When the Password Doesn't Work

Secondary Authentication for Websites

The authors discuss secondary authentication mechanisms, emphasizing the importance of assembling an arsenal of mechanisms that meet users' security and reliability needs.

Robert W. Reeder
Microsoft Trustworthy Computing

Stuart Schechter
Microsoft Research

Nearly all websites that maintain user-specific accounts employ passwords to verify that a user attempting to access an account is, in fact, the account holder. However, websites must still be able to identify users who can't provide their correct password, as passwords might be lost, forgotten, or stolen. In this case, users will require a form of secondary authentication to prove that they are who they say they are and regain account access. Websites can use a variety of secondary authentication mechanisms, such as

- sending users an email with an access key, often embedded in a hyperlink;
- sending users an SMS (text) message with an access key;
- asking users to answer a security question;
- asking users to supply an old password; and
- asking a friend or other third party to verify users' identity.

The consequences of secondary authentication failures—that is, falsely rejecting a legitimate user or falsely accepting an impostor—are high. Unless additional backup authentication mechanisms are available, a false reject will result in permanent account loss. On the other hand, if attackers can target secondary authentication more easily than passwords, the mere presence of secondary authentication mechanisms makes accounts less secure.

Despite the high stakes of secondary authentication systems, little research exists on how to design them well. Consequently, secondary authentication systems still have widespread weaknesses. Common problems include an overreliance on weak security questions that don't match the authentication system's security level to the account's value, and failure to keep secondary authentication data up to date.

Different users will want to make different tradeoffs between preventing false rejects and false accepts. Users who place high value on their accounts might be more willing to spend time configuring a robust secondary authentication strategy than users with less valuable accounts. Thus, a one-size-fits-all approach to backup authentication might not be appropriate. We offer suggestions for developing secondary authentication systems that let users choose a strategy that's right for them.

Secondary Authentication Mechanisms

Recent news stories have shown just how vulnerable secondary authentication mechanisms can be. In 2008, an operative broke into US vice-presidential candidate Sarah Palin's webmail account by correctly guessing—with help from publicly available information—that the answer to her security question "Where did you meet your spouse?" was "Wasilla High School."[1] In 2009, a French hacker obtained and publicly shared confidential Twitter company data from executives' accounts.[2] This hacker's methods included both guessing answers to security questions and having a password-reset email sent to an account he owned.

 1540-7993/11/$26.00 © 2011 IEEE COPUBLISHED BY THE IEEE COMPUTER AND RELIABILITY SOCIETIES

Secondary authentication presents unique usability challenges in comparison with passwords and other primary authentication mechanisms for which it's often expected that users memorize a complex series of characters or learn to use complex mechanisms through repeated practice. Because the need for secondary authentication mechanisms is intended to be rare, users won't learn how to use them through repeated practice. Indeed, users often resort to secondary authentication because they haven't adequately practiced their primary authentication mechanism—perhaps they've forgotten their password after a long period of inactivity, or have been relying on a device-bound password manager but recently changed devices. Therefore, any expectation of practicing a secondary authentication mechanism is unrealistic. In addition, whereas users who forget their password might resort to secondary authentication, those who fail secondary authentication are simply out of luck.

Because many different authentication mechanisms are available, the question arises of what a website should look for in a secondary authentication mechanism. We consider four key criteria:

- *reliability*—the likelihood that the account holder can successfully authenticate.
- *security*—the likelihood that an attacker can impersonate (falsely authenticate as) an account holder, or the time and effort required to do so. Separate security evaluation criteria are usually required for each threat category.
- *authentication efficiency*—the cost to the account holder (time and effort) to authenticate, as well as any costs the Web service incurs.
- *setup efficiency*—the cost to the account holder (time, effort, and privacy sacrificed) and Web service to configure and reconfigure the authentication mechanism.

A perfect authentication mechanism would exhibit all these criteria to a high degree. However, in practice, no mechanism is perfect. Whereas a successful authentication system should exhibit all these criteria to some degree, they can be traded off against each other through mechanism selection and configuration to suit a particular purpose. For secondary authentication, given the dire consequences of both false accepts and false rejects, reliability and security are generally both extremely important and should only be traded off against each other to a small extent. Fortunately, because secondary authentication is relatively rare, these mechanisms don't need the same efficiency as passwords. In fact, all the secondary authentication mechanisms we describe require more time and effort—that is, less authentication and setup efficiency—than entering a series of characters from muscle memory. Secondary authentication mechanisms are less efficient than passwords; if they had reliability, security, and efficiency competitive with passwords, we'd use them for primary authentication.

Our criteria are similar to those others have used to evaluate secondary authentication mechanisms. In particular, Mike Just developed reliability and security criteria for evaluating security questions.[3] Just's criteria are specifically designed to evaluate security questions (for example, he considers guessing difficulty and memorability), whereas we chose more general criteria applicable to any authentication mechanism.

Note that a website can implement and configure a given authentication mechanism in myriad ways that affect how it would rate on these criteria. For example, a website using security questions can require the answer to one security question for authentication, or it can require k correct answers to n questions. We use our criteria to discuss the trade-offs among mechanisms and their configurations, but we don't attempt to formally rate them.

We divide the secondary authentication mechanisms into two categories: knowledge-based authentication, in which the user knows or stores information and must provide that information to the server to authenticate, and transitive authentication, in which the system delegates the authentication task to another system.

Knowledge-Based Secondary Authentication Mechanisms

Knowledge-based authentication systems are popular with websites and other service providers because they're relatively simple to implement and don't rely on external infrastructure, such as other systems or special hardware. If a user's infrastructure is working well enough to connect to a website, knowledge-based authentication can be used.

Security questions. Security questions—also called *secret questions* or *challenge questions*—rely on the assumption that their answers, like passwords, are easy for account holders to answer but hard for attackers. The setup cost of a security question is relatively low: users select from a choice of questions and provide the correct answer—the answer that the website will require if this question is used for authentication. Researchers often categorize security questions by whether they ask for factual information about users (for example, "What was the name of your first school?") or their preferences or opinions (for example, "Who is your favorite singer?").

A security question can fail because the answer is

- *not configurable*—some questions can't be answered

because they don't apply to a user (for example, "What is the name of your first pet?") or because the user might not know the correct answer at setup time (for example, "What is your library card number?").
- *forgettable*—users might forget the content or formatting of the answer they selected at setup time. This is especially a problem for answers to questions about preferences, which might change over time (for example, "What is your favorite song?").
- *known*—the attacker might be someone who already knows the answer, such as a significant other or family member.
- *guessable*—the correct answer to a question might be popular or drawn from a small set of possible answers, and thus vulnerable to attacks that guess the most popular answers for the entire population of users or a subset of the population to which the user is known to belong. For example, in the US, "George Washington" is a popular answer to the question, "Who is your favorite historical figure?"
- *researchable*—the correct answer might be found online or by asking someone who knows the answer.

Research on security questions dates back to the late 1980s,[4] and the early results would be unlikely to inspire a website to adopt a single question as an authenticator. Because many websites did adopt a single question for backup authentication, one might assume that questions improved in the intervening decades. However, our 2008 research showed that the security questions that the four largest webmail providers use remained inadequate.[5] Half the questions had answers that at least 20 percent of users forgot within six months. Half the questions were configured with answers that an untrusted acquaintance could guess at least 15 percent of the time. One-third of the questions were susceptible to a statistical guessing attack at least 15 percent of the time. All questions exhibited at least one of the first four failures above for at least 9 percent of the users. None of the questions we examined was reliable and secure enough to be sufficient to authenticate users on its own.

Moreover, security questions have become considerably weaker authenticators in the age of social networking, as answers become more easily researchable.[6] Treating knowledge of users' work history, schooling, family, or preferences as secrets makes less sense as people share increasingly more personal information online.

Some sites let users compose their own security questions in hopes that personally customized questions will be more memorable and secure. Both our work and that of Mike Just and David Aspinall show that these questions are no better.[7] This is initially surprising to those who find user-selected questions intuitively attractive. To understand why this approach fails, consider that to avoid the pitfalls of secret questions, users selecting a question would need to choose something that has an answer that's simultaneously memorable, not researchable online, reasonably unpopular with other users, and unknown by any untrusted acquaintances. In addition to learning all these possible failure models, users would need to keep them all in mind and make accurate judgments about the likelihood of each. When stated this way, it's not surprising that our sample of user-generated security questions was littered with questions that exhibited one or more of these failures—for example, "What color are my eyes?"

