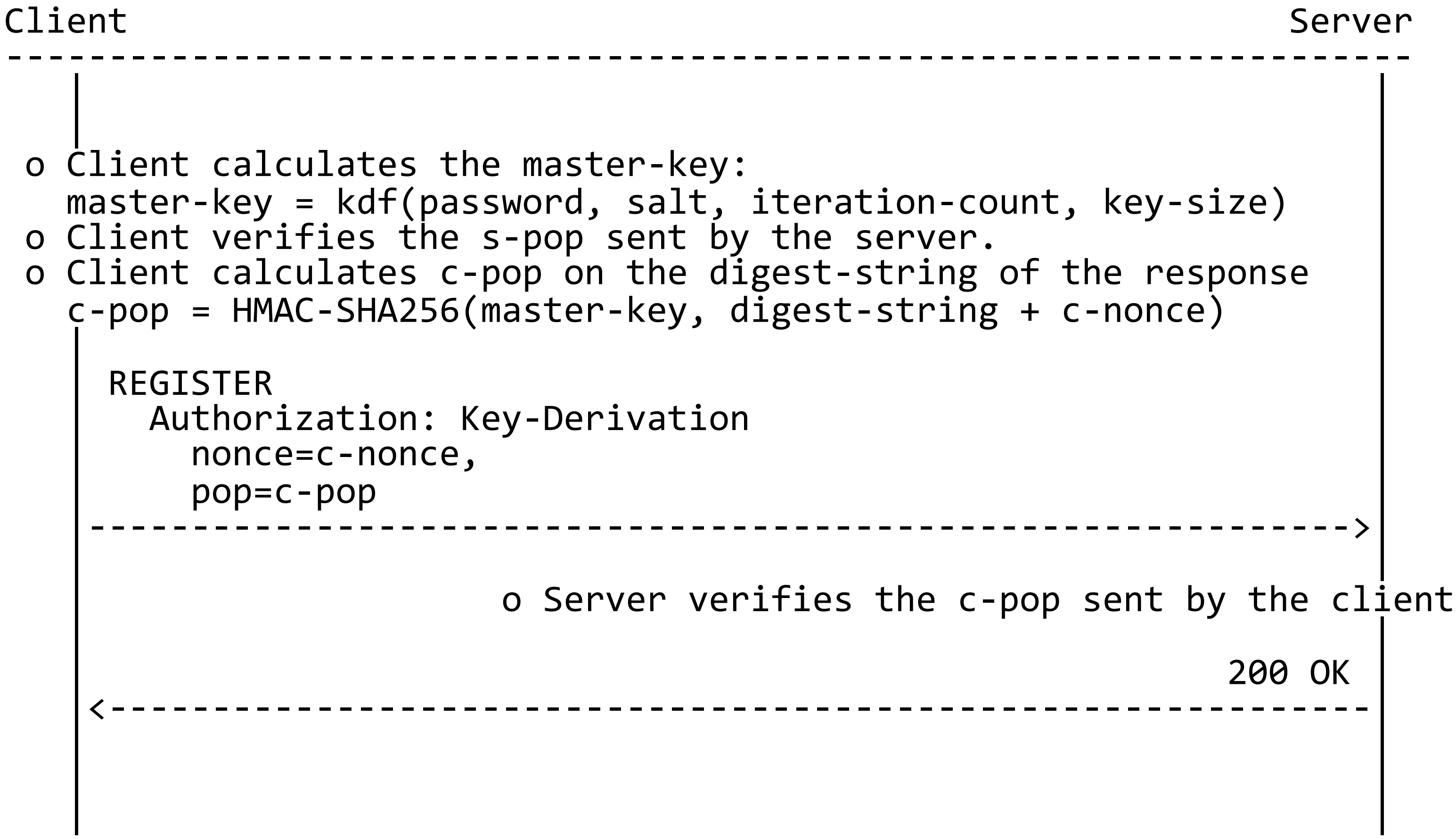
Create User Account

- When an account is created, the server uses the user's **password**, a **KDF**, a **salt**, a **key length**, and an **iteration count** to create a **master-key**.
- The server then stores the following information in the database:
  - username
  - iteration count
  - salt
  - master-key

Challenge

```
Client                                                        Server
--------------------------------------------------------------------
   |                                                              |
   | REGISTER                                                     |
   |   username@domain.com                                        |
   |------------------------------------------------------------->|
   |                                                              |
   |    o Server calculate s-pop on the digest-string of the challenge
   |      s-pop = HMAC-SHA256(master-key, digest-string + s-nonce)|
   |                                                              |
   |                         401 Unauthorized                     |
   |                             WWW-Authenticate: Key-Derivation |
   |                                kdf="HMAC-SHA256",            |
   |                                salt=<some-salt>,             |
   |                                key-size="256",               |
   |                                iteration-count=10000,        |
   |                                nonce=s-nonce,                |
   |                                pop=s-pop                     |
   |<-------------------------------------------------------------|
   |                                                              |
```

Response



```
Client                                                         Server
---------------------------------------------------------------------

 o Client calculates the master-key:
   master-key = kdf(password, salt, iteration-count, key-size)
 o Client verifies the s-pop sent by the server.
 o Client calculates c-pop on the digest-string of the response
   c-pop = HMAC-SHA256(master-key, digest-string + c-nonce)

     REGISTER
       Authorization: Key-Derivation
         nonce=c-nonce,
         pop=c-pop
    --------------------------------------------------------------->

                        o Server verifies the c-pop sent by the client

                                                            200 OK
    <---------------------------------------------------------------
```

Subsequent Request

```
Client                                                        Server
--------------------------------------------------------------------
   |                                                              |
   |                                                              |
 o Client calculates c-pop on the digest-string of the request    |
   c-pop = HMAC-SHA256(master-key, digest-string + c-nonce)       |
   |                                                              |
   | INVTIE                                                       |
   |   Authorization: Key-Derivation                              |
   |     nonce=c-nonce,                                           |
   |     pop=c-pop                                                |
   |------------------------------------------------------------->|
   |                                                              |
   |                    o Server verifies the c-pop sent by the client
   |                                                              |
   |                                                       200 OK |
   |<-------------------------------------------------------------|
   |                                                              |
   |                                                              |
   |                                                              |
```

Benefits

- Better storage protection
- Mutual authentication
- Better dictionary attack protection
- Better replay attack protection
- Less traffic

References

- **PBKDF2**
  - "NIST Special Publication 800-132 - Recommendations for Password-Based Key Derivations", December 2010. http://csrc.nist.gov/publications/nistpubs/800-132/nist-sp800-132.pdf

- **HTTP Digest**
  - Shekh-Yusef, R., Ahrens, D., Bremer, S., "HTTP Digest Access Authentication", draft-ietf-httpauth-digest-08, (Work In Progress), August 2014.

- **RFC5802**
  - Newman, C., Menon-Sen, A., Melnikov, A., and N. Williams, "Salted Challenge Response Authentication Mechanism (SCRAM) SASL and GSS-API Mechanisms", RFC5802, July 2010.

- **JPAKE**
  - Hao, F., "J-PAKE: Password Authenticated Key Exchange by Juggling", draft-hao-jpake-01, (Work In Progress), December 2013.