Wireless Authentication using Remote Passwords

Andrew Harding, Timothy W. van der Horst, Kent E. Seamons
Internet Security Research Lab
Brigham Young University
{hardinga, timv, seamons}@cs.byu.edu

ABSTRACT

Current wireless authentication mechanisms typically rely on inflexible shared secrets or a heavyweight public-key infrastructure with user-specific digital certificates and, as such, lack general support for environments with dynamic user bases where guest access is frequent. Simple Authentication for the Web (SAW) facilitates dynamic user bases in the context of web site logins by enabling users to authenticate to personal messaging identifiers (e.g., email addresses, IM handles, cell phone numbers). SAW, however, is ill-suited for wireless authentication because, in most cases, it is dependent on client-side Internet connectivity. Wireless Authentication using Remote Passwords (WARP) overcomes this constraint by building a hybrid protocol that combines the principles of SAW authentication with the Secure Remote Password (SRP) protocol.

Categories and Subject Descriptors

K.6.5 [**Management of Computing and Information Systems**]: Security and Protection—*Authentication*

General Terms

Security, Management

Keywords

decentralized wireless authentication, SAW, SRP

1. INTRODUCTION

Wireless networks provide an attractive means for network access as they enable convenient roaming and device deployability without the burden of network cables and port accessibility. Although convenient, wireless communications are subject to passive eavesdropping, active injection, and modification attacks. For these reasons, it is important that wireless networks provide secure authentication to prevent unauthorized access to network resources and provide confidentiality and integrity to transmitted information.

Current wireless authentication mechanisms are usually based on user-specific certificates (PKI), global passphrases, or username/password pairs (see Section 7). These methods are either too heavy or inflexible and lack general support for environments with dynamic user bases, such as corporations or universities where guest access is frequent.

Wireless Authentication using Remote Passwords (WARP) augments the Secure Remote Password (SRP) [12] protocol using concepts from Simple Authentication for the Web (SAW) [11]. By proving ownership of an authorized personal messaging identifier (e.g., email address, IM handle, cell phone number), WARP enables users to authenticate without pre-established secrets or the heavy cost and inconvenience of user-specific certificates. The burden of wireless access control in dynamic user bases is dramatically reduced as authentication of these globally unique identifiers is performed by trusted third parties.

2. BACKGROUND

Simple Authentication for the Web (SAW) [11] enables decentralized authentication of globally unique personal messaging identifiers (e.g., email addresses, IM handles, cell phone numbers). This approach is ideal for systems with dynamic user bases because no shared secrets (e.g., passwords) are required between clients and relying parties (e.g., web sites). Instead, SAW leverages unmodified personal messaging providers (e.g., email, text and instant message services) to act as identity providers to relying parties.

SAW builds on the same basic technique employed by the "Forgot your password?" link common to many web sites; users must retrieve personal messages (e.g., email, text and instant messages) sent to them by the relying party through their messaging provider. Relying parties depend on providers to deliver messages only to authenticated recipients. As WARP represents a significant departure from SAW, a detailed protocol overview is omitted.

WARP builds on the following principles of a SAW authentication:

1. Reuse existing identifiers and authenticators.
2. Tightly couple identifiers and identity providers, e.g., email addresses specify the locations of their respective mail servers.
3. Authentication requires that users obtain two tokens known to the relying party. The first token is given to the initiator of an authentication, while the second is only obtained after a successful authentication to the identity provider.

2.1 The Chicken and the Egg

As many personal messaging providers (e.g., email, instant messaging) rely on client-side Internet connectivity for message retrieval, an interesting chicken and egg problem must be overcome to adapt SAW for wireless authentication: How do clients communicate with their personal messaging providers when the reason they are authenticating in the first place is to obtain network and Internet connectivity? Four potential solutions have been identified:

Temporary Connectivity.
A user has limited time to access the required personal messaging resources before his connectivity is terminated. This approach carries increased liability and is undesirable as it allows anyone to have temporary access to network resources; access that could be used to launch attacks on or from the local network.

