Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on September 5, 2007.

Copyright Notice

Copyright (C) The IETF Trust (2007).
Abstract

This memo proposes requirements for protocols between web identity providers and users and for requirements for protocols between identity providers and relying parties. These requirements minimize the likelihood that criminals will be able to gain the credentials necessary to impersonate a user or be able to fraudulently convince users to disclose personal information. To meet these requirements browsers must change. Websites must never receive information such as passwords that can be used to impersonate the user to third parties. Browsers should perform mutual authentication and flag situations when the target website is not authorized to accept the identity being offered as this is a strong indication of fraud.

Table of Contents

1. Introduction .................................................. 3
2. Requirements notation ........................................ 5
3. Threat Model .................................................. 6
   3.1. Capabilities of Attackers .............................. 6
   3.2. Attacks of Interest ................................. 7
4. Requirements for Preventing Phishing ....................... 8
   4.1. Support for Passwords ................................. 8
   4.2. Trusted UI ............................................. 8
   4.3. No Password Equivelents ............................. 9
   4.4. Mutual Authentication ............................... 10
   4.5. Authentication Tied to Resulting Page .............. 10
   4.6. Restricted Identity Providers ....................... 11
   4.7. Protecting Enrollment .............................. 11
5. Is it the right Server? ..................................... 13
6. Iana Considerations ......................................... 14
7. Acknowledgments .............................................. 15
8. Security Considerations .................................... 16
9. References .................................................... 18
   9.1. Normative References ................................ 18
   9.2. Informative References ............................... 18
Appendix A. Change History .................................... 19
   A.1. Changes since 02 .................................... 19
   A.2. Changes since 01 .................................... 19
Author’s Address ............................................... 20
Intellectual Property and Copyright Statements ............ 21
1. Introduction

Typically, web sites ask users to send a user name and password in order to log in and authenticate their identity to the website. The user name and plaintext password is often sent over a TLS [RFC4346] encrypted connection. As a result of this, the server learns the password and can pretend to be the user to any other system where the user has used the same password. The security of passwords over TLS depends on making sure that the password is sent to the right, trusted server. TLS implementations typically confirm that the name entered by the user in the URL corresponds to the certificate as described in [RFC2818].

One serious security threat on the web today is phishing. Phishing is a form of fraud where an attacker convinces a user to provide confidential information to the attacker believing they are providing the information to a party they trust with that information. For example, an email claiming to be from a user’s bank may direct the user to go to a website and verify account information. The attacker captures the user name and password and potentially other sensitive information. Domain names that look like target websites, links in email, and many other factors contribute to phishers’ ability to convince users to trust them.

It is useful to distinguish two targets of phishing. Sometimes phishing is targeting web authentication credentials such as user name and password. Sometimes phishing is targeting other confidential information. This memo presents requirements that significantly reduce the effectiveness of the first category of phishing: these requirements guarantee that even if the user authenticates to the wrong server, that server cannot impersonate the user to a third party. However to combat phishing targeted at other confidential information the best we can do is try to help the user detect fraud before they release confidential information.

So, the approach taken by these requirements is to handle these two types of phishing differently. Users are given some trusted mechanism to determine whether they are typing their password into a secure browser component that will authenticate them to the web server or whether they are typing their password into a legacy mechanism that will send their password to the server. If the user types a password into the trusted browser component, they have strong assurances that their password has not been disclosed and that the page returned from the web server was generated by a party that either knows their password or who is authenticated by an identity provider who knows their password. The web server can then use confidential information known to the user and web server to enhance the user’s trust in its identity beyond what is available given the
social engineering attacks against TLS server authentication. If a user enters their password into the wrong server but discovers this before they give that server any other confidential information, then there exposure is very limited.
2. Requirements notation

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].
3. Threat Model

This section describes the assumed capabilities of phishers, describes assumptions about web security and describes what vulnerabilities exist.

We assume that the browser and operating system are secure and can be trusted by the end user. There are many attacks against browsers and operating systems. There are also many efforts to improve the security of browsers and operating systems. It is important to consider browser and operating system security when analyzing the security of the entire system. However in the interest of limiting our focus to one aspect of the problem, we assume that these efforts are successful. As a consequence of this assumption, these requirements are insufficient to provide protection against phishing if malicious browser extensions, trojan software or other malicious software is installed into a trusted part of the local computer.

We assume that users have limited motivation to combat phishing. Users cannot be expected to read the source of web pages, understand how DNS works well enough to look out for spoofed domains, or understand URI encoding. Users do not typically understand certificates and cannot make informed decisions about whether the subject name in a certificate corresponds to the entity they are attempting to communicate with. As a consequence of this assumption, users will likely be fooled by strings either in website names or certificates that look visually similar but that are composed of different code points.

3.1. Capabilities of Attackers

Attackers can convince the user to go to a website of their choosing. Since the attacker controls the web site and since the user chose to go to the website the TLS certificate will verify and the website will appear to be secure. The certificate will typically not be issued to the entity the user thinks they are communicating with, but as discussed above, the user will not notice this. Mechanisms attackers use to accomplish this include links where the link name and URI target are misleading; email with links; DNS attacks; and on-path network attacks.

The attacker can convincingly replicate any part of the UI of the website being spoofed. The attacker can also spoof trust markers such as the security lock, URL bar and other parts of the browser UI. There is one limitation to the attacker’s ability to replicate UI. The attacker cannot replicate a UI that depends on information the attacker does not know. For example, an attacker could generally replicate the UI of a banking site’s login page. However the
attacker probably could not replicate the account summary page until the attacker learned the user name and password because the attacker would not know what accounts to list or approximate balances that will look convincing to a user.

The attacker can convince the user to do anything with the phishing site that they would do with the real target site. As a consequence, if we want to avoid the user giving the attacker their password, we must transition to a solution where the user would not give the real target site their password. Instead they will need to cryptographically prove that they know their password without revealing it.

3.2. Attacks of Interest

The ultimate goal of these requirements is to provide protection against disclosure of confidential information to unintended parties. These requirements focus on two such disclosures and handle them separately. The first category is disclosure of credentials that could allow a unintended party to impersonate the user, possibly gaining access to additional confidential information. The second attack is disclosure of confidential information not directly related to authentication. The second class of attack cannot be directly defeated, but we can help users know when they are communicating with an unintended party.

Note that some authentication systems such as Kerberos [RFC4120] provide a facility to delegate the ability to act as the user to the target of the authentication. Such a facility when used with an inappropriately trusted target would be an instance of the first class of attack.

Of less serious concerns at the present time are attacks on data integrity where a phisher provides false or misleading information to a user.
4. Requirements for Preventing Phishing

This section describes requirements for web authentication solutions. These solutions are intended to prevent phishing targeted at obtaining web authentication credentials. These requirements will make it more difficult for phishers to target other confidential information.

The requirements discussed here are similar to the principles outlined in "Limits to Anti-Phishing" [ANTIPHISHING]. Most of this work was discovered independently but work from that paper has been integrated where appropriate. Google’s perspective on phishing is very interesting because of their operational experience.

4.1. Support for Passwords

The web authentication solution MUST support passwords and MUST be secure even when passwords are commonly used. In many environments, users need the ability to walk up to a computer they have never used before and log in to a website. Carrying a smart card or USB token significantly increases the deployment cost of the website and decreases user convenience. The smart card is costly to deploy because it requires a process for replacing smart cards, requires support staff to be trained in answering questions regarding smart cards and requires a smart card to be issued when an identity is issued. Smart cards are less convenient because users cannot gain access to protected resources without having their card physically with them. Many public access computers do not have smart cards available and do not provide access to USB ports; when they do they tend not to support smart cards. It is important not to underestimate the training costs (either in money or user frustration) of teaching people used to remembering a user name and password about a new security technology. Sites that aggregate identity—for example allowing a user to log into an identity provider and then gain access to other resources may be a significant part of a solution. However we cannot assume that a given user will have only one such website: there are valid and common reasons a user (or the relying party) would not trust all identity information to one such site.

IT is desirable that a solution support other forms of authentication such as smart cards and one-time passwords as these are useful in some environments.

4.2. Trusted UI

Users need the ability to trust components of the UI in order to know that the UI is being presented by a trusted component of the browser.
The primary concern is to make sure that the user knows the password is being given to trusted software rather than being filled into an HTML form element that will be sent to the server.

There are three basic approaches to establishing a trusted UI. The first is to use a dynamic UI based on a secret known by the user; the Google paper [ANTIPHISHING] recommends this approach. A second approach is to provide a UI action that highlights trusted or non-trusted components in some way. This could work similarly to the Expose feature in Apple’s OS X where a keystroke visually distinguishes structural elements of the UI. Of course such a mechanism would only be useful if users actually used it. Finally, the multi-level security community has extensive research in designing UIs to display classified, compartmentalized information. It is critical that these UIs be able to label information and that these labels not be spoofable.

See Section 5 for another case where confidential information in a UI can be used to build trust.

4.3. No Password Equivelents

A critical requirement is that when a user authenticates to a website, the website MUST NOT receive a strong password equivalent [IABAUTH]. A strong password equivalent is anything that would allow a phisher to authenticate as a user with a different identity provider. Weak password equivalents MAY only be sent when a new identity is being enrolled or a password is changed. A weak password equivalent allows a party to authenticate to a given identity provider as the user.

There are two implications of this requirement. First, strong cryptographic authentication protocol needs to be used instead of sending the password encrypted over TLS. The zero-knowledge class of password protocols such as those discussed in section 8 of the IAB authentication mechanisms document [IABAUTH] seem potentially useful in this case. Note that mechanisms in this space tend to have significant deployment problems because of intellectual property issues.

The second implication of this requirement is that if an authentication token is presented to a website, the website MUST NOT be able to modify the token to authenticate as the user to a third party. The party generating the token must bind it to either the website that will receive the token or to a key known only to the user. Binding could include cryptographic binding or mechanisms such as issuing a one-time password for use with a specific website. If tokens are bound to keys, the user MUST prove knowledge of this key.
as part of the authentication process. The key MUST not be disclosed to the server unless the token is bound to the server and the key is only used with that token.

4.4. Mutual Authentication

The Google paper [ANTIPHISHING] describes a requirement for mutual authentication. A common phishing practice is to accept a user name and password as part of an attempt to make the phishing site authentic. The real target is some other confidential information. The user name and password are captured, but are not verified. After the user name and password are entered, the phishing site collects other confidential information.

Authentication of the server at the TLS level and authentication of the client within the TLS session is sufficient to meet the requirement of mutual authentication. If authentication is based on a shared secret such as a password, then the authentication protocol MUST prove that the secret or a suitable verifier is known by both parties. Interestingly the existence of a shared secret will provide better protection that the right server is being contacted than if public key credentials are used. By their nature, public key credentials allow parties to be contacted without a prior security association. In protecting against phishing targeted at obtaining other confidential information, this may prove a liability. However, public key credentials provide strong protection against phishing targeted at obtaining authentication credentials because they are not vulnerable to dictionary attacks. Such dictionary attacks are a significant weakness of shared secrets such as passwords intended to be remembered by humans. For public key protocols, this requirement would mean that the server typically needs to sign an assertion of what identity it authenticated.

4.5. Authentication Tied to Resulting Page

Users expect that whatever party they authenticate to will be the party that generates the content they see. One possible phishing attack is to insert the phisher between the user and the real site as a man-in-the-middle. On today’s websites, the phisher typically gains the user’s user name and password. Even if the other requirements of this specification are met, the phisher could gain access to the user’s session on the target site. This attack is of particular concern to the banking industry. A man-in-the-middle may gain access to the session which may give the phisher confidential information or the ability to execute transactions on the user’s behalf.

The authentication system MUST guarantee to the user and the target
server that the response is generated by the target server and the contents of the response will only be seen by parties authorized by either the target server or the user. This can be done in several ways:

1. Assuming that only certificates from trusted CAs are accepted and the user has not bypassed certificate validation, it is sufficient to confirm that the identity of the server at the TLS level is the same at the HTTP authentication level. In the case of TLS client authentication this is trivially true.

2. Another alternative is to bind the authentication exchange to the channel created by the TLS session. The general concept behind channel binding is discussed in section 2.2.2 of [BTNS]. Work is ongoing to adapt this concept to HTTP over TLS [TLS-CB]

3. Redirect based schemes in which the identity provider is told what site to return the user to meets this requirement provided again that certificate validation is done at the TLS layer.

4.6. Restricted Identity Providers

Some identity providers will allow anyone to accept their identity. However particularly for financial institutions and large service providers it will be common that only authorized business partners will be able to accept the identity. The confirmation that the the relying party is such a business partner will often be a significant part of the value provided by the identity provider, so it is important that the protocol enable this. For such identities, the user MUST be assured that the target server is authorized by the identity provider to accept identities from that identity provider. Several mechanisms could be used to accomplish this:

1. The target server can provide a certificate issued by the identity provider as part of the authentication.

2. The identity provider can explicitly approve the identity. For example in a redirect-based scheme the identity provider knows the identity of the relying party before providing claims of identity to that party. A similar situation happens with Kerberos.

4.7. Protecting Enrollment

One area of particular vulnerability to phishing is enrollment of a new identity in an identity provider. Protecting against phishing targeted at obtaining other confidential information as a new service is established is outside the scope of this document. If
confidential information such as credit card numbers are provided as part of account setup, then this may be a target for phishing.

However there is one critical aspect in which enrollment impacts the security of authentication. During enrollment, a password is typically established for an account at an identity provider. The process of establishing a password MUST NOT provide a strong password equivalent to the identity provider. That is, the identity provider MUST NOT gain enough information to impersonate the user to a third party while establishing a password.
5. Is it the right Server?

In Section 4, requirements were presented for web authentication solutions to minimize the risk of phishing targeted at web access information. This section discusses in a non-normative manner various mechanisms for determining that the right server has been contacted. Authenticating to the right party is an important part of reducing the risk of phishing targeted at other confidential information.

Validation of the certificates used in TLS and verification that the name in the URI maps to these certificates can be useful. As discussed in Section 3, attackers can spoof the name in the URI. However the TLS checks do defeat some attacks. Also, as discussed in Section 4.5, TLS validation may be important to higher-level checks.

A variety of initiatives propose to assign trust to servers. This includes proposals to allow users to indicate certain servers are trusted based on information they enter. Also, proposals to allow third parties including parties established for this purpose and existing certificate authorities to indicate trust have been made. These proposals will almost certainly make phishing more difficult.

In the case where there is an existing relationship, these requirements provide a way that information about the relationship can be used to provide assurance that the right party has been contacted.

In Section 4.2, we discuss how a secret between the user and their local computer can be used to let the user know when a password will be handled securely. A similar mechanism can be used to help the user once they are authenticated to the website. The website can present information based on a secret shared between the user and website to convince the user that they have authenticated to the correct site. This depends critically on the requirements of Section 4.5 to guarantee that the phisher cannot obtain the secret. It is tempting to use this form of trusted UI before authentication. For example, a website could request a user name and then display information based on a secret for that user before accepting a password. The problem with this approach is that phishers can obtain this information, because it can be obtained without knowing the password. However if the secret is displayed after authentication then phishers could not obtain the secret. This is one of the many reasons why it is important to prevent phishing targeted at authentication credentials.
6. Iana Considerations

This document requests no action of IANA.
7. Acknowledgments

I’d like to thank Nicolas Williams, Matt Knopp and David Blumenthal for helping me walk through these requirements and make sure that if a solution met them it would actually protect against the real world attacks consumers of our technology are facing. I was particularly focusing on attacks that financial institutions are seeing and their help with this was greatly appreciated.

I’d like to thank Eric Rescorla and ben Laurie for their significant comments on this draft.
8. Security Considerations

This memo discusses the security of web authentication and how to minimize the risk of phishing in web authentication systems. This section discusses the security of the overall system and discusses how components of the system that are not directly within the scope of this document affect the security of web transactions. This section also discusses residual risks that remain even when the requirements proposed here are implemented.

The approach taken here is to separate the problem of phishing into phishing targeted at web authentication credentials and phishing targeted at other information. Users are given some trusted mechanism to determine whether they are typing their password into a secure browser component that will authenticate them to the web server or whether they are typing their password into a legacy mechanism that will send their password to the server. If the user types a password into the trusted browser component, they have strong assurances that their password has not been disclosed and that the page returned from the web server was generated by a party that either knows their password or who is authenticated by an identity provider who knows their password. The web server can then use confidential information known to the user and web server to enhance the user’s trust in its identity beyond what is available given the social engineering attacks against TLS server authentication. If a user enters their password into the wrong server but discovers this before they give that server any other confidential information, then there exposure is very limited.

This model assumes that the browser and operating system are a trusted component. As discussed in Section 3, there are numerous attacks against host security. Appropriate steps should be taken to minimize these risks. If the host security is compromised, the password can be captured as it is typed by the user.

This model assumes that users will only enter their passwords into trusted browser components. There are several potential problems with this assumption. First, users need to understand the UI distinction and know what it looks like when they are typing into a trusted component and what a legacy HTML form looks like. Users must care enough about the security of their passwords to only type them into trusted components. The browser must be designed in such a way that the server cannot create a UI component that appears to be a trusted component but is actually a legacy HTML form; Section 4.2 discusses this requirement.

IN addition, a significant risk that users will type their password into legacy HTML forms comes from the incremental deployment of any
web authentication technology. Websites will need a way to work with older web browsers that do not yet support mechanisms that meet these requirements. Not all websites will immediately adopt these mechanisms. Users will sometimes browse from computers that have mechanisms meeting these requirements and sometimes from older browsers. They only gain protection from phishing when they type passwords into trusted components. If a password is sometimes used with websites that meet these requirements and sometimes with legacy websites, and if the password is captured by a phisher targeting a legacy website, then that captured password can be used even on websites meeting these requirements. Similarly, if a user is tricked into using HTML forms when they should not, passwords can be exposed. Users can significantly reduce this risk by using different passwords for websites that use trusted browser authentication than for those that still use HTML forms.

The risk of dictionary attack is always a significant concern for password systems. Users can (but typically do not) minimize this risk by choosing long, hard to guess phrases for passwords. The risk of offline dictionary attack can be removed once a password is already established by using a zero-knowledge password protocol. The risk of online dictionary attack is always present. The risk of offline dictionary attack is always present when setting up a new password or changing a password. Minimizing the number of services that use the same password can minimize this risk. When zero-knowledge password protocols are used, being extra careful to make sure the right server is used when establishing a password can significantly reduce this risk.
9. References

9.1. Normative References


9.2. Informative References

[ ANTIPHISHING ]


[ BTNS ]

[ IABAUTH ]

[ RFC2818 ]

[ RFC4120 ]

[ RFC4346 ]

[ TLS-CB ]
Appendix A. Change History

A.1. Changes since 02

Updated discussion of TLS authentication to point out that it does meet the requirement of mutual authentication.

Added pointer to HTTP TLS channel bindings work

A.2. Changes since 01

Updated threat model to give examples of attacks that are in scope and to more clearly discuss host security based on comments from Chris Drake.

Clarify attacks of interest to be consistent with the introduction.

Fix ups regarding one-time passwords. I’m not sure that OTPs can meet all the requirements but clean things up where they clearly can meet a requirement.

Clarify that in the mutual authentication case I’m concerned about authentication of client to the server.

Clean up bugs in security considerations
Author’s Address

Sam Hartman
Massachusetts Institute of Technology

Email: hartmans-ietf@mit.edu