

Jonathan Buhacoff's Reflections on Chapter 15 of Practical Cryptography

Chapter 15 of Practical Cryptography is about key negotiation protocols. The authors Bruce Schneier and Niels Ferguson describe their key negotiation protocol by introducing something very close to the Diffie-Helman key exchange protocol from a previous chapter, then identifying weaknesses and fixing them until they arrive at their final recommended protocol.

The main ideas in this chapter are:

1. Everything should be authenticated, because it's not safe to assume that you know all the ways that an unauthenticated parameter can be used against you
2. Use random values to prevent replay attacks by proving that the other party is "live"
3. Make protocols as short as possible but no shorter; don't take any shortcuts or make optimizations that compromise what needs to be trusted; don't take 4 steps to do what can be done in 3
4. Don't set hard limits because they will cause interoperability problems in the future
5. Hide algebraic relations from your attackers by hashing values that may reveal them accidentally
6. Check all the information you get from other parties to make sure it's valid; if it's a size parameter it should fall inside a range, or if it's a selector it should have one of a predefined set of values
7. Protocols should be modular; every component should be designed to work with minimal assumptions about the robustness or security of the other components

In 2005, I had an idea for a protocol that would allow users to login to websites without a password. The idea is that most websites request visitors to choose a username and password so they can be authenticated on future visits. Hardly any website at all actually verifies anything about the username and password, or the other profile details a user may enter. They're arbitrary. So I thought, why not replace the username and password with an automatically generated certificate? The user's browser doesn't even have to include any personally identifying information in the certificate, because it doesn't matter. All that matters is that the user can prove to the website on a later visit that it is the same user who created the account.

Now, since the user is using a web browser, it's easy for the web browser to keep track of website certificates and to know which automatically generated client certificate to send to which website. Typically it would be a 1-to-1 relation and this can be completely automatic. If a user wants to create multiple profiles on a site, then the browser prompts the user to label the first and subsequent certificates so they can be selected from a menu. To keep things secure, the browser would just have to ask the user for ONE pass phrase, whenever it opens (or periodically) which would unlock the certificate storage. Voila, a password-free web. Less annoying, and less vulnerable to password attacks since the password is only used locally with a user interface and can't be attacked over a network.

The certificate storage can actually be on a server to allow user mobility & defense against theft, and in that case each computer in use would be responsible for securing just one private key that is allowed access to the user's centralized certificate storage repository. The user can have the same password or different passwords for each device he uses, his choice. This has the added advantage that if a single device is compromised its private key can be revoked by the central server.

I'm working on an authentication system that I call PassFree, which incorporates these ideas.

For more information about improved electronic access control using PassFree, contact:

Cheyenne Software, Inc.
(800) 935-9637
someone@mypassfree.com