Filtered Connectivity.
Attempt to allow clients to exchange traffic with only their personal messaging providers. For example, clients obtain IP-level connectivity to a restricted network with limited Internet access. The client is switched to the regular network after successfully authenticating.

Protection against abuse is complex and potentially impossible. Sophisticated filters could restrict traffic to authorized personal messaging protocols, but filtering is impossible if encryption is used. Data-limiting caps could be used but would be difficult to fine-tune. The complexity of the safeguards needed to decrease the liability of this approach make it largely unacceptable.

Out-of-band Message Delivery.
The chicken and egg problem does not exist when out-of-band channels are employed to deliver messages to users, e.g., text messages sent through a cellular providers' network (SMS). This approach does not work in locations without cellular coverage and requires SMS-capable devices.

Surrogate Authentication.
A surrogate approach requires the party to which users are authenticating to relay small messages to their personal messaging providers on their behalf. In this approach an IP-level of connectivity is not required; the Extensible Authentication Protocol (EAP) [2] carries the authentication traffic. Since the authenticator, not users, directly communicates with the personal messaging provider it has greater control and can limit unauthorized data transfer.

As users are unlikely to trust authenticators to log in as them to their personal messaging providers, this approach must provide assurances that user login credentials cannot be stolen or misused by the authenticator.

WARP is a surrogate solution that provides strong protections to user credentials.

3. WARP

In this paper, user (**U**) refers to the user and its wireless supplicant, relying party (**RP**) refers to the wireless access point (and potentially an associated authentication server, e.g., RADIUS server) and identity provider (**IDP**) refers to the personal messaging provider.

In general, WARP should be both convenient and secure.

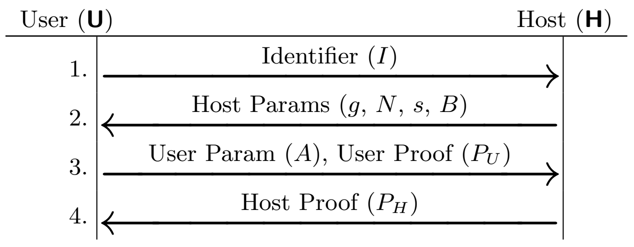
User (**U**)
Host (**H**)
1.
Identifier (I)
2.
Host Params (g, N, s, B)
3.
User Param (A), User Proof (P_U)
4.
Host Proof (P_H)

Figure 1: SRP protocol messages.

Specifically, it should:

- Preserve the three principles of SAW (see Section 2)
- Authenticate **U**
- Provide confidentiality and integrity to the session established between **U** and **RP**

To satisfy the first principle of SAW, WARP also reuses existing personal messaging identifiers and authenticators, e.g., email addresses and email account passwords.

With regard to the second principle, the location of **IDP** must be learned from/based on the user's identifier. For example, if an email address is used **IDP** would be hosted on a known port of the domain specified by the identifier, or specified in DNS similar to the `MX` (mail server) entry. Note that this requires **IDP**s to support this protocol; a significant departure from SAW, which leverages unmodified providers.

The third principle is the most challenging to fulfill. The purpose of the two token scheme is to allow **IDP** to assist **U** in the creation of an encrypted session with **RP** without **IDP** being able to compromise that session. It is simple for **RP** to deliver a token to each **U** and **IDP**, and relay small messages between the two. The difficultly arises by requiring **IDP** to confidentially deliver its token to only an authenticated **U**.

Conventional wisdom dictates that the simpliest way to accomplish this is to use a secret key known only to **U** and **IDP**. WARP's solution to this problem is to build on the Secure Remote Password (SRP) protocol, which is a password authenticated key agreement scheme that enables the establishment of a strong ephemeral session key from a potentially weak password.

The remainder of this section is organized as follows. Section 3.1 gives an overview of SRP. Section 3.2 documents sSRP, which contains the SAW-based additions to SRP. Section 3.3 shows how sSRP is used for wireless authentication.

3.1 Secure Remote Password

Secure Remote Password (SRP) [12] is a protocol designed to provide password-based mutual authentication between a user (**U**) and a host (**H**), and establish an ephemeral session key. SRP reveals no information to eavesdroppers during authentication that can be used to mount an offline attack against the password. It is also resilient against well-known passive and active attacks. The host does not store passwords for each identifier in plaintext but instead a unique salt and verifier. The salt is used with the plaintext password to generate the verifier. The verifier is not password-equivalent and cannot be used to impersonate the user.

As WARP treats SRP as a black box that results in a shared key between **U** and **H**, only a high level overview of SRP is given here. Figure 1 shows a condensed version of

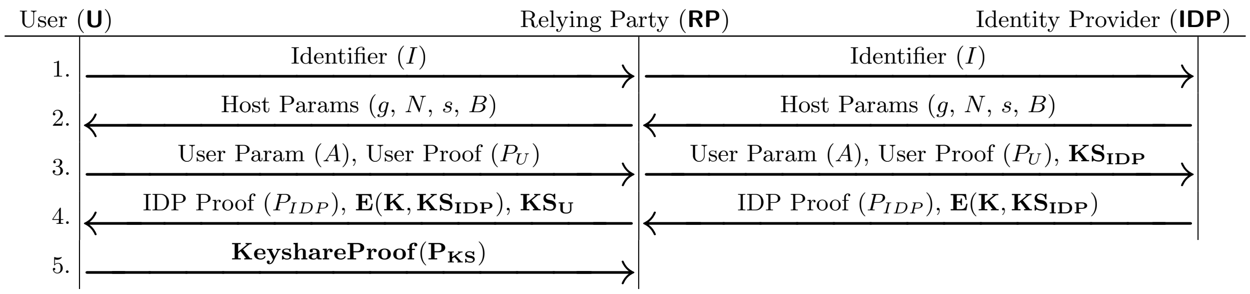
User (**U**)
Relying Party (**RP**)
Identity Provider (**IDP**)
1.
Identifier (I)
Identifier (I)
2.
Host Params (g, N, s, B)
Host Params (g, N, s, B)
3.
User Param (A), User Proof (P_U)
User Param (A), User Proof (P_U), $\mathbf{KS_{IDP}}$
4.
IDP Proof (P_{IDP}), $\mathbf{E}(\mathbf{K}, \mathbf{KS_{IDP}})$, $\mathbf{KS_U}$
IDP Proof (P_{IDP}), $\mathbf{E}(\mathbf{K}, \mathbf{KS_{IDP}})$
5.
KeyshareProof($\mathbf{P_{KS}}$)

Figure 2: sSRP protocol outline (augmentations to SRP in bold).

the SRP protocol currently being proposed as an extension [8] to TLS that operates as follows:

1. **U** sends its identifier (I) to **H**.

2. **H** looks up the public group parameters (g and N) for I as well as the user's salt and verifier (s and v). **H** computes B (its ephemeral session parameter) and returns it, along with g, N, and s to **U**.

3. **U** generates A (its ephemeral session parameter). Next, **U** computes the session key K using the values from **H**. **U** then sends A and a proof (P_U) of K to **H**.

4. **H** computes the session key K, verifies P_U, and sends its own proof (P_H) to **U**.

3.2 Surrogate SRP (sSRP)

WARP augments SRP with principles of SAW to create the Surrogate Secure Remote Password (sSRP). In sSRP, an identity provider (**IDP**) assumes the role of the SRP host. This protocol introduces a relying party (**RP**) through which messages between **U** and **IDP** are relayed. Like SRP, the protocol is initiated by the user (**U**).

sSRP treats the internals of SRP as a black box and only relies on SRP to create password-based ephemeral session keys between **U** and **IDP**. sSRP simply adds several message elements that are unrelated to the original, unmodified SRP messages. At a high level, we're piggybacking SAW token distribution over unmodified SRP. Figure 2 shows the sSRP protocol. Here is a description:

1. **U** submits its identifier (I) to **RP**, which authorizes the identifier and forwards it to **IDP**.

2. **IDP** looks up the public group parameters (g and N) for I as well as the user's salt and verifier (s and v). **IDP** computes B (its ephemeral session parameter) and returns it, along with g, N, and s to **U** via **RP**.

3. **U** generates its SRP session parameter (A), computes the session key K. and then sends A and a proof (P_U) of K to **RP**. **RP** generates a random value (KS) and splits it into two keyshares: KS_{IDP} and KS_U. **RP** appends KS_{IDP} to **U**'s message and sends it to **IDP**.

4. **IDP** computes the session key K, verifies P_U, and then encrypts KS_{IDP} using K. **IDP** then computes its own proof (P_H) and sends it, along with the encrypted KS_{IDP} to **RP**. **RP** appends KS_U before forwarding the message on to **U**.

5. **U** performs the following:

   (a) Decrypts KS_{IDP} using K.

   (b) Recreates KS using KS_{IDP} and KS_U

   (c) Creates the keyshare proof (P_{KS}).
   $P_{KS} = H(I\|g\|N\|s\|B\|A\|P_U\|P_{IDP}\|KS)$.

   (d) Sends P_{KS} to **RP**.

sSRP leaves both **U** and **RP** with shared key KS, which is used as keying material to encrypt future transmissions. This key is different than the SRP shared key K.

In order to protect against eavesdropping and impersonation attacks, the link between **U** and **RP** must provide confidentiality, integrity, and authentication of **RP**. This protects the transmission of KS_U as it is sent to the user (see Section 5.1).

Upon first inspection, it may seem like KS_U provides no additional assurances. KS_U serves two purposes: 1) Prevents IDP from having the full keying material (IDP never sees KS_U); and 2) Makes KS_{IDP} by itself worthless, as an attacker needs both keyshares to either impersonate the user or decrypt post-authentication transmissions.

sSRP enables proof of personal messaging identifier ownership without employing the personal messaging medium itself. Ubiquitously deployed, this service provides a mechanism useful not only to WARP, but also to SAW and many other mechanisms that rely on proof of identifier ownership through password-based authentication.

Using other identity providers.

Although this description of sSRP used personal messaging identifiers, e.g., email addresses or instant messaging handles, sSRP can be used with any password-based identity provider where an sSRP service can be deployed. This means that users could authenticate to a wireless network using identifiers such as OpenID or Unix logins.

3.3 Employing sSRP in WARP

WARP is an incarnation of sSRP for wireless authentication. In WARP, the wireless supplicant **S** takes on the role of the user and the authentication server (**AS**) that of the relying party.

EAP-WARP, a new EAP method, has been created to support WARP. EAP-WARP encapsulates the sSRP protocol as it travels between the supplicant and the authentication server. EAP-WARP works as follows:

1. **S** and **AS** use EAP-TTLS [5] to authenticate **AS** and provide confidentiality and integrity for the link.

2. **S**, **AS**, and **IDP** perform sSRP. Upon submission of the sSRP keyshare proof P_{KS}, **S** has proven ownership of an authorized identifier to **AS**.

3. **AS** sends an EAP-Success message back to **S**.

4. **S** and **AS** use KS to derive the EAP Master Session Key (MSK), which provides confidentiality and integrity for the subsequent wireless session.

4. IMPLEMENTATION

Software to support wireless authentication using WARP has been developed and will soon be available.

`libssrp` is a general purpose library written in C that provides the functionality needed to conduct sSRP authentication. It is meant to be used by applications that supply their own transport functionality.

