SiliconSAFE Password Protect

Developers Guide

Snap – The SiliconSAFE Protocol

Contents

1. Basic Snap

The Snap protocol is silicon safe's proprietary protocol for talking to the Password Protect appliance. It is possible to talk Snap directly to the Nanowall port on the Password Protect appliance or to talk Encapsulated Snap to the encryption processor port on the Password Protect appliance.
The client message format to talk to the password protect appliance is:
!!!<c><space><arg1><space><arg2><space><arg3>\c\n
where :

- • <c> is a single character command
- • <arg1>, <arg2> and <arg3> are optional arguments
- • command termination is carriage return <c> and newline <n>

The reply is a single byte indicating success or failure of the command.
1 SNAP is not an acronym – the protocol is named after the card game Snap – when two passwords match SNAP!

Command	Arguments	Meaning
w	username, password	Create a user account with the given 'username' and 'password'
c	username, password_attempt	Test the account for 'username' against password 'password_attempt'
u	username, password_attempt, new_password	If the account for 'username' has a password that matches 'password_attempt' changes the account password to 'new_password'
R	Username, admin_password, new_password	If the adminstrator password is admin_password, new_password 'admin_password' reset the password for user account 'username' to 'new_password'. This becomes a reset only account meaning that the account cannot be used for authentication ('c' command) until the password has been changed by a 'u' command
S	Username, Admin_password	Suspend the use of the account for user 'username'
E	Username, Admin_password	Enable a suspended account
D	Username, Admin_password	Delete an account
p		Pings the password protect appliance to see if it is responding

2. Encapsulated Snap

Encapsulated Snap wraps Snap in an encrypted and authenticate packet. The generic message format for encrypted Snap is:

l+n+t+i+{<iv+m,[n+t+i+iv+m]k2>}k1 where

Symbol	Meaning
+	Concatenation
l	Total length of message
n	Message identifier nonce (unique random)
t	Message type
i	Key index
iv	Initialisation vector to randomise encryption
m	Plan text message
{...} k1	Encrypt field under key k1
<...>	Padded field
[...]k2	Create message authentication code under key k2

Message exchanges are in pairs request/reply and are initiated by the client. In a request/reply pair the client chooses a unique random number as a message identifier and the appliance replies using the same message identifier. The initialisation vector iv is a random number chosen to randomise encrypted text. The client chooses a distinct iv for each request and the appliance chooses a distinct iv for each reply.
This generic structure is used for both creating session keys and for wrapping standard Snap exchanges.

3. The Integration Library

Silicon Safe provides a PHP class Python integration library for both Snap and encapsulated Snap. If we consider the code flow in a standard web platform the password check will look something like the picture on the left of figure 4. The replacement code flow is shown on the right.

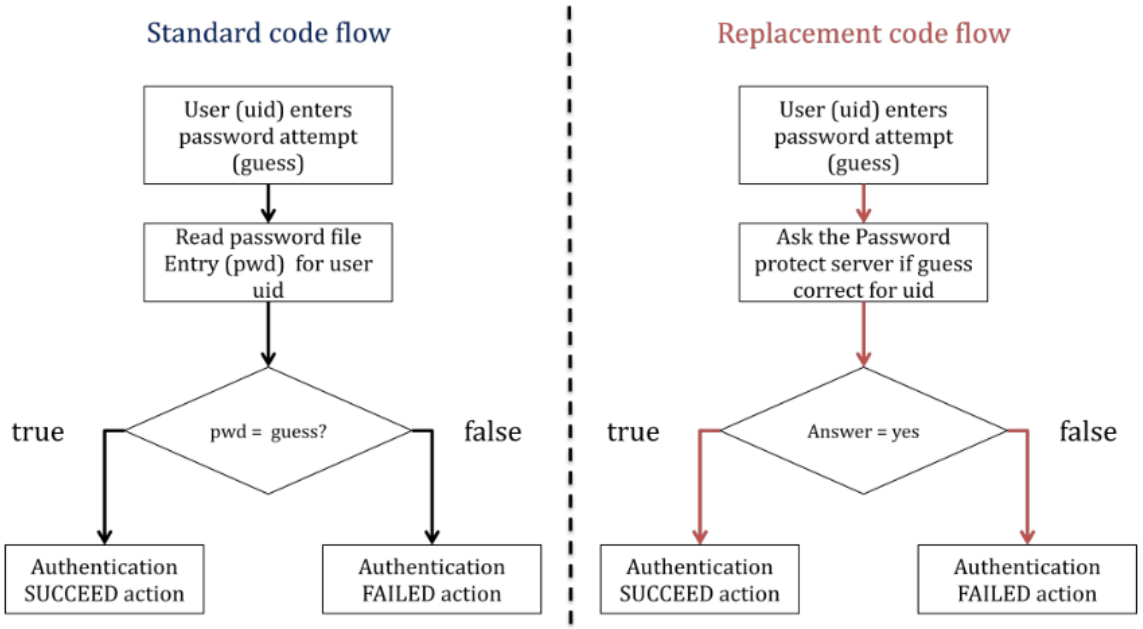

Figure 4

The integration is essentially the same whether one is using a remote Password Protect appliance as a service or a local Password Protect appliance as part of one's infrastructure.
The mechanics of actual integration, key management and SSL tunnelling and differences between Password Protect as-a-service and as-an-appliance are addressed in the Integration Manual.