Password Recovery

Charles Miller <cmiller@pastiche.org>

20th October 2002

Copying

Contents

1 Introduction

Password recovery becomes necessary when the user of a system is no longer able to authenticate themselves because they have lost or forgotten their password. Any systems that require authentication will need to have some policy or procedure for password recovery.

The simplest policy is: "Password recovery will not be performed." When combined with a policy that locks inactive accounts, it is also the most secure approach. Users who lose their password are required to repeat the registration process and acquire a new identity. The user will lose their previous identity and any saved information, and the cost of annoying your more careless users must be offset against any benefits in security and simplicity.

If the "no recovery" policy is impossible, then there are two forms of recovery mechanism that may be implemented in some combination:

- Communicating a secret with the user over some pre-arranged secure channel, and having them use that to re-authenticate.
- Providing a secondary means of authentication for the purpose of recovering or resetting the first.

In a controlled environment, the second is the most popular. In a corporate environment, the user presents themselves to their manager or IT department, who verifies they are who they say they are, and resets their password. (Of course there's also the "secure channel" of how the IT department communicates the new password to the user, but that's trivial once the authentication has occurred.)

In such environments, password recovery for web applications becomes part of the broader account maintenance policies for the company. Web applications for which the users of the system are customers or clients, on the other hand, require separate techniques, which will be the focus of this paper.

2 General Advice

Good Practices:

- When recovering a password, always assign, or require the user to choose a new password.
  - You shouldn't be storing unhashed passwords anywhere anyway
  - If the recovery procedure is compromised, the real user will be unable to log in. This makes it easier to detect fraud.
- (obviously) Log all recovery attempts
- Only allow recovery once within a particular time-period (three months) Subsequent recovery attempts will result in a direction to call customer service, where a rep will manually perform the recovery procedure.
  - When the user calls, the rep can guarantee that the previous recovery attempt was genuine.
  - It's an opportunity to educate the user.

Things to Consider:

- Is the process automated, or does it require human intervention?
- If the process requires human contact, are you ready to deal with users from other countries, or who do not speak the same language as you?
- To what degree can you inconvenience your users before they decide not to bother? Is it a problem if they do?
- What steps can you take to make sure the users don't forget their passwords in the first place?

3 Specific Techniques

3.1 In Person Identification

Obviously, the best thing to do is to have the user physically present themselves to you with 100 points worth of photographic identification. This is a form of secondary-authentication recovery, and is very secure. Like all optimal solutions, this is generally impossible in a web environment, and is a particular inconvenience to users.

That said, if you need security, and you have the luxury of physical offices near the vast majority of your customers (I'm thinking of banks and government departments here), this is worth thinking about.

Advantages

- Best, legally defensible security.
- In case of fraud, you might remember what they look like

Disadvantages

- Requires human intervention in all cases.
- Requires the user to take time out (during business hours) to recover their password.
- May be difficult or impossible for the user dependent on their distance from your offices.

3.2 Faxed Documentation

Another form of secondary authentication, the user does not present themselves, but transmits a facsimile of some kind of official identification such as a passport or drivers license. This is a quite common technique, but may not be as reliable as it seems.

Good Practices

- Keep a copy of the identification on file from signup-time, to be compared in the event of recovery.
  - ID documents look different from state to state and country to country. Unless you have some idea what it *should* look like, you can easily be fooled by a competent graphic artist.

Advantages

- So long as the identification is on file, this is reasonably secure, as an attacker must be a competent forger, and know the correct serial numbers for the documents. If not, this is easily defeated by a committed attacker.

Disadvantages

- If the identification papers are on file with you, they may be on file with many other companies. Similarly, if the user is willing to fax you their passport for identification purposes, they may be socially engineered into sending the same documentation to an attacker.
- Requires human intervention in all cases.
- User must have access to a fax machine.
- User must be willing to store their documentation with you (the user may quite justifiably fear identity theft if your records are compromised or misused)
- Digital forgery is possible, and is masked by the poor quality of fax transmissions.
- Identity documents are replaced or renewed, so some secure mechanism for replacing the filed ID is necessary.
- Does not cover how to get the new password to the user after they've identified themselves over the fax.

3.3 Simple Email recovery

E-mailing the password to the user's registered address is the most common form of recovery on public websites. It is a weak form of "secure channel" recovery, where it is assumed the path between the server and the user's email client is secure. Of course, this assumption isn't particularly defensible, but the process requires almost no human intervention, and is "secure enough" for most applications.

Good practices

- Send the user a one-time password, based on the user's password hash and a timestamp. That way, the recovery password can only be used until the main password is changed, and can be expired if not used for 24 hours.
  - The email does not become a security problem if it's archived.

Advantages

- Highly automated
- Familiar, and very easy for the user to understand
- Attacks rely on compromising some part of the communications chain between server and user. Thus, targetting an attack against a particular individual requires some effort.

Disadvantages

- Vulnerable at every point that the email could be intercepted
- Users may not understand that by sharing their email account with family members, they are making passwords available o susceptible to opportunistic attacks
  - once an attacker has access to any mailserver, they can methodically break the accounts of all users of that server.
- If the mailserver or gateway from which the recovery mails originate is compromised, it's game over for everybody.

3.4 Encrypted Email Recovery

A secure channel method, sending an email encrypted with some secret only known to the customer is possible. The obvious method is to have the user provide some public key at registration time, and send their recovery emails encrypted with that key.

Good Practices

- If you are going to provide this, it must be optional.
  - Most people do not have public keys.
  - If encrypted email recovery is selected by the user, disable alternative methods of recovery
- All practices for good email recovery still apply here.
- Remember to provide mechanisms to cater for key expiration and revocation.

Advantages

- Highly secure

Disadvantages

- Not a good general solution, the PGP-using population of the world is infinitesimal.
- Key management remains one of those issues that always looks easy on paper, but tends to be much harder to implement.

3.5 Question and Answer

Another form of secondary authentication. When registering the account, the user records some personal information. To recover the password, the user must answer questions based on this personal information, either to a web page, or over the phone to a representative.

The questions and answers become a shared secret, a form of secondary password. Remember how we warn people not to base their passwords on their names, pets names or dates of birth? The security of this technique varies based on the choice of questions: the ease of an attacker guessing the questions, and the ease of researching the answers. It's amazing how many institutions believe that your date of birth and your mother's maiden name are sufficiently obscure to protect your bank account.

One implementation would be to request the information that was supplied during registration, such as address, phone number and credit-card details. This sort of information may, however, be quite easy for an attacker to come across or socially engineer.

Properly managed, this technique is more secure than e-mailing the password in the case of generalised or opportunistic attacks. However, due to its susceptibility to research, it is less secure against targetted attacks.

Good Practices

- Have a large pool of questions, of which the user only has to answer a subset during signup/recovery.
  - The more questions in the pool, the more research an attacker must do.
  - On the other hand, having a set of 50 personal questions in the sign-up process will scare people away.
- Some systems allow the user to choose the questions as well. This is a bad idea, as users don't understand security, and will either make things too easy for an attacker to guess, or too hard for themselves to work out what the hell they were thinking, six months hence.
- If the process is not automated, and for some insane reason you have the original password on file, "What did you think the password was?" can be an effective question.

Advantages

- Less susceptible to opportunistic attacks than mailed passwords.
- Still able to be automated, or performed by untrained employees following scripts and exercising next to no personal judgement.

Disadvantages

- It's hard to come up with a good set of questions.
- The better the attacker knows the target, the less secure it is. Unless the questions got *very* personal, my brother could probably answer 90% of them for me.
- Users may consider personal questions to be an intrusion.
- The answers may not be easy to match automatically, leading to false rejections.

3.6 Callback

Another secure channel method. The user makes a request for a new password, and the recovery secret is sent to a phone or pager number supplied by the user during registration.

This method can be combined with "Question and Answer" for a pretty effective recovery system. It mostly limits attacks to friends and family (who have access to the telephone and the personal information), and since the time of the call is logged and the attacker has to talk to the representative, it's easier to trace fraud.

Proprietary systems exist that automate this process, adding secondary authentication via voice recognition.

Advantages

- When combined with Question and Answer, this is probably the most secure method after encrypted email.

Disadvantages

- Company bears the cost of phone calls, which might be difficult if you have international users.
- Resource-intensive if done manually.

4 Acknowledgements

This document contains much input from a thread on the webappsec mailing list, contributed to by: Brecrost Jones <brecrost at hotmail dot com>, David Bullock <davidbullock at tech-center dot com>, Mark Curphey <mark at curphey dot com>, Kevin Spett <kspett at spidynamics.com>, Haroon Meer <haroon at sensepost dot com>, Jeroen

Latour <jlatour at calaquendi dot net>, Chris Shepherd <chriss at whstuart dot com>, Bill Smith <wsmith at icsalabs dot com>, Sverre Huseby <shh at thathost dot com> and Matthew Chalmers <Matthew_Chalmers at bankone dot com>.

5 Revision History

- 2002-10-19 original email sent to webappsec@securityfocus.com
- 2002-10-20 incorporated suggestions from Jeroen Latour into the faxed documentation section, from Sverre Huseby into the encrypted email recovery section, and from Matthew Chalmers into the callback section. First version to be released under the GNU FDL.