The current version of `libssrp` relies on `OpenSSL` [10] for its cryptographic primitives and arbitrary precision integers. `libssrp`, by default, relies on an SRP-compatible password file populated with salts and verifiers generated from plaintext passwords. An API can alternatively be used to allow flexible retrieval of salts and verifiers. An argument to stay with the default configuration is given in Section 6.

The `wpa_supplicant` [7] open-source package has been extended to support EAP-WARP. The extension consist of two files: 1) One C source file to provide EAP-WARP support; and 2) A patch file that modifies `wpa_supplicant` to include the extension. The extension is less than 400 lines of code.

This simple extension provides a layer that extracts sSRP packet data and provides it to `libssrp`. sSRP packet data returned from `libssrp` is inserted into an EAP packet that is returned to `wpa_supplicant`. The extension also exports the EAP MSK, derived from KS, to `wpa_supplicant`.

`FreeRADIUS` [9], an open-source RADIUS server, has also been extended with EAP-WARP support. The extension is around 800 lines of C code and comments, and like the extension for `wpa_supplicant`, provides extraction and insertion of sSRP messages to and from EAP packets, as well as exporting keying material. The `libssrp` library is again used to provide the bulk of the functionality.

An incarnation of the sSRP service has been written using the `libssrp` library and provides sSRP over TCP/IP. The service is written in C, can daemonize, and supports logging to syslog.

5. THREAT ANALYSIS

This section contains a threat analysis of WARP and the underlying sSRP protocol to enable proper risk evaluation by those deploying WARP.

SRP and SAW are the parent protocols of sSRP. SRP is already resilient to passive eavesdropping and active modification or impersonation attacks. sSRP purposefully inserts a middle party in between the user and identity provider in SRP. This creates two channels for attackers to target. Section 5.1 discusses threats to the channel in between **U** and **RP**. Threats on the channel between **RP** and **IDP** is provided in Section 5.2. Section 5.3 evaluates threats when both channels are available to an attacker.

A discussion regarding impersonation by **IDP** is provided in 5.4. Section 5.5 theorizes how WARP could be used to mount a denial-of-service attack. Section 5.6 concludes the threat analysis by discussing how sSRP could be abused as a covert channel and how such an attack is limited.

5.1 Channel between U and RP

The channel between **U** and **RP** must provide confidentiality, integrity, and authentication of **RP** in order to protect the transmission of KS_U. A man-in-the-middle attack is still possible over this channel depending on how authentication of the relying party is implemented. It is therefore necessary for sSRP to provide protection if the user connects to an attacker instead of the intended relying party.

For example, WARP uses EAP-TTLS to provide security on this channel and authenticate **AS**. In order to prevent man-in-the-middle attacks, the supplicant would need to verify **AS**'s certificate before accepting the connection. A careless supplicant choosing not to verify the certificate would allow a man in the middle to place himself between the supplicant and **AS**. The supplicant would establish a TLS session with the attacker, who would then establish a TLS session with **AS**. The supplicant would be oblivious to such an attack. All traffic would now flow through the attacker. The attacker can now observe KS_U.

KS_U, by itself, is useless since, without knowledge of KS_{IDP}, the attacker is unable to derive KS. KS_{IDP} is encrypted with the SRP session key K before travelling across this channel to prevent the attacker from obtaining it. The attacker, who does not know K, is unable to decrypt the keyshare and subsequently unable to impersonate the user.

A man-in-the-middle attack could allow an attacker to intercept P_{KS} before it arrived at the relying party. The proof could then be sent by the attacker to impersonate the user. This is fruitless, however, as knowledge of KS is required to further communicate with **RP**. In WARP terms, this means that the attacker would not have the correct EAP keying material to export and would therefore be incapable of communicating further with the access point.

5.2 Channel between RP and IDP

The user relies on **RP** to connect to the correct identity provider. Even if **RP** connects to a malicious identity provider, sSRP prevents that provider from learning anything about the user's password.

The channel that is established between **RP** and **IDP** is insecure. SRP parameters are protected by the built-in protections provided by SRP. KS_{IDP} is sent across this channel twice in this protocol: 1) In the clear on its way to the IDP; and 2) Encrypted with the shared SRP session key K as it is sent back to **U**. **AS** has knowledge of KS_{IDP} and could attempt to brute-force K by encrypting KS_{IDP} with all possible values for K. Even if **AS** is able to obtain K, K is not helpful in discovering the password [13].

5.3 Both Channels

If Mallory, who can observe the channel between **U** and **RP**, colludes with Eve, who can observe the channel between **RP** and **IDP**, then a one-time impersonation of the user is possible. To initiate this attack, Eve passively observes KS_{IDP} as it is sent from **RP** to **IDP**. Mallory likewise obtains KS_U as described in Section 5.1. If Eve is able to communicate KS_{IDP} to Mallory, then Mallory can construct KS and impersonate the user. This impersonation is limited to a single authentication because the keyshares are single-use and short-lived.

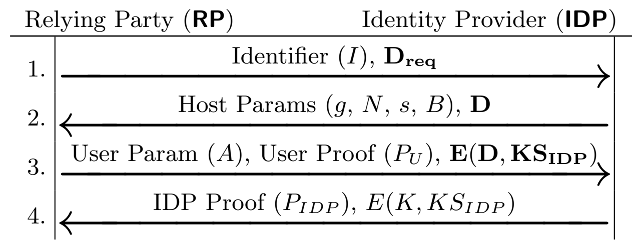
Relying Party (**RP**)
Identity Provider (**IDP**)
1. Identifier (I), $\mathbf{D_{req}}$
2. Host Params (g, N, s, B), $\mathbf{D}$
3. User Param (A), User Proof (P_U), $\mathbf{E(D, KS_{IDP})}$
4. IDP Proof (P_{IDP}), $E(K, KS_{IDP})$

Figure 3: One-time impersonation resistant sSRP (augmentations shown in bold).

Although this one-time impersonation attack is complex and unlikely, it can be avoided. Encrypting KS_{IDP} with **IDP**'s public key before it is sent to **IDP** provides confidentiality, since only **IDP** is be able to decrypt it. An alternative approach would be to encrypt the entire channel between **RP** and **IDP**, but this creates unnecessary overhead as the remainder of the message elements are already protected.

Figure 3 shows a slight modification to the interaction between **RP** and **IDP** in sSRP, which prevents this one-time impersonation attack. The modifications are as follows (other interactions remain unchanged):

1. **RP** appends the optional public key request, D_{req}, to A as it travels to **IDP**.

2. **IDP** fills the request by sending his public key certificate along with B and s to **U** through **RP**. The certificate contains his public key (D).

3. **RP**: verifies and strips off the certificate.

4. **RP**: encrypts KS_{IDP} with D before it is sent to **IDP**.

The encrypted KS_{IDP} is only decryptable by **IDP** and is therefore useless to an eavesdropper.

5.4 Impersonation By IDP

Identity providers can impersonate any of their users. If an attacker or malicious insider takes control of the sSRP service hosted by the provider, they can impersonate users to gain wireless connectivity. For example, the attacker could replace the password verifier of an authorized identifier with his own or alter the sSRP service to always grant successful authentication.

WARP relies on an SRP password file on the identity provider to provide verifiers, salts, and public parameters for authorized users. Because the password verifiers are not plaintext equivalent to the password, an attacker who steals the password file is unable to use the verifier to impersonate the user without first determining the user's password.

SAW, and therefore WARP, are built around existing trust given to identity providers. An organization adopting WARP must make judgments regarding the identity providers they choose to trust. Some organizations may alternatively require use of identifiers within its own systems (e.g., organizational email addresses) to satisfy a required trust level.

