Network Working Group   Scott Kelly, Jim Knowles
INTERNET-DRAFT   RedCreek Communications
draft-kelly-ipsra-userauth-00.txt   Bernard Aboba, Microsoft
Expires in 6 months   October, 1999

User-level Authentication Mechanisms for IPsec

Status of This Memo

Comments on this document should be sent to "ietf-ipsra@vpnc.org", the IPsec remote access discussion list.

Abstract

IPsec, when used with IKE [RFC2409], provides for authentication of endpoints from the device level to the user level. However, there has been movement within the IPsec development community to provide additional support for legacy user-level authentication mechanisms such as those supported by RADIUS [RFC2138]. At least 2 approaches to this problem have been proposed thus far, both using the same basic underlying framework, but that underlying framework relies upon extending IKE in ways that may not be prudent. This document proposes an alternative approach which provides much of the same functionality without requiring any modification to the existing IPsec framework.

1. Introduction

IPsec, when used with IKE [RFC2409], provides for strong authentication of endpoints from the device level to the user level. The authentication methods supported range from preshared secrets to X.509 certificates. The entity authenticated by these mechanisms may be only the device, especially in cases where no user input is required in order for the subject device to access the IKE authentication credential. However, varying levels of user input may be required to provide the device with access to the authentication credential, in which case a single-factor or two-factor authentication mechanism may be employed.

An example of a single-factor authentication mechanism which provides relatively weak user-level authentication would be one in which the user provides a password which is then used as a preshared key for IKE authentication. A second example, perhaps slightly stronger, might consist in the user plugging in a smartcard which contains the authentication material (and perhaps applies it using an embedded processor), but which does not require the user to somehow "unlock" it. An example of strong two-factor authentication in this context might be realized if the user were required to first plug in a smart card containing the user's identity certificate, and then unlock that card using a non-trivial challenge/response mechanism. Even this mechanism has its limitations, depending upon one's level of paranoia, but these limitations are not due to any shortcoming in IPsec per se; rather, they are inherent in the nature of computer security.

While numerous authentication mechanisms (including some very strong ones) are currently supported by IPsec, there has been movement within the IPsec development community to provide additional support for legacy authentication mechanisms such as those supported with RADIUS [RFC2138] and GSS_API [GSS]. RADIUS-supported mechanisms include authentication methods supported in PPP [RFC1661], including PAP, CHAP, and EAP [RFC2284]. Such support is generally viewed as a necessary stepping stone to broad support for public key mechanisms.

The movement for legacy support has been especially concerned with deployments supporting remote access, wherein users typically access the resources of a corporate network from a remote location. In many of these cases, other remote access mechanisms such as those discussed above have been in use prior to the IPsec deployment, and in such cases, the administrators of these deployments often wish to continue using installed authentication and accounting mechanisms in order to preserve their investment in these technologies and lessen administrative costs.

At least 2 approaches to this problem have been proposed [XAUTH, HYBRID], both using the same basic underlying framework, but differing in otherwise substantial ways. However, the underlying framework detailed in [XAUTH] relies upon extending IKE in ways that may not be prudent. This document proposes an alternative approach which provides much of the same functionality without requiring any extension to the existing IPsec framework.

1.1 Reader Prerequisites

Reader familiarity with RFCs 2401-2412 is a minimum prerequisite to understanding the concepts discussed here. Familiarity with RADIUS, L2TP, and PPP [RFC1661] will also be helpful, though not strictly necessary.

1.2 Requirements Keywords

The keywords "MUST", "MUST NOT", "REQUIRED", "SHOULD", "SHOULD NOT", and "MAY" that appear in this document are to be interpreted as described in [RFC2119].

2. User-level Authentication

2.1 General considerations

In general, the motivation for extended authentication methods arises from the network administrator's desire to authenticate the remote user in a meaningful manner prior to providing access to valued resources. While such functionality may ultimately be provided by a reliable Public Key Infrastructure (PKI), such a PKI is not yet widely deployed. As a result, many administrators providing remote access desire IPsec support for the authentication and accounting mechanisms which they currently utilize with other remote access mechanisms.

In addition to the lack of ubiquitous PKI, there is other motivation for continued support of existing mechanisms. While we might imagine a future in which all computer users carry a hardware mechanism such as a smart card, and in which all computer systems have peripheral hardware capable of utilizing such mechanisms, this day may never arrive. Regardless, cases will exist in which PK certificates are somehow stored upon the system from which they are used. While these certificates may be password (or otherwise) protected, there is always the possibility that either the certificate access mechanism has been compromised, or that the current session user is not the user who initially provided access to the certificate. In these cases, the end user is not authenticated; rather, the machine itself has been authenticated.

As a matter of clarification, it is important to make a distinction between authenticating a device and authenticating a user. In cases where no user interaction is required in order for a device to form an IKE security association with another system, it is clearly the device which has been authenticated, and not the user. In cases where user input is required which influences the authentication credential selection or production process, some degree of user-level authentication results. However, as noted above, this degree of authentication may be dependent upon other factors, and may vary with time.

Given the varying levels of user authentication which are achievable using PKI mechanisms, it will likely remain desirable to continue to provide support for existing and new User-Level Authentication (ULA) techniques which may be used in conjunction with PKI mechanisms. In general, existing ULA techniques are based on one of 2 mechanisms: static pass phrases, or dynamic tokens. Dynamic tokens may be time-dependent, or may be formulated as a response to a challenge of some sort, while static pass phrases are generally text strings of some sort which are associated with a given identity. Static pass phrases are usually a very weak form of authentication. On the other hand, they will likely remain popular due to their convenience.

It seems clear that the ideal solution to the ULA problem entails an approach which supports existing ULA mechanisms while paving the way for PKI deployment, since such mechanisms will likely continue to be required even after a PKI is deployed. Therefore, our goal should be to provide an integration strategy which couples these mechanisms in the most secure manner. Such a strategy necessarily entails a PKI transition. This is discussed in the next session.

2.2 The PKI Transition

The need for integration of a PKI transition strategy with the deployment of other ULA mechanisms is evidenced by the movement toward providing ULA mechanisms within IKE as a response to remote access requirements. In general, some sort of PKI is a hard requirement for widespread deployment of secure remote access solutions, since the security of preshared keys is highly dubious in this context. The issues associated with such a PKI transition are discussed below.

The transition to a PKI may be divided into several steps:

a. Support for PKI on VPN servers. This typically requires that machine certificates be deployed on the servers, along with appropriate certificate authorities and stores. It also requires that clients be capable of verifying the server's certificate against a current Certificate Revocation List (CRL). Since this will often require a client software upgrade, the work to transition to server certificates is comparable to the work required to deploy SSL/TLS-capable Web server and certificate-capable browsers.

   Note that while client software support for PKI must be assumed, in this step, it is not necessary for the clients to obtain their own machine or user certificates. Thus it is possible for the clients to continue to authenticate using only legacy methods during this phase of the transition.

b. Support for machine certificates on VPN clients. This requires that machine certificates be deployed on VPN clients. Completion of the previous step (a) often requires a client upgrade, which will typically also include support for client certificates. If the infrastructure for machine auto-enrollment has also been put in place as part of the server PKI rollout, then there may not be much additional work required to complete this step, above what was already required for the previous step. Note that if the VPN client only supports a machine certificate, then this may imply the use of a non-PKI method for user authentication in addition to the machine certificate.

c. Support for user certificates. This requires that user certificates be provided to users. Since storage of user certificates on the machine creates new vulnerabilities, smartcards may typically be used to store the user certificates. Thus, a smartcard rollout may often be a prerequisite to deployment of user certificates. This in turn may require integration of smartcard provisioning with the existing identification system, such as the distribution of combined employee badge/smartcards. Since this step may require considerable work above and beyond the tasks required to carry out transition steps a and b, support for legacy authentication methods will likely be required at least until this transition step is complete.

2.3 Previously Proposed Solutions

There have been at least two mechanisms proposed as solutions for the ULA problem thus far, and these are discussed in [XAUTH] and [HYBRID]. While these documents do not explicitly acknowledge the need for a PKI transition, they rely heavily upon it. [XAUTH] relies upon it because it requires a reliable authentication mechanism for the communicating systems (i.e. PK certificates) in order to be meaningful in a security context, and [HYBRID] relies upon it due both to its incorporation of PK certificates on the server side (and verification on the client side), and to its reliance on [XAUTH] as an underlying framework. These are discussed individually below, followed by a discussion of an alternative mechanism.

2.4 Extended Authentication within IKE (xauth)

The xauth mechanism [XAUTH] utilizes an existing phase 1 IKE Security Association (SA) to protect a secondary authentication exchange within IKE prior to negotiation of an IPsec protocol SA. This is often referred to as "phase 1.5" due to its juxtaposition between IKE phases 1 and 2. This mechanism currently relies upon another proposed IKE protocol extension mechanism [ISACFG], and both substantially modify the existing IKE protocol. The following diagram clarifies the relationships of the various components:

```
                  +-------------------+
+----------+      | security gateway |           +---------+
| radius   |------| +----------------+|  IKE SA   | remote  |
| server   |      | | co-resident    ||<=========>| client  |
|          |<======>|authentication ||           |         |
+----------+      | | proxy appl.    ||           +---------+
                  | +----------------+|
                  +-------------------+
```

Issues with xauth include the following:

o susceptible to man-in-the-middle attack if preshared key is used for IKE authentication, and so requires both server and client PK certificates for most deployments.

o depends upon yet another framework for its basis [ISACFG]

o adds significant complexity to the key exchange protocol, not only by adding an open-ended number of challenge-response exhanges, but by providing proxy support for 16 different legacy authentication mechanisms. The resultant implementation complexity introduces significant security risks.

o there is substantial known plaintext in the encrypted exchanges.

These are serious issues, and should disqualify this proposal from serious consideration as a security protocol extension. To clarify, each of these issues is discussed individually below.

Man-in-the-middle attacks

The vulnerability to man-in-the-middle attacks occurs because in preshared key authentication in main mode, it is necessary for keys to be computed prior to the receipt of the identification payload. Therefore the selection of the preshared key may only be based on information contained in the IP header. However, in remote access situations, dynamic IP address assignment is the rule, so that it is typically not possible to map an IP address to a user identity and a preshared key. Thus the preshared key can no longer function as an effective shared secret; the same preshared key must be shared by a group of users. In this situation, neither the client nor the server individually authenticates itself during IKE phase 1; it is only known that both parties are a member of a group with access to the shared secret. This permits anyone with access to the group secret to act as a man-in-the-middle. Hence, [XAUTH] requires both server and client certificates in most cases.

Note that this vulnerability does not occur in aggressive mode since the identity payload is sent earlier in the exchange. Of course, when aggressive mode is used the user identity is exposed and this may be undesirable. In fact, aggressive mode raises other security concerns, but these are not discussed here.

Dependency on [ISACFG]

[XAUTH] requires support for the additional proposed configuration mechanism described in [ISACFG]. This presupposes that this configuration framework is appropriate in its own right, but this has not been demonstrated. Binding [XAUTH] to [ISACFG] increases the difficulty associated with complexity analysis, and requires that the two proposals advance together.

Complexity Issues

The approach described in [XAUTH] significantly complicates IKE by adding a new exchange type, by extending the negotiation process in an open-ended fashion, by binding itself to yet another IKE extension, and by adding text-string processing requirements. While either of the first two issues taken alone might not be cause for much alarm, the aggregation of these, along with the others, render this mechanism significantly more complex than the base IKE exchange, and much more prone to error. Given the critical nature of the key exchange with respect to the resulting session security, it is imprudent to unnecessarily introduce complexity to this protocol component.

Known Plaintext

There is a significant amount of known plaintext in the exchanges described in [XAUTH]. Below is an example exchange. Note that the text given here in upper case is precisely what is encrypted and sent over the wire (for the given authentication type), as documented in [XAUTH]:

```
IPSec Host                         Edge Device
--------------                  -----------------
                    <-- REQUEST(TYPE=GENERIC NAME=""
                                PASSWORD="")

REPLY(TYPE=GENERIC NAME="joe"
          PASSWORD="foobar") -->

                    <-- REQUEST(TYPE=GENERIC
                                MESSAGE="Enter your password
                                followed by your pin number"
                                NAME="" PASSWORD="")

REPLY(TYPE=GENERIC NAME="joe"
      PASSWORD="foobar0985124") -->

                    <-- SET(STATUS=OK)
```

Without question, there is a significant amount of known plaintext here. An adversary could passively collect any number of these exchanges, and then analyze them at leisure. Note that it is not necessary to break the session in real time in order to compromise a password and subsequently gain access to the remote network. However, real time attacks are a possibility in sufficiently long-lived sessions.

It must be emphasized that it is not the general functionality that [XAUTH] strives for which is in question here, though the advisability of providing proxy protocol support for 16 different legacy authentication mechanisms is certainly questionable. Rather, it is the insertion of this mechanism into the key exchange protocol, and the general manner in which it is constructed, which should be recognized as imprudent.

Note that the support for such a large number of legacy methods appears unnecessary, given that most of the mechanisms are already supported within existing IETF standards-track frameworks such as GSS_API [GSS], SASL [SASL] and EAP [RFC2284]. Thus, the introduction of yet another user authentication framework is highly inefficient. By leveraging existing frameworks it would be possible to simultaneously provide wider legacy support while dramatically decreasing the code and complexity required to provide this functionality.

2.5 A Hybrid Authentication Mode for IKE

The hybrid authentication method [HYBRID] uses [XAUTH] as a framework upon which to build. Hence, it suffers from many of the deficiencies of [XAUTH], namely excessive key exchange protocol complexity, dependency on [ISACFG], and substantial known plaintext in the exchanges. However, the hybrid mechanism resolves the man-in-the-middle susceptibility by requiring the security gateway to authenticate using a certificate, while permitting the user to be authenticated based upon some challenge-response mechanism.

By only requiring a server certificate, Hybrid authentