Authentication Protocol Description

The Key Exchange [Algorithm 1] allows the two parties to agree on a shared secret key s from a common *password*. According to the specification, the algorithm is symmetrical for the two interacting parties with the only difference in one being the initiator of the protocol (Client).

Algorithm 1 Key exchange

Given public p secure prime

Client(password)		**Server**(password)
$g \leftarrow \text{bytes\_to\_long}(password)^2 \mod p$		
$a \leftarrow \text{randint}(2, p-1)$		
$A \leftarrow g^a \mod p$		
	$\xrightarrow{A}$	
		$g \leftarrow \text{bytes\_to\_long}(password)^2 \mod p$
		$b \leftarrow \text{randint}(2, p-1)$
		$B \leftarrow g^b \mod p$
	$\xleftarrow{B}$	
$k \leftarrow B^a \mod p$		$k \leftarrow A^b \mod p$
$s \leftarrow \text{SHA-256}(\text{long\_to\_bytes}(k))$		$s \leftarrow \text{SHA-256}(\text{long\_to\_bytes}(k))$

To overcome possible MITM attacks Algorithm 1 may be vulnerable to, a Key Confirmation [Algorithm 2] step is executed in which an explicit check on the agreed key is performed. Assuming no cryptographic vulnerability against the selected hash function, as the shared key s is never sent in clear over the channel the key confirmation allows the two parties to securely verify their keys and authenticate. According to the specification, the two parties are classified as *Challenger* and *Responder*. No correlation should be assumed w.r.t. the client/server classification of Algorithm 1, each of the two parties can behave either as a challenger or as a responder. Actual implementation could employ a timer after which the Challenger process is started if no challenge is received from the other party.

Algorithm 2 Key confirmation

Challenger(s)		**Responder**(s)
$C \leftarrow \text{SHA-256}(\text{SHA-256}(s))$		
	$\xrightarrow{C}$	
		$R \leftarrow \text{SHA-256}(s)$
	$\xleftarrow{R}$	

if $R = \text{SHA-256}(s)$ **then**
    Authentication OK
    Send flag
else
    Authentication KO
end if