5.5 Denial-Of-Service (DoS)

SRP is resilient against attackers modifying the information being exchanged between **U** and **H**. sSRP does nothing to compromise this resilience. At most, an attacker could cause authentication to fail. This could be used to deny service to an otherwise authorized user. As simpler, jamming-based denial-of-service attacks (DoS) already exist, the potential for DoS attacks using WARP is negligible.

5.6 Covert Channels

A covert channel is a method of communication that uses another channel's bandwidth to transmit data without knowledge or consent. Covert channels take many different forms (e.g., steganography, timing between transmissions, text manipulation). WARP could be used to send data between **U** and **IDP**. Because WARP only opens up a channel of communication between the supplicant and the provider of an authorized identifier, it cannot be used as a covert channel to any arbitrary party.

Using WARP as a covert channel would not be an effective means of transferring large amounts of data; the number and size of WARP's messages are quite small. Similar authentication request throttling used to limit DoS attacks could also be employed to greatly reduce the amount of information that could be exchanged over the covert channel.

Since the link between the AS and IDP is unencrypted, the supplicant could communicate information to a passive eavesdropper on that link.

6. DEPLOYABILITY

Deployability of WARP can be discussed from three points of view: wireless users, organizations providing wireless access, and identity providers.

Users.

WARP leverages familiar and convenient password-based interfaces while decreasing the total number of required user passwords by reusing existing login credentials.

Organizations.

WARP softens the burden of password and account management. Providing access to regular organizational staff could be automated by generating the access control list using existing knowledge of staff identifiers. Guest access could then be maintained through a second access control list.

WARP is unusable in situations where the authentication server is unable to communicate with identity providers (e.g., ad-hoc networks without Internet connectivity).

WARP assumes the same level of trust extended to identity providers by SAW and may be inappropriate for use in organizations where the existing trust extended to third-party identity providers is insufficient.

Identity Providers.

WARP requires identity providers to host an sSRP service. Until incentives outweigh the cost, it is unlikely that every identity provider will be willing to meet this requirement. Fortunately, sSRP is simple and easy to implement and therefore comes at a minimal cost.

As mentioned previously, sSRP relies on an SRP password file to provide verifiers, salts, and public parameters for authorized users. Migration of existing non-SRP password files is tricky because these files generally contain non-plaintext forms of the passwords, e.g., password hashes, and entries in an SRP password files must be generated from plaintext user passwords.

Identity providers who currently use services that receive a plaintext password from the user, e.g., Unix logins, could modify the service to intercept the password and generate an SRP password file entry. This removes the need for a formal re-enrollment process; users log in once to enable sSRP. The modified service could be deployed well in advance of sSRP deployment to generate SRP password entries for active users. Since existing password files are not used to generate the verifiers, they can be safely maintained side-by-side the SRP password file; facilitating gradual deployment.

7. RELATED WORK

Many authentication mechanisms rely on pre-established shared secrets. Global passphrases used in systems like WPA-PSK[1] can be difficult to distribute. If the shared secret is compromised a new one must be deployed to each device. Guest access can only be achieved by disclosure of the shared secret. It is also difficult to audit access because each user connects using the same passphrase.

Individual user accounts with passwords are employed by challenge/response-based systems such as MSCHAPv2 [14]. Account-level password-based authentication provides reasonable deployability for a static group of users. It also lends itself to auditing because authentication is tied to a set of credentials. However, these systems suffer from the same issues of other password-based authentication mechanisms: password re-use and difficulty in remembering strong passwords. It is also difficult to configure guest access where account creation incurs significant overhead or time.

Recent authentication mechanisms, e.g., EAP-TLS [3], rely on user-specific digital certificates to provide authentication. However, managing user-specific certificates and securing public/private key-pairs are challenging tasks for even savvy end-users. The sign-up process can be resource intensive, adding additional overhead on administrators who must verify user identity and issue certificates according to stringent company policies. Often times this process must be performed not only for each user, but for each of the user's devices, which increases complexity.

Greenpass [6] leverages EAP-TLS for authentication. EAP-TLS relies on PKI to provide the client and server certificates used in the TLS handshake. Greenpass also provides decentralized delegation of access by allowing delegators to sign a SPKI/SDSI certificate that a guest can then use as their EAP-TLS client-side certificate. Greenpass still involves a CA issuing user-specific certificates to regular users.

Network-in-a-Box (NiaB) [4] enrolls devices in the wireless network by employing location-limited communication channels (e.g., infrared or a USB key) to securely distribute the necessary PKI keys and certificates. Although key distribution is simplified, NiaB still requires an administrator be present to authorize each device during enrollment. This location-limited channel approach creates a physical bottleneck when many devices need to be enrolled within a short time period (i.e., conference wireless access where most delegates arrive within a short time period).

8. CONCLUSIONS AND FUTURE WORK

WARP is a convenient and secure wireless authentication mechanism. By preserving the principles of SAW, WARP enables decentralized user authentication and facilitates dynamic user bases in the wireless realm without requiring client-side Internet or IP-connectivity. WARP's use of SRP enables users to authenticate using existing personal messaging account identifiers and passwords, without fear that relying parties or eavesdroppers can compromise their login credentials. Although identity providers assist users in creating authenticated, encrypted sessions with relying parties, the identity providers cannot compromise these sessions.

sSRP could replace the use of email or instant messages in the original SAW protocol for website logins. Using sSRP in this manner increases SAW's ability to thwart active impersonation attacks and eliminate latency issues associated with personal message delivery.

We are currently investigating how to integrate the intuitive delegation between personal messaging identifiers and natural client-side auditing capabilities of SAW into WARP.

Forthcoming revisions to sSRP eliminate the need to use PKI-based approaches, e.g., EAP-TTLS, to prevent passive observation of KS_U and KS_{IDP}. These revisions also enable the authentication of relying parties to identity providers.

Acknowledgments. This research was supported by funding from the National Science Foundation under grant no. CCR-0325951, prime cooperative agreement no. IIS-0331707, and The Regents of the University of California.

9. REFERENCES

[1] *Wi-Fi Protected Access.* http://wifialliance.com.

[2] B. Aboba, L. Blunk, J. Vollbrecht, and J. Carlson. RFC 3748: Extensible Authentication Protocol (EAP), June 2004.

[3] B. Aboba and D. Simon. RFC 2716: PPP EAP TLS Authentication Protocol, October 1999.

[4] D. Balfanz, G. Durfee, R. Grinter, and D. Smetters. Network-in-a-Box. In *USENIX Security Symposium*, August 2004.

[5] P. Funk, S. Blake-Wilson. Internet Draft: EAP Tunneled TLS Authentication Protocol Version 0, draft-funk-eap-ttls-v0-02, November 2007.

[6] N. Goffee, S. Kim, S. Smith, P. Taylor, M. Zhao, and J. Marchesini. Greenpass: Decentralized, PKI-based Authorization for Wireless LANs. In *PKI Research and Development Workshop*, April 2004.

[7] J. Malinen. wpa_supplicant. Available from http://hostap.epitest.fi/wpa_supplicant/.

[8] D. Taylor, T. Wu, N. Mavrogiannopoulos, and T. Perrin. Internet Draft: Using SRP for TLS Authentication, draft-ietf-tls-srp-14, June 2007.

[9] The FreeRADIUS Server Project. FreeRADIUS. Available from http://www.freeradius.org/.

[10] The OpenSSL Project. OpenSSL. Available from http://www.openssl.org/.

[11] T. van der Horst and K. Seamons. Simple Authentication for the Web. In *Security and Privacy in Communication Networks*, September 2007.

[12] T. Wu. The Secure Remote Password Protocol. In *Network and Distributed System Security Symposium*, March 1998.

[13] T. Wu. SRP-6. Technical white paper, Stanford University, October 2002.

[14] G. Zorn. RFC 2759: Microsoft PPP CHAP Extensions, Version 2, January 2000.