The author of this paper developed an experimental prototype software system, the S/KEY one-time password system, as an authentication system for a UNIX environment. The S/KEY system prevents attackers from eavesdropping on network connections and obtaining user login ids and passwords for replay attack. The S/KEY system is very simple and effective. It uses computation to generate a finite sequence of single and one-time use passwords from a single secret. Afterwards, each time the user authenticates with the system, he should use a different password from the sequence generated. Since each time the user uses a different password for authentication, attackers having the password obtained from eavesdropping will no longer be useful, and replay attacks will be prevented. One weakness about this system is that the author only considers passive network attacks such as eavesdropping and does not protect passwords against active attacks.

The goals of the S/KEY authentication system are to prevent passwords from passive eavesdropping, to make the system easy to use, to provide automated operation, and does not have secret algorithms and stored secrets in the system. The S/KEY system has a remote client side that generates one-time passwords and a host server side that verifies the one-time password sent by the client. The secure hash function that used in the S/KEY system to apply to the user password is the MD4 Message Digest algorithm. Upon logging, after the user sent out its login name, the server will send a challenge including the sequence number N of the one-time password expected and a “seed”. The seed is used for the client to concatenate with the user password, and allows the user to securely use a single secret password for several machines. By applying MD4 N times over the password, the first one-time password is produced. The next one-time password is generated by running the password through MD4 only N-1 times. Each round by decreasing the times running the password through MD4, the user will finally has N one-time passwords. The user should use the first one-time password for authentication. And the next time the user wants access, he will need to send in the next one-time password in sequence and so on.

The author has implemented the S/KEY authentication system for experimental use. When using non-programmable terminals, users will need to print the list of their one-time passwords out and enter them accordingly. This really inconveniences users. So I think it is a drawback of the system. Furthermore, installing the S/KEY system requires replacing the login program. By doing so, the system source code must be obtainable. Thus, it makes the S/KEY system not possible to be implemented on any existing system. However, the S/KEY system has many advantages. One is that no secret information is stored anywhere, and the underlying algorithm can be made public without leaking the secret password. Another advantage is the user’s password is never out on the network. Only the one-time passwords that are hashes of the real password are sent over the network. Overall, I think the S/KEY authentication system is a simple, effective, and useful design.