Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: April 25, 2013

P. Hallam-Baker
Comodo Group Inc.

October 22, 2012

# HTTP Authentication Considerations
## draft-hallambaker-httpauth-01

Abstract

This draft is input to the HTTP Working Group discussion of HTTP authentication schemes.

Since the topic is one that the intended audience is more than familiar with, the presentation style is maybe not what is usual in such papers.

Status of this Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on April 25, 2013.

Table of Contents

## 1. What is Wrong in Web Authentication

What is wrong with Web Authentication is that twenty years later we still depend on passwords and what is worse, the weakest possible password infrastructure.

Little has changed in the use of password authentication since the publication of crack in 1990. Before crack it was a point of pride for many Unix admins that the one way encryption used in UNIX "made their password system more secure than VMS". It was argued that making the one way encrypted password file public made the system more secure than the 'security-through-insecurity' approach of VMS. VMS also used one way encryption but the password file was only readable with admin privileges. Endless USENET flames explained 'why' this was a terrible, terrible idea. When crack appeared these flames were quietly forgotten and it is widely imagined that UNIX systems have always had shadow passwords.

In the wake of crack it was discovered that most users chose terrible passwords that could be guessed through a brute force or dictionary attack in a few hours. The use of a salt made guessing passwords moderately more difficult as did minimum length password requirements and a requirement to use a non alphabetic character.

In the two decades since the standard rules for passwords were set in scripture, computing power has increased by over a billion times and it is now possible to buy a computer that will fit in the heel of a shoe that is more powerful than the fastest mainframe on CERN campus in 1992. The ad-hoc rules developed to make brute force attacks an order of magnitude harder in 1990 have been rendered utterly irrelevant by the six orders of magnitude improvement in available computing power. We have arrived at a situation where we use passwords that are maximally hard for people to remember while being trivial for modern computers to break by brute force.

This approach to password security: complacency followed by ad hoc patches followed by complacency has remained firmly in place since.

### 1.1. Password Promiscuity

Today the typical Internet user has to remember hundreds of passwords and account names. Since this is of couse silly, most users, myself included do no such thing. Most users have the same password for every account. Security concious users have a different password for every account they care about and an easy to remember password for the rest. My New York Times account is phil@hallmabaker.com and the password is GuessEasy. Go have a party. That information protects an asset that the New York Times wants to protect. It is not an

asset that I care to spend time or effort protecting.

Password promiscuity is a natural strategy that users have adopted to protect the asset they care about: Their time. Creating and remembering strong passwords takes time and effort. Blaming users for not taking time and effort to protect the assets of other people is futile.

If Alice has shared her password at 100 sites then a single corrupt actor who can access just one of those sites can gain access to the other 99.

Account names pose a harder problem since most sites require that the account name be unique for that site. So a user who finds their preferred account name is taken has to choose another.

Expecting users to remember all this stuff is stupid, just stupid.

### 1.1.1. Password Recovery Schemes

Password and account name recovery schemes are necessary because users cannot remember the hundreds of passwords or account names that using the Web now involves.

The most common password recovery mechanism is the email callback authentication mechanism first used by Mallery and Hurwitz on their Open Meeting system. A challenge (usually in the form of a link) is sent out to the user to their registered email address. The user must respond to the challenge to reset their account.

One important consequence of the email callback scheme is that virtually every Web site that has an accounts mechanism requests an email address during registration. There is thus no loss of privacy if the email address is used as the account name. Many sites have realized this fact and avoid the need for the user to choose an account name by allowing them to give their email address instead.

A site need not and in fact should not disclose the email address to other users when this approach is used. Shadow account names allow the user the courtesy of not having to remember the account name for that site.

### 1.1.2. Phishing

Another circumstance that made it difficult to implement strong security mechanisms at the time was the popular misconception that the attackers were exclusively teenage miscreants engaged in the cyber equivalent of joy-riding rather than criminals motivated by

money. People were warned that this was not the case but they did not want to hear.

The core defect of the Web authentication mechanism is that passwords are presented en-clair to the verifier. Thus any party who can present a login page to the user stands a good chance of capturing their credentials.