Many techniques can improve questions' security and reliability, though in a limited way. First, the provider can prevent users from choosing a question if the answer they provide is dangerously popular. For example, if more than one in every 500 users had provided a particular answer, the system would ask users to choose another question. We don't recommend asking users to choose another answer—they would be less likely to remember it than the first answer.

Second, providers can require answers to multiple security questions to let users regain access to their account. Using multiple questions strengthens security, but might exacerbate the problem of users forgetting their answers. Providers can mitigate this problem by requiring a subset of correct answers to an even larger number of questions. However, this approach quickly becomes unwieldy, especially given the small number of good security questions available. One system that achieves its goals by growing the number of questions is Markus Jakobsson and colleagues' preference-based authentication scheme, which reduces the per-question effort by placing answers in a Likert scale rather than free response. However, the scheme requires users to answer 15 security questions for both configuration and authentication.[8]

Finally, periodically requiring users to reenter or change their answers—to keep the answers fresher in their memories—might reduce the problem of forgotten answers.

Printed shared secrets. Unable to consistently remember the answers to security questions, many users write down their answers, obtaining reliability at some potential cost to security. Because users are already storing secrets on paper—and, in many cases, the benefits outweigh the risks—system designers might not only want to condone this behavior, but support printed shared secrets.

Authentication mechanisms designed expressly for printed shared secrets can have many benefits over simply letting users write down the answers to secret ques-

tions. The system can generate random keys to use as a shared secret, rather than the answer to a personal question, eliminating the possibility of guessing by a friend or statistical attacks that leverage popular answers. The system designer can limit the impact of man-in-the-middle attacks by printing an indexed array of keys on a page and requiring a small, randomly selected subset (for example, three) identified by an index to authenticate. An MITM attacker might be able to convince the user to type the correct keys for one authentication request, but will find it difficult to collect enough information to authenticate in the future without continued assistance.

Previously used passwords. Users frequently forget passwords when they're forced to change them. Previously used passwords can be used in conjunction with other secondary authentication mechanisms to verify identity and can be particularly useful when an account has become compromised via other secondary authentication mechanisms, such as security questions. The legitimate user will know the account password that predates the compromise, but the attacker who had compromised the account would not.

Transitive Secondary Authentication Mechanisms

Transitive authentication systems "pass the buck" for authenticating users to a system or person that's hopefully better equipped to authenticate users than the website.

Email-based authentication. The most common form of transitive authentication is email-based authentication, which relies on the assumption that only users will be able to access a secret sent to their email account.[9,10] The website often sends the access code embedded in a link back to the website, so that users don't have to type the code or copy and paste it. Many websites already employ users' email addresses as their user ID, ensuring an address will be available without any setup cost.

Transferring the task of authenticating users to email providers often makes sense; email is a highly valued information asset, and users already entrust providers to authenticate them in a reliable and secure manner. Thus, many websites rely exclusively on email-based authentication when users forget their passwords.

However, email-based authentication has its limitations. Users might mistype an email address. When users part ways with a job, school, ISP, or other institution that has hosted their email address, they might no longer have access to it—another person might even be assigned that email address in the future! Websites can deal with the problem of mistyped email addresses by sending an email when the address is configured, including a link with which users can confirm that they received the email. If the email isn't confirmed, the website can ask the user to correct or confirm that address. However, this process adds to setup cost.

The popularity of email-based secondary authentication puts webmail service providers in a difficult position. Other services' reliance on email-based authentication increases the consequences of any authentication failure for the email service; an attacker who compromises a webmail account might obtain transitive access to the user's other accounts by initiating email-based secondary authentication for those sites. Users who lose access to their email account might transitively lose access to other accounts for which they don't frequently use their password, assuming that they'll always be able to change it using email-based authentication. Moreover, webmail providers can't as easily rely on email-based secondary authentication to help users who have forgotten their webmail passwords; the user might not have another reliable and secure email account to which the buck can be passed.

Phones or other devices. Phone numbers, like email addresses, can be used for transitive authentication. A website can send users an SMS message or use an automated voice system to call users and provide an account recovery code. Like email-based authentication, security and reliability rest on the phone's longevity and security. In theory, any device that can receive communications on behalf of users can serve in this role.

