HTTP SRP AuthN Scheme Proposal

draft-yusef-httpauth-srp-scheme-00

Rifaat Shekh-Yusef, Yaron Sheffer

IETF 93, HTTPAuth WG

Prague, Czech Republic

July 20, 2015

Background

- **SRP** is royalty-free worldwide for commercial and non-commercial use.
  - http://srp.stanford.edu/license.txt
- **IETF RFCs**
  - RFC2945 - The SRP Authentication and Key Exchange System
  - RFC2944 - Telnet Authentication: SRP
  - RFC5054 - Using the Secure Remote Password (SRP) Protocol for TLS Authentication
- A variety of SRP implementations are available
  - http://srp.stanford.edu/links.html

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.
- Resistant to phishing
- Resistant to dictionary attacks on the server

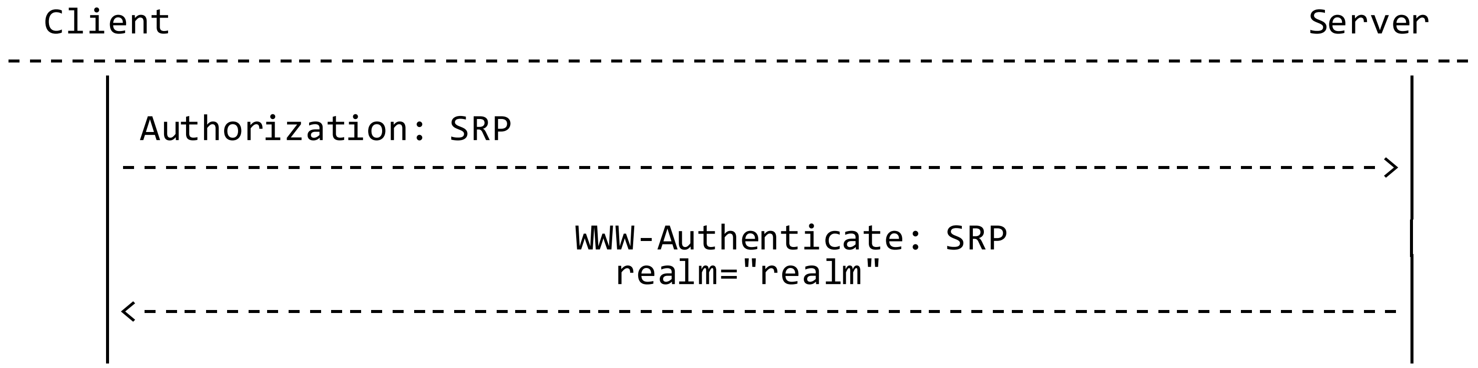
Server Setup

- The server must choose a **large-prime** and a **generator**.
- When a user account is created, the server selects a **hash function** and a **user salt**, and uses 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

- The server then stores the following information in the database:
  - Username
  - Password-verifier
  - Hash-algorithm
  - 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** to allow the user to select the proper **username**.



Authentication

```
 Client                                                        Server
-----------------------------------------------------------------------
     
      Authorization: SRP
        username="username"
      ------------------------------------------------------------>

                            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, allowing even **weak** passwords to be used safely.
- Offers **perfect forward secrecy**, which protects past sessions and passwords against future compromises.
- User **passwords** or **hashes** are **not** stored in the DB. Instead, only **password verifiers** are stored, which in the case of DB compromise the password verifiers cannot be used directly to compromise the security and gain immediate access to the host.

Questions?