This problem was of course anticipated a few days after the BASIC authentication mechanism was proposed and was the original motivation for DIGEST authentication. But DIGEST authentication did not permit the re-use of legacy UNIX password files and so implementation did not take place until it was to late to deprecate BASIC.

Even today, IE7 makes no distinction between a request for authentication using BASIC vs DIGEST. Thus an attacker can easily capture the user's credentials through a downgrade attack. DIGEST was intended to replace BASIC and for BASIC to be expunged from the spec as dangerous to use.

In the event authentication moved into the HTML layer which likewise communicated the password enclair to the verifier. Thus enabling phishing.

## 1.2. Provider Lock In

Many schemes have been developed to provide the user with a portable form of credential but all those created to date present the security risk of lock-in to both users and relying parties.

Such schemes are often described as 'identity management' a term that seems to hurt rather than help comprehension of the problems involved.

Many users have found to their cost that when their FaceBook or Twitter account is disabled, they lose access to every other Web site linked to it.

While end users are faced with a minor inconvenience, relying party sites are faced with the risk that the 'identity provider' will decide to change their terms of service to their great disadvantage with little or no notice. Essentially the relying parties have agreed to channel all their traffic through the portal of another without any form of contractual agreement to prevent the portal owner setting up a toll booth.

A particular form of lock in that will doom any scheme to an inevitable and deserved death is any attempt to tie the scheme to the

use of any naming registry other than the DNS.

In one recent exercise in mindboggling futility a group of otherwise rational people spent over five years on a project where the two identifier forms to be supported were http://www.whowoulddothis.org/username and =username. It was rather obviously and abundantly clear that the only rational reason for the first choice was to corrale users to the inevitable choice of the second. Well they didn't did they?

One of the problems with such commercial interests is that it is often considered impolite or somehow improper to point out that they exist. Even when it is rather clear that their presence is going to doom the scheme to extinction.

While a naming ragistry can be profitable in theory, there have only been four such registries established on an international scale in the history of human civilization. Each supported a new form of communication, these being the postal system, the telephone system, the barcode product identifier system and the Internet. While commercial control of such a registry would of course bring ritches almost beyond the dreams of MBAs, the chance that such proposals might succeed is negligible to nil.

### 1.3. Strong Credentials Compromised by Weak Binding

A secondary Web authentication problem is that use of strong credentials is compromised by the inadequacy of the Web protocols. For example, a One Time Password (OTP) token provides strong authentication when used in the context of a VPN or other application that can guarantee that the passcode is only presented to the intended service. When entered into a HTML page, OTP passcodes are as vulnerable to interception as passwords.

Use of an OTP or public key token provides strong evidence that the hardware device concerned was in some way involved in a transaction but not the intention of the token holder.

Consider the case where a wire transfer of $10,000 to be sent to Nigeria is requested, the bank asks for an OTP value to be entered as confirmation. The bank can only verify that the value entered is the next OTP value in sequence. The bank has no means to determine whether the token holder was looking at a Web page that asked them to enter the value to confrim the wire transfer or whether they were looking at a Web page asking if they would like to buy a fluffy pink bunny for their daughter's birthday.

If an authentication system cannot tell the difference between a con

trick and a pink fluffy bunny, it should not be considered strong.

### 1.3.1. Confirmation vs Authentication

Modern smartphones are ubiquitous and relatively cheap. They provide a computing capability, a communication capability and a display in a single package. This is a platform that can and should be leveraged as an authentication device that can provide proof not only that the user was involved but that they were committed to the transaction concerned.

This technology provides us for the first time with a technology platform that is capable of presenting the user with the actual transaction they are being asked to confirm and to thus provide a strong binding between the device and the transaction.

My prefered hardware device for such a use would be a wristwatch with an intelligent display.

## 1.4. What passwords get right

Overlooked in most security discussion of passwords is what they get right, indeed it is often assumed that they have no redeeming features. Yet if that were the case they would have been gone long ago. The real problem of passwords is that they work just well enough to be almost always better than the alternatives.

The biggest advantage of passwords is that every Internet device has had to develop the affordances to support them. My Internet enabled weighing scale has a USB socket whose sole purpose is to enable me to plug it into a computer to set the WiFi network name and password. I can log into my personal email account from every desktop computer, laptop, tablet or mobile in the house. I can only acces my corporate email from the one machine configured for the corporate VPN and smart token.

Passwords require no dongles, tokens or cards which in turn means that they don't require device drivers or administrator privileges. While vendors of such systems strive to present low barriers for such devices, low barriers can never compete with no barriers if the user has a choice in the matter. People take effort to secure the assets they care about which is why banks can't give authentication tokens to customers while the same customers will go and pay for a battle.net token to secure the assets that matter to them.

## 2. User Authentication is Three Separate Problems

The term 'authentication' tends to cause confusion due to the fact that there are actually three separate activities that it can refer to. When Alice establishes an account at a Web site, the site may verify her email address is correct. When Alice presents her username and password, the site verifies the data and if correct issues a HTTP cookie. When Alice re-visits the same Web site to make further requests, the cookie is verified each time. Each of these verifications is a form of authentication but they are totally different in character and only the last of these is a form of authentication that is directly related to HTTP.

Attempts have been made to distinguish between these as 'initial authentication' and 're-authentication' but this also creates confusion as some people consider the first contact with the user as the 'initial' authentication and others consider that to be the start of a Web session.

### 2.1. Registration

Registration comprises all activities related to the establishment and maintenance of an account. These include the initial dialogu in which Alice picks out an account name for display on the site and her authentication credentials and all subsequent updates including password resets.

In a small number of circumstances, registration involves authentication of dopcumentary evidence such as articles of incorporation, a passport, business license or professional qualifications. The term validation seems to have emerged as the term of art for this activity.

Today registration is almost exclusively managed through HTML forms. Any new system will probably have to respect this approach. Registration establishes an account that is in almost every case to be accessible from multiple Web browsers rather than just the browser on which the registration process was completed.

### 2.2. Credential Presentation

Unlike the FTP and Telnet protocols that preceded it, HTTP Web sessions typically span multiple TCP connections. In the typical case the use will 'log in' to a Web site to establish a session that then continues for several hours, days or even months.

Like the registration phase, credential presentation has largely transitioned out of the browser to HTML. Although many users employ

plugins and applets that effectively reverse this, filling in the account an password field from a password database that is stored locally or in the cloud.

While such 'password managers' have traditionally been considered part of the problem they are in fact a necessary part of any solution. If we are going to improve matters we must first offer users a solution that meets their needs better than current solutions.

## 2.3. Message Authentication

Credential presentation has a necessary impact on the user. Since it would be tedious to enter a username and password for every Web page view, a lightweight mechanism is necessary to re-authenticate the user and demonstrate that they are the user that authenticated themselves when the session was created.

This is the only part of the process that is currently within the scope of HTTP rather than HTML and probably the only part for which it will be possible to get agreement on a better mechanism than the existing one.

The best available mechanism is the HTTP 'cookie'. This is highly unsatisfactory as a technical mechanism as they are weakly bound to the HTTP transaction and disclosed to the verifier. These circumstances in turn lead to numerous forms of cookie-stealling and cookie-stuffing attacks.

Using cookies for authentication also involves privacy concerns. In the context of authentication schemes, the use of cookies does not raise new privacy concerns as the purpose of the authentication scheme is to establish a session and uniquely identify the user. Unfortunately the cookie spec permits sites to establish cookies for tracking purposes without user permission or knowledge. The fact that cookies may be used for illegitimate purposes compromises legitimate uses and creates unnecessary transaction costs including customer serivce calls, congressional subpoenas and accusations on slashdot.

My proposal for fixing this part of the problem isdescribed in [I-D.hallambaker-httpintegrity].

While there are many, many concerns that need to be considered in the registration and credential presentation operations, the problem of message authentication can be reduced to the problems of:

Identifying a security context consisting of at minimum an authentication algorithm, key and account identifier.

Applying the security context to parts of the HTTP message.

## 3. Deployment Approach

Proposing a better authentication mechanism for the Web is easy. In fact there is nobody involved in Web Security that could not develop a better scheme given a free hand and a group of people interested in deployment. The development of a secure scheme should be well within the capabilities of a competent first year Computer Science undergraduate.

The much harder problem is deployment and in particular how to deploy a new scheme in the context of the legacy infrastructure. Here the ease with which a proposal can be made makes deployment harder, not easier since there are always competing proposals.