Device-based authentication can be somewhat less efficient than email-based authentication, as users might need to copy an access code manually from phone to website. On the other hand, the phone might require less authentication than the email account. However, given the frequency with which users lose phones and other portable devices, a device alone might not provide sufficient security for many situations.

Trustees. People are very good at identifying family members, friends, and close acquaintances in real-world social situations. We can simultaneously leverage such cues as appearance, voice, and both physical and verbal mannerisms. Websites can also leverage this human ability. John Brainard and colleagues were among the first to propose authentication based on "somebody you know."[11] We subsequently implemented and tested a system that relies on social authentication, in which other people attest to a user's identity.[12] Our system collects the names and contact information (email or phone number) of people users trust. Should users run out of other authentication options, they can contact these trustees, in a hard-to-spoof way (preferably

over the phone or in person), to ask for help. Each trustee then connects to the service for transitive authentication of contact information (for example, email- or SMS-based authentication). Once trustees prove their own identity and identify the account holder they wish to help, they might be asked to answer questions designed to help them avoid unwittingly assisting an impostor intent on compromising the account. Finally, the system provides trustees a code to give to the account holders, who provide the code to the website as evidence that the trustees have authenticated their identity.

Naturally, a user's trustees might not all be available at a given time. To compensate, websites can require a subset of the configured trustees to assist the user. Still, contacting multiple trustees is far less efficient than answering a question or receiving an email. Social authentication works best as a last resort if other approaches fail, or in combination with other authentication mechanisms. For example, a system might require users to answer a security question and contact any one of four trustees.

When we tested our system's efficiency, reliability, and security, usability results showed it to be reliable but, as predicted, far less efficient than most other schemes. To test security, we simulated two kinds of attacks. First were generic, email-based attacks, in which trustees receive generic-sounding attack emails purportedly from account holders who have forgotten their password. Second were targeted, phone-based attacks, in which attackers known to the account holders (family members or close friends) call the account holders' trustees and try to convince them to give up a recovery code. Very few trustees (roughly 4 percent) fell for the attack, and attackers would need multiple trustees to do so to compromise an account. In the phone-based attack, trustees were much more likely to help a potential attacker when the request came in the form of a phone call from someone the victim knew. Still, fewer than 50 percent of calls that reached a trustee yielded an account recovery code.

In-person proofing. In some situations, two undifferentiated individuals might claim to be owners of the same account. In such a case, having a trusted authority, such as a bank or a government official, attempt to ascertain and attest to the individual's identity might be beneficial. One obvious limitation of this approach is that there might be insufficient information with which to link an account to an in-the-flesh person. However, for accounts with conflicting ownership claims, in-person proofing has a distinct advantage: an impostor might not want to take the risk of appearing before an authority person, at a scheduled time, to make an ownership claim. Thus, requiring that an account in conflict be resolved via in-person proofing might resolve the identity conflict even if the mechanisms can't achieve a true proof of identity.

Assembling a Secondary Authentication Strategy

Different secondary authentication mechanisms achieve different reliability, security, and efficiency levels. The right mechanism for a given website or user account might depend on each account holder's requirements. For example, the security of an account used to read the news or check the weather is likely less important to a user than one that stores personal email or medical history. Some users might value reliability over security, whereas others with more private data might feel otherwise. Accounts' value also changes over time. Users who have just signed up for an account to try out a new service likely have less invested in the account than users who have been using their account for a year.

For websites with predominantly low-value accounts, a single email address or security question might be sufficient information with which to authenticate a user. For websites with more valuable accounts—especially email providers that act as secondary authenticators for other accounts—more complex authentication strategies might be required to facilitate stronger authentication security requirements. We target our guidance at these websites, as their requirements are the most challenging to meet.

Combine Authentication Mechanisms

Combining multiple authentication mechanisms can improve both reliability and security. The more authentication options available, the more likely users will be able to recover their account. Security can be achieved by requiring users to successfully complete more than one authentication task, raising the amount of evidence users must provide to prove that they're the account holder. Some authentication mechanisms can be used more than once, such as by configuring multiple security questions or trustees.

Designing an interface with which users can configure multiple authentication mechanisms and choose the subset sufficient to authenticate might seem like a daunting task. However, we found the metaphor of an exam—in which each available authentication task lets users score "points" toward a passing score—to be quite effective.[13]

Give Users Control over the Authentication Strategy

Users are unlikely to configure a robust secondary authentication strategy if they aren't convinced of the se-

curity or reliability needs. For example, users who don't believe security questions have value might respond to required questions with random strings or glib answers.

Moreover, account creation—when authentication is first configured—is a time when users might be the least motivated to properly configure their secondary authentication options. Users might be trying the service, uncertain whether they'll actually use the account in the future. A newly created account won't initially contain valuable data. In addition, at account setup time, users might not yet trust the website with the personal information required for secondary authentication. Thus, convincing users to invest heavily in secondary authentication when setting up an account is difficult.

Account holders might be more likely to comply with requests to configure more complicated authentication mechanisms if the request comes after they've been using their account for some time—when it has a higher value to them. The website can further encourage users by providing them with reminders of the account's value, such as the number of emails or contacts stored that would be lost if they were locked out of the account or if it was compromised.

In the end, account holders are the ones who will suffer in the event of authentication failure. Because they bear the consequences, they should have the option to add more secondary authenticators and adjust the authentication threshold in favor of security or reliability as they see fit. Different users will place different values on their account, and those values will change over time. Websites can periodically encourage users to check and configure authentication options.

Regularly Verify and Update Authentication Information

As users come to value and rely on a service, an increasing imbalance might grow between the value they place on the security and reliability of their account access and the quality of the authentication information they provided. The passage of time reduces the reliability of authentication configuration information, such as an email address or answer to a security question.

To bring the quality of authentication configuration information provided by the user to parity with the growing value users place on their account, the website can encourage users to update and augment their secondary authentication configuration over time. For example, a website might periodically remind users of their choice of alternate email address (for use in email-based secondary authentication) and ask if it's still valid. Websites might periodically ask users to reanswer security questions, and encourage them to choose new questions if they're unable to respond with the correct answer in a small number of guesses.

Adjust Security Requirements Based on Account Activity

Users who just changed their password, or who haven't logged in to their account for some time, are more likely to forget their password than users who log in regularly with the same password. A website might want to relax the amount of proof required to authenticate the longer an account goes unused. It might also want to relax the secondary authentication requirements in other circumstances, such as when users have just changed their password at the website's request—in this case, they might be likely to forget their new password until they've used it a few times.

In other cases, a password change might indicate that the account is at risk. For example, users might change their password after believing the account was compromised, or after the souring of a relationship with someone with whom they had shared the password. If an account has competing claims, security requirements should become more stringent.

Design under the Assumption that Accounts Will Be Compromised

No matter how well a secondary authentication system is designed, some impostors will succeed in authenticating. An authentication system should be designed to limit the damage resulting from account compromise.

If attackers want to permanently capture an account, the first thing they might do is change the password and authentication options. One way to limit the efficacy of such an attack is to delay the impact of any changes, remind users that changes have been made each time they log in, and allow the changes to be rolled back. However, some users with legitimate security concerns will want this information changed immediately. Some websites might allow such changes—for example, if the user meets a higher threshold of authentication to authorize the change.

Regardless, situations will still arise in which attackers can provide as much evidence that they are the account holder as the account holders themselves—that is, attackers can pass just as many secondary authentication mechanisms as account holders can. This is especially likely to be the case for websites that usually consider a single authentication mechanism sufficient evidence of account ownership.

If two competing parties can both successfully authenticate, an escalation process is necessary. If security is more important than reliability, the account provider might decide to prevent anyone from logging in to the account during the escalation period.

Because the provider won't have any way to ver-

ify the true owner's identity, the escalation process relies on the assumption that attackers will place less value on the account and, failing that, will be less likely to want to risk appearing in front of an authority to make an ownership claim. A first step might be to require all parties claiming account ownership to contact the customer support team to provide their contact information. A fraction of impostors will be reluctant to do so. The provider might then ask for a donation to a charity of the account holder's choice, in an amount that would deter attackers without imposing too much of a cost on genuine account holders (for example, US$10). The value of the required donation would need to exceed the value placed on a single account by attackers who compromise accounts en masse, yet not be unreasonable for the account holder. Finally, in-person proofing that requires a visit to a location where an attacker might fear arrest should deter all but the most fearless attacker. For example, the service could require those claiming to own an account to obtain a police report about the account theft, the authenticity of which can be verified. This escalation process should deter attackers from starting down this path and keeps this expensive process sufficiently rare enough to be sustainable.

A variety of secondary authentication mechanisms and configurations of those mechanisms exists; each strikes a different balance among the desirable criteria of reliability, security, and efficiency. For most websites, fairly simple secondary authentication mechanisms, such as email-based authentication, are usually adequate. However, websites whose accounts have greater value, such as webmail services, should put careful thought into designing secondary authentication systems. We recommend combining multiple mechanisms, enabling the system to adapt to meet the account holder's security and reliability requirements. We also recommend regularly refreshing authentication information, encouraging users to adjust authentication options on the basis of account activity, and designing with compromised accounts in mind. □

References

1. N. Hines, "Sarah Palin's Private E-Mail Account Accessed by Hacking Group Anonymous," timesonline.co.uk, 18 Sept. 2008; www.timesonline.co.uk/tol/news/world/us_and_americas/us_elections/article4780133.ece.
2. N. Cubrilovic, "The Anatomy of the Twitter Attack," TechCrunch, 19 July 2009; http://techcrunch.com/2009/07/19/the-anatomy-of-the-twitter-attack.
3. M. Just, "Designing Authentication Systems with Challenge Questions," *Security and Usability*, L.F. Cranor and S. Garfinkel, eds., O'Reilly, 2005, pp. 143–155.
4. M. Zviran and W.J. Haga, "User Authentication by Cognitive Passwords: An Empirical Assessment," *Proc. 5th Jerusalem Conf. Information Technology* (JCIT 90), IEEE CS Press, 1990, pp. 137–144.
5. S. Schechter, A.J. Bernheim Brush, and S. Egelman, "It's No Secret: Measuring the Security and Reliability of Authentication via 'Secret' Questions," *Proc. 2009 IEEE Symp. Security and Privacy*, IEEE CS Press, 2009, pp. 375–390.
6. A. Rabkin, "Personal Knowledge Questions for Fallback Authentication: Security Questions in the Era of Facebook," *Proc. 4th Symp. Usable Privacy and Security* (SOUPS 08), ACM Press, 2008, pp. 13–23.
7. M. Just and D. Aspinall, "Personal Choice and Challenge Questions: A Security and Usability Assessment," *Proc. 5th Symp. Usable Privacy and Security* (SOUPS 09), ACM Press, 2009.
8. M. Jakobsson et al., "Love and Authentication," *Proc. 26th Ann. SIGCHI Conf. Human Factors in Computing Systems* (CHI 08), ACM Press, 2008, pp. 197–200.
9. S.L. Garfinkel, "Email-Based Identification and Authentication: An Alternative to PKI?" *IEEE Security & Privacy*, vol. 1, no. 6, 2003, pp. 20–26.
10. C.K. Karlof, "Human Factors in Web Authentication," PhD thesis, Electrical Engineering and Computer Sciences Department, Univ. of California, Berkeley, 6 Feb. 2009.
11. J. Brainard et al., "Fourth-Factor Authentication: Somebody You Know," *Proc. 13th ACM Conf. Computer and Comm. Security* (CCS 06), ACM Press, 2006, pp. 168–178.
12. S. Schechter, S. Egelman, and R.W. Reeder, "It's Not What You Know, but Who You Know: A Social Approach to Last-Resort Authentication," *Proc. 27th Ann. SIGCHI Conf. Human Factors in Computing Systems* (CHI 09), ACM Press, 2009, pp. 1983–1992.
13. S. Schechter and R.W. Reeder, "1 + 1 = You: Measuring the Comprehensibility of Metaphors for Configuring Backup Authentication," *Proc. 5th Symp. Usable Privacy and Security* (SOUPS 09), ACM Press, 2009.

Robert W. Reeder *is a senior security program manager on Microsoft's Usable Security team, where he conducts research and educates engineers on how to better design security-related user experiences. Reeder has a PhD in computer science from Carnegie Mellon University. Contact him at roreeder@microsoft.com.*

Stuart Schechter *is a researcher at Microsoft Research. His research interests include computer security, human behavior, and computer architecture. Schechter has a PhD in computer science from Harvard's School of Engineering and Applied Sciences. Contact him at stus@microsoft.com.*