HTTP SRP Authentication

draft-yusef-httpauth-srp-scheme-02
Overview

• **Secure Remote Password (SRP)** is an **Augmented PAKE** protocol that is used to authenticate users and exchange keys over an untrusted network, based on a shared password, without requiring a **Public Key Infrastructure (PKI)** or any **trusted third party**.
Proposal Highlight

• A generic authentication framework based on the HTTP Authentication Framework [RFC7235] and SRP.
• Can be used for HTTP, SIP, as well as other protocols.
  – Not expected to be used for generic Web traffic.
Server and Account Setup

• **Server Setup**
  – Select a large-prime and a generator.

• **Account Setup**
  – Select a hash function and a user salt
  – Use a realm and the user password to create a password-verifier as follows:
    • derived-private-key = H(username:realm:password:salt)
    • password-verifier = generator ^ derived-private-key
  – Discard derived-private-key.

• **Database**
  – Store the following: Username, Password-verifier, Hash-algorithm, and Salt.
Realm Discovery

• The initial request that starts the SRP authentication process must include the username parameter.
  – To allow the user to select the proper username, the **Realm** is needed.
  – The discovery step is an optional step that allows the client to discover the **Realm**.

<table>
<thead>
<tr>
<th>Client</th>
<th>Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Authorization: SRP</td>
<td>WWW-Authenticate: SRP realm=&quot;realm&quot;</td>
</tr>
<tr>
<td></td>
<td>&lt;</td>
</tr>
</tbody>
</table>
Authentication

Client

Authorization: SRP
username="username"

Server

WWW-Authenticate: SRP
large-prime="large-prime"
generator="generator"
hash-algorithm="hash-algorithm"
salt="salt",
server-public-key="server-public-key"

Authorization: SRP
server-public-key="server-public-key"
client-public-key="client-public-key"
client-pop="client-pop"

WWW-Authenticate: SRP
server-pop="server-pop"
Benefits

• Resists **passive** and **active** dictionary attacks.
• Offers **perfect forward secrecy**.
• User **passwords** or **hashes** are **not** stored in the DB.
  – Only **password verifiers** are stored, which cannot be used directly to compromise the security of the system in the case of DB compromise.
• **Royalty-free** worldwide for commercial and non-commercial use.
• A variety of SRP implementations are available
• **IETF RFCs**
  – RFC2945 (SRP), RFC2944 (Telnet SRP), RFC5054 (SRP with TLS).
Questions?

- Can the WG adopt this work?