### 3.1. Password Managers as Transition Path

If the users are going to participate in any new scheme it must work at least as well as the existing password scheme. In particular it must be possible for the user to access all their accounts from their browser in a transparent fashion with minimal hassle.

Storing passwords in the cloud is a very good way to achieve the user's interest even if the sites whose assets may be compromised as a result may see it as a bad thing. Let us agree for the sake of argument that we consider passwords to be such an intrinsically insecure form of authentication that the user choice to store them in the cloud has negligible impact on the security of the system.

Since all the proposals to improve the Web authentication infrastructure involve some form of 'identity provider', why not give that provider the secondary purpose as a password manager?

If the communication between the client and the password manager was a widely supported Internet standard, users could choose any provider they liked as their password manager and access their credentials from any Web browser that they might need to use.

Such a confluence could in itself improve security as once the user is assured that every browser they need to use has access to their password manager, there is no need for the password to be memorable. It is thus possible for the password used for credential presentation to the site to be long and strong.

Use of a standards based password management protocol permits the user to take their security into their own hands irrespective of what attempts the sites might attempt to prevent it. What is perhaps more interesting is that it also sets the scene to enabling use of strong credentials.

### 3.2. Non-Transferable Credentials

While Web sites concerned with the risk that their customers store credentials in the cloud might attempt to frustrate a cloud based password infrastructure, a much better approach is to co-opt it and use it as a basis to build on. At the very least, a cloud based authentication infrastructure provides a useful pre-authentication step that could be used as a preliminary to a site specific log-in.

But just as an identity provider can also be a cloud password manager, the converse is also true. The cloud password manager can also support one or more existing mechanisms that enable credential presentation authentication without disclosure of the credential being verified. These could be based on SAML, OpenID, OAUTH or even some new proposal.

## 4. Action Plan

My proposal to improve Web authentication has the following parts:

Open Protocol for credential acquisition

HTTP Integrity Header

Bindings for credential presentation employing the commonly used federated authentication schemes, vis SAML, OAUTH, OpenID, etc.

### 4.1. Open Protocol for credential management

This protocol would be responsible for managing users legacy credentials (i.e. passwords) and to support whatever strong mechanisms for credential exchange were developed.

The protocol itself would require registration, credential presentation and query components. Once the user had established an account they would have to be able to bind any browser or device of their choice to that account, possibly doing so for a very limited time or for limited sites.

The query component would be of the form 'how do I access site X with identity Y'. For general applications the broker might then respond

with a username and password or a secret to be used to respond to a challenge. For circumstances requiring higher security the system might support some form of key splitting or sharing or other control limiting exposure.

### 4.2. HTTP Integrity Header

This is described at length in [I-D.hallambaker-httpintegrity]

The biggest challenge in the design of SAML, OAUTH and OpenID was establishing a secure binding between the authentication token and the HTTP messages. Using cookies and URI fragments for this purpose is deeply unsatisfactory.

### 4.3. Bindings

While we might imagine that we can get by with one single protocol that is going to solve all our authentication problems we now live in a world where there is much legacy that demands attention. Even if we decide that in future we are all going to use one single new protocol, we have to find a mechanism that allows the credential management systems to trade in their old lamps for new.

At minimum we require a mechanism that allows Web sites to specify which forms of authentication credentials they might accept, which mechanisms for useing them for validation and of transporting the challenges and responses for such mechanisms within HTTP message headers.

## 5. Security Considerations

Since this is a discussion document and the whole thing is talking about security, I think I will just leave the headings here to remind people about the security issues I think people should consider.

### 5.1. User Lock In

### 5.2. Site Lock In

### 5.3. Impersonation

### 5.4. Credential Disclosure

### 5.5. Credential Oracle

### 5.6. Randomness of Secret Key

## 6. IANA Considerations

## 7. Normative References

[I-D.hallambaker-httpintegrity]
Hallam-Baker, P., "HTTP Integrity Header", draft-hallambaker-httpintegrity-01 (work in progress), October 2012.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

## Author's Address

Phillip Hallam-Baker
Comodo Group Inc.

Email: philliph@comodo.com