



# Enhancing One-Time Passwords for Protection Against Real-Time Phishing Attacks

## TECHNOLOGY BACKGROUNDER

Phishing attacks are stealing consumer identities and creating major burdens on e-businesses. While one-time passwords deter offline phishing, by themselves they may not defend consumers and e-businesses against real-time “man-in-the-middle” phishing attacks.

This whitepaper provides an overview of a unique anti-phishing research concept being explored at RSA Security. It demonstrates how a simple change in the authentication interface can significantly improve protection against phishing and other possible online attacks.

The paper shows how *password hashing* can be utilized within a new *Password Protection Module* to systematically protect against phishing of passwords, and puts forth a proposed solution. Because this research concept continues to evolve at RSA Security, please contact your RSA Security account representative for updated information on the program and technology.



## Table of Contents

The Need to Protect One-Time Passwords .....	1
Enhancing Basic Countermeasures .....	1
The Role of Two-Factor Authentication .....	2
Looking for the “Man-in-the-Middle” .....	3
Hashing Passwords with Application Identifiers .....	5
Preventing Real-Time Attacks .....	6
The Password-Protection Module .....	6
Ensuring Application Authentication .....	8
Technology Sidebar .....	9
Summary .....	9
About RSA Security .....	10



## **The Need to Protect One-Time Passwords**

Phishing attacks the fundamental premise of e-business—the trusted online relationship between two parties. This is an emerging class of online fraud in which “phishers” generate e-mail that appears to be legitimate in an attempt to trick users into providing userIDs, passwords, Social Security numbers, credit card numbers and other private information that can be used for identity theft.

Phishers employ online social engineering to lure users to reveal passwords and other sensitive information. For example, they often send mass quantities of e-mails claiming to be from a legitimate organization with which targets are likely to already have trusted business relationships. Individuals receiving the e-mail may click through and be directed to a site with a familiar look-and-feel, where they are encouraged to enter personal information. As the victim submits a form to the illegitimate site, the data is sent to the criminal who can use it to crack personal accounts and conduct identity theft.

Phishing is a threat to both B2B and B2C e-business, and this threat is growing. Phishers have learned lessons from spammers and realize that e-mail is an inexpensive way of “phishing” for potential victims. The Anti-Phishing Working Group estimates that phishers are able to convince about five percent of recipients to provide personal information. This non-profit industry association estimates there were 1,422 unique phishing attacks in June of 2004. This compares with a total of 176 in January to present an increase in the crime rate of over 700 percent in just a half a year.

Both static passwords and stronger authentication measures like one-time passwords were designed to establish proof of identity to applications in which the user has a great deal of trust. However, even one-time passwords do not provide complete protection against real-time attacks. New solutions are needed to enhance one-time passwords for maximum protection against real-time phishing attacks to prevent sophisticated thieves from intercepting passwords and using them immediately to perpetrate fraud.

*For an overview of phishing threats and detailed information on the role of two-factor authentication in mitigating phishing threats, please visit [www.rsasecurity.com](http://www.rsasecurity.com) and download the whitepaper “Protecting Against Phishing by Implementing Strong Two-Factor Authentication.”*

## **Enhancing Basic Countermeasures**

B2B and B2C companies can take a multi-layered approach to protect against phishing attacks. User education is critical, because users must learn to become suspicious about web links sent by e-mail. For example, a Gartner study found that 41 percent of respondents thought they had received an e-mail that either looked like or definitely was a phishing attack.



As phishing attacks become more prominent, users are gradually learning to be suspicious of web links sent by e-mail, and over time increased user education will be the best long-term antidote to phishing.

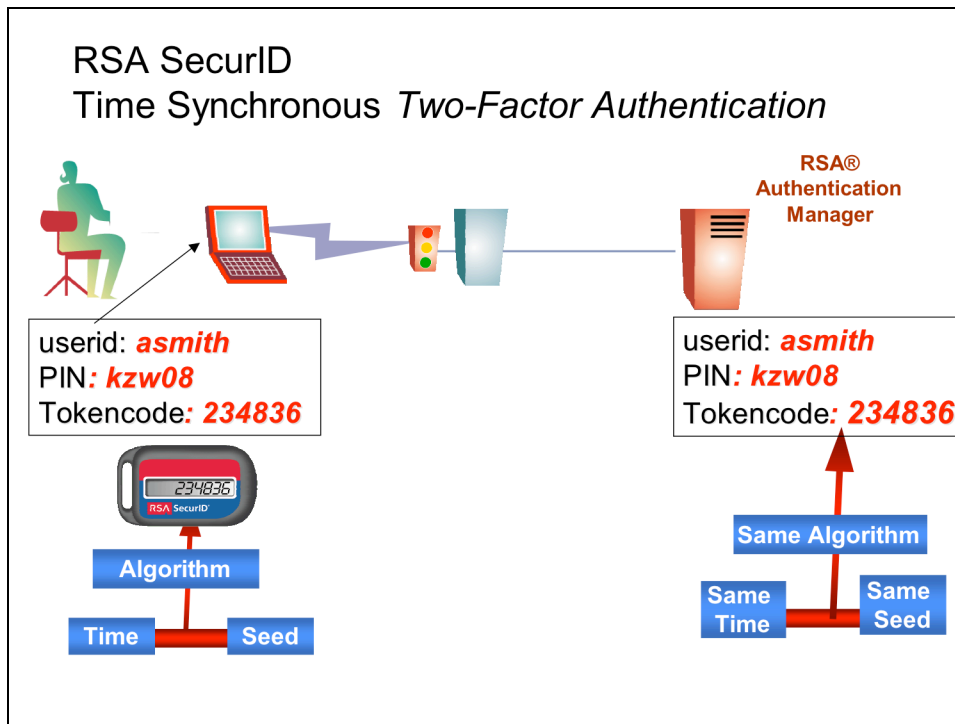
Organizations can also implement phishing filtering using trust lists and content filters, just as many organizations do today to combat spam. However, there is the risk that users might override the settings of the trust lists and content filters if they are deceived by the phisher into thinking that the settings might be wrong. Although user education and phishing filtering are important countermeasures, a window of attack remains open because social engineering efforts will occasionally override users' caution and circumvent detection.

## **The Role of Two-Factor Authentication**

Passwords are the most basic forms of authentication, but they are easily guessed, frequently forgotten and expensive to maintain. Passwords are static, typically have a relatively long lifespan and are often reused for multiple applications, thereby making them particularly attractive targets. They are vulnerable to phishing attacks because the user can potentially be duped into entering a password into a fraudulent web site or application. A phisher can then collect the information and use it later for fraud and theft.

Two-factor authentication provides dramatically improved protection, since users enter something they know—a Personal Identification Number (PIN)—and something they have—the changing code on an authenticator the size of a keychain. Strong, two-factor authentication is a proven solution for security within the enterprise, but it is designed for authenticating the user to the application—not necessarily for authenticating the application to the user.

For example, many enterprise users today use two-factor authentication to access e-mail, Virtual Private Network (VPN) services and wireless networks. An organization can provide each user with an RSA SecurID® authenticator so they can enter their PIN and a code from a token or USB device that changes every 60 seconds. Each authenticator has a unique symmetric key which generates a new, unpredictable passcode every sixty seconds. (See Figure 1).



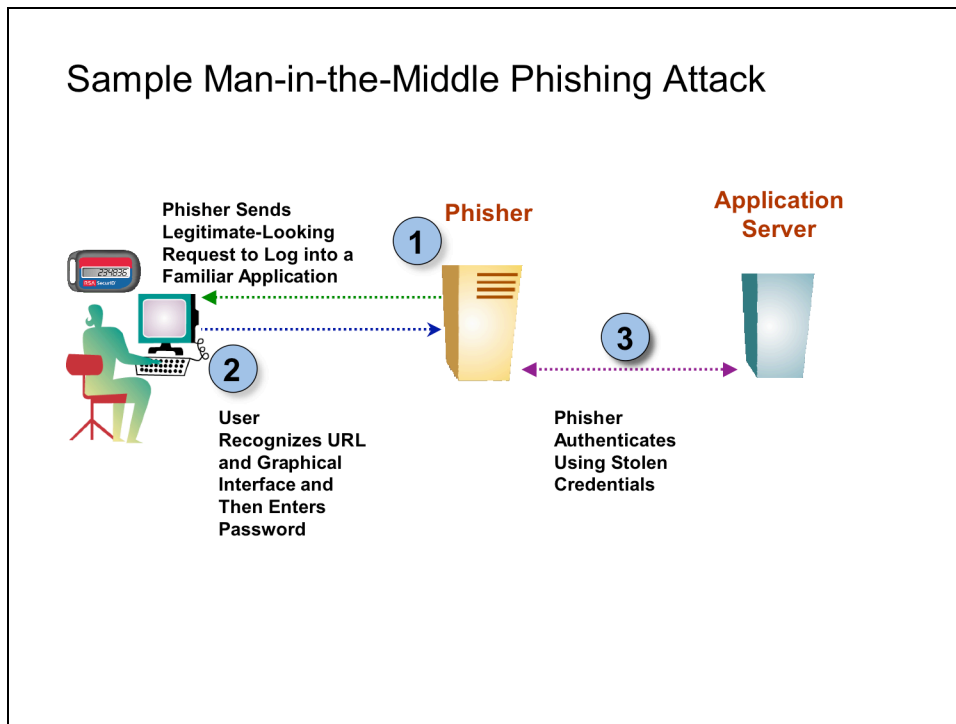
The RSA SecurID® solution provides excellent protection against the vast majority of phishing attempts because the one-time passcode changes every 60 seconds. A phisher therefore cannot harvest passcodes for later use because the passcode changes so frequently. An attempt to use a passcode hours or even minutes old will result in failure.

### Looking for the “Man-in-the-Middle”

RSA Security continues to develop new countermeasures to defend against emerging threat categories. If the online threat involves capturing passwords for later intrusion attempts, RSA SecurID two factor authentication solutions provide excellent protection and can help mitigate risk.

There is an increasing threat posed by technologically advanced phishers that try to insert themselves into an authentication process so they can conduct real-time fraud. “Man-in-the-middle” is a traditional cryptographic term for an attacker who interposes between two endpoints and forwards messages back-and-forth, perhaps changing them.

There are many variations on the man-in-the-middle threat, but in a typical attack the phisher sends a legitimate-looking request supposedly from a trusted site asking the user to log into a familiar application. (See Figure 2).



Since the user recognizes the “similar” URL and the familiar visual look-and-feel, they might be persuaded to enter the one-time passcode. The phisher then immediately authenticates to the legitimate application using the stolen credentials and conducts fraud. The phisher may even pass back an error message to the user asking them to authenticate again, leaving the user completely unaware of the identity theft.

Man-in-the-middle attacks threaten many authentication method, including:

- Static passwords
- One-time passwords based on time and/or events
- Challenge-response authentication, where a user enters a challenge value into a token

If the user has confidence in the application—for instance, because the user knows and enters the correct URL, or launches from a trusted shortcut—then the user is safe in sending a password over a secure channel to this application. The secure channel itself protects against eavesdropping, and server certificates defend against URL spoofing. But once that trust is broken—for instance, if the user clicks on an un-trusted URL that turns out to be a man-in-the-middle phisher—some additional protection of the password is needed.

One vendor has suggested a solution based on allowing the application to return a portion of the next one-time passcode. The user could then verify that the legitimate application



is involved and conclude that the application is not just a phisher harvesting passcodes for later use. But if the phisher is *already* in the middle and has intercepted the user's initial login, the phisher itself will also obtain the portion of the next one-time passcode—and could thereby impersonate the actual application to the user. The actual application is indeed involved, and the user learns this. But the user does not realize that the phisher is involved as well. In addition to this major shortcoming, this approach requires that the user check or enter the next one-time passcode, creating an additional step that would result in user frustration.

Man-in-the-middle attacks create an increased exposure point for organizations concerned with protecting against phishing attacks. If a new means of protecting against these real-time attacks is not developed, phishers will increasingly employ clever methods to insert themselves between users and applications to steal identity information and perpetrate theft.

The techniques to be outlined in the rest of this paper are well suited to one-time password systems where the one-time password is the combination of a PIN and a time-sensitive tokencode. However, they can also be applied beneficially to less secure, static password systems. The generic term “password” is used to cover both static and one-time password authentication processes throughout the rest of the white paper.

## **Hashing Passwords with Application Identifiers**

Hashing (re-coding) passwords with an identifier associated with the requesting application is a well-known technique for protecting against misuse of the password by a rogue application that receives it, and this capability is built into many password protocols today.

The application identifier would be different for every application so that each application has a unique application identifier. If a phisher intercepts the hashed password it would have to reverse the hash function—which is extremely difficult—or perform a brute-force search for a one-time passcode in real-time—which is theoretically possible but economically unattractive.

The greatest challenge in this approach is to ensure that the hashing actually takes place. Passwords are typically entered into a phisher's application like any other form data, except that the characters are not “echoed” or displayed back to the user. The password is then provided directly to the requesting application, with no hashing taking place. While legitimate applications would likely be willing to perform the hashing, a phisher's application would not—and it is the phishers' applications that we need to be concerned about.

The hashing therefore needs to be done by some other agent on the user's computer other than the requesting application. One alternative is to immediately hash the password as soon as the user enters it. This can be done for instance, using the clever PwdHash



browser plug-in from Stanford University. It hashes the fields before sending, but first the plug-in must “know” that a field has a password.

Phishers may be able to circumvent solutions like this using special application code. In addition, the user has no direct confidence that hashing has taken place correctly—since an untrustworthy application could claim to a user that it has applied a hash when indeed it has not. This method could lead to a potential “cat-and mouse” game with the plug-in looking for password fields while phishers develop new methods to escape detection.

A new approach is needed to ensure that passwords are hashed before transmission, and this requires a new user interface and the ability to hash passwords with application identifiers before they are transmitted.

## **Preventing Real-Time Attacks**

RSA Security is exploring a new technology for hashing passwords to provide protection from real-time attacks. It will allow passwords to be hashed with an application identifier before transmission to protect against reuse in a different context.

This approach is based on standard-and-proven security protocol engineering techniques. It requires that the application identifier be something specific to an application so that it is difficult for one party to effectively employ another party’s application identifier. Depending on the trust assumptions, the application identifier could be based on an IP address, URL, domain name, public key, etc.

For example, if a URL is used as the application identifier it would be difficult for a rogue application to receive messages sent by the user’s computer to the legitimate application’s URL without going to the extra effort of corrupting routing tables or spoofing a DNS server. Thus, if the illegitimate application claims the legitimate URL used as the application identifier, it will not be able to receive the hashed password because it will instead be sent to the legitimate URL. Any illegitimate application could only receive hashed passwords computed with its own URL.

By hashing passwords before transmission, organizations can provide static or one-time passwords with significant protection against phishing attacks. This method is analogous to the difference between paying in cash versus paying by check. A thief can spend cash anywhere, but if the thief cashes a check there is more of an audit trail, and a thief cannot easily cash a check made out to someone else. Hashing applies more context around a session to increase protection levels and lower the likelihood of attack.

## ***The Password-Protection Module***

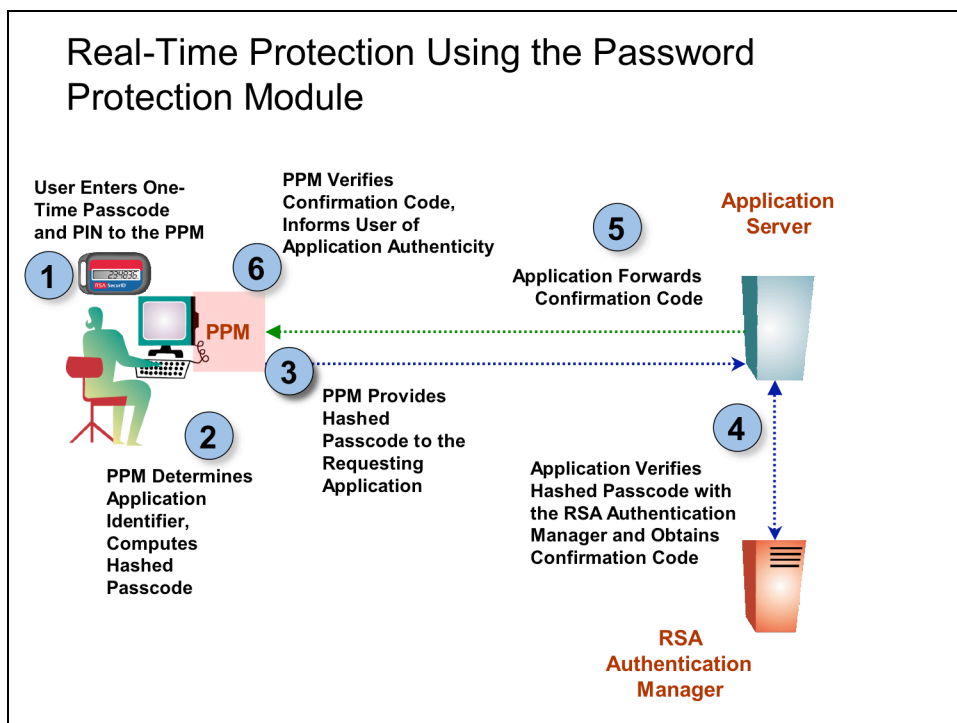
As an improvement on the existing practice of treating passwords like any other data to be entered into a form, RSA Security is exploring a new technology, tentatively referred to as the Password-Protection Module (PPM).

Passwords would be entered into a special software module on the client platform beyond the purview of the requesting application. The PPM could be part of the operating system or the browser, or it could potentially be made available as a third-party application. It would hash the password before transmission, thus protecting a phisher from obtaining passwords by presenting a fake application interface.

Instead, the user would invoke the PPM whenever a password is required. The PPM would be accessible only via a reserved control sequence on the client. For example, it could use a function key not available to web applications.

The industry term for this concept is a “secure attention sequence”. It can be compared to other system-wide key sequences, such as Alt-Control-Delete to terminate an application and/or lock or unlock a client computer.

When users are asked to enter a password, they would execute the secure attention sequence and then enter the password into the PPM. The PPM would then hash the password with an application identifier before providing it to the application. Passwords would not be entered as other form data, but would instead be entered into the PPM as in the following example. (See Figure 3)



The user is presented with an application that is requesting a password. As a general habit for authenticating to applications or web sites, the user invokes the PPM via a to-be-determined keyboard shortcut for the secure attention sequence.



This launches the PPM, and the user enters the PIN and token code directly into the PPM application, which examines the application that the user was interacting with prior to invoking the PPM, determines its application identifier (e.g., URL, domain name, etc.), and hashes this passcode with the application identifier.

It provides a hashed password to the requesting application, perhaps placing it within the field that initially requested the password. The requesting application then verifies the hashed password with the authentication server (e.g., the RSA® Authentication Manager), which provides a confirmation code to the application.

While phishers have become extremely proficient at simulating application interfaces to steal passwords, the PPM avoids this problem by encouraging users to enter passwords into a secure module where they can be hashed with an application identifier before being transmitted.

The PPM complements filtering, because even if the filter does not catch the phishing attack—or if the user overrides the filter—the user can be sure that the password was hashed before transmission. It also encourages the easy adoption of more trustworthy computing habits. Since the user is confident that the hashing will occur at the desktop, activating the secure attention sequence becomes as easy a security habit as locking a car as you get out.

### ***Ensuring Application Authentication***

While hashing protects the password against misuse, it does not confirm to the user that the application is trustworthy. For example, a phisher could pretend to “authenticate” the user, and then request other sensitive data. Mutual authentication is necessary to provide this assurance. Under one proposed approach being evaluated at RSA Security, passwords are hashed by the application in a different way to provide a confirmation code to the PPM. (See Figure 3)

The confirmation code is a second hash value obtained from the same principle of hashing the password and an application identifier. The PPM verifies the confirmation code and indicates to the user that the application indeed knows the user’s password. This results in mutual authentication, since the user has been authenticated to the application and the application has been authenticated to the user.

## Technology Sidebar

The phisher's goal is to either impersonate the user to a legitimate application or to persuade the user that the phisher's application is legitimate. This technology sidebar provides a more in-depth look at the protocol details, using the following notation:

- $P_t$  = password at time  $t$
- $ID_A$  = application A's identifier
- $P_{A,t}$  = hashed password =  $H(P_t, ID_A)$
- $C_{A,t}$  = confirmation code =  $H'(P_t, ID_A)$
- $H, H'$  = one-way hash functions

The following example provides the protocol details:

- User enters  $P_t$
- Password protection module determines  $ID_A$
- Module computes hashed password  $P_{A,t} = H(P_t, ID_A)$
- Module provides  $P_{A,t}$  to application
- Application verifies  $P_{A,t}$
- (*Mutual authentication steps*)
- Application computes confirmation code  $C_{A,t} = H'(P_t, ID_A)$
- Module obtains and verifies  $C_{A,t}$
- Module indicates result to user

Impersonating the user to application B requires the phisher given  $P_{A,t}$  to find  $P_{B,t}$ . Similarly, persuading the user that application A is legitimate requires the phisher, given  $P_{A,t}$ , to find  $C_{A,t}$ . The goal is to ensure that a brute force search is not at all desirable to a phisher, and this scenario would require a brute-force search for  $P_t$  if  $H, H'$  are one-way. This can be demonstrated by considering the following example parameters:

- $H()$  takes 1 sec. on typical client
- $P_t$  is 12 digits long (e.g., 6-digit PIN, 6-digit tokencode)
- $P_t$  is valid for three minutes, including the time synchronization window
- Phisher has attack machines 100 times faster than client

Using these assumptions, a brute-force search in real-time would require the phisher to have  $0.5 \times 10^{12} / 18,000$ , or 28 million attack machines to succeed in real-time with a probability for success of 50 percent—which is not an appealing business proposition for the phisher.

## Summary

Password hashing is a helpful tool for protecting one-time passwords against phishing attacks, but only if it is done systematically. Technology under evaluation by RSA Security proposes a new user interface to ensure systematic hashing.



When an application requests a static password or one-time passcode, the user invokes the PPM and enters the information so the PPM can hash the codes before transmission. This approach creates a new user paradigm for more trustworthy computing. Users will learn never to tell their passwords to strange applications, and they just routinely enter the secure attention sequence before entering any password information.

While strong two-factor authentication mitigates against the threat of non real-time attacks—which still comprise the vast majority of phishing attempts—the PPM enhances one-time passwords for protection against real-time phishing attacks.

RSA Security continues to develop innovative approaches to protect against emerging threats. This research concept for protecting one-time passwords against phishing attacks is not yet productized, but is being developed within RSA Laboratories and the office of the CTO. If you are interested in further information, please contact RSA Laboratories.

### ***About RSA Security***

RSA Security Inc. helps organizations protect private information and manage the identities of people and applications accessing and exchanging that information. RSA Security's portfolio of solutions – including identity & access management, secure mobile & remote access, secure enterprise access, secure transactions and consumer identity protection – are all designed to provide a more seamless e-security experience. Our strong reputation is built on our history of ingenuity, leadership, proven technologies and our more than 15,000 customers around the globe. Together with more than 1,000 technology and integration partners, RSA Security inspires confidence in everyone to experience the power and promise of the Internet. For more information, please visit [www.rsasecurity.com](http://www.rsasecurity.com).

RSA, RSA Security, SecurID, the RSA logo, and Confidence Inspired are either registered trademarks or trademarks of RSA Security Inc. in the United States and/or other countries. All other products and services mentioned are the trademarks of their respective owners.

© 2004 RSA Security Inc. All rights reserved.

RTPH-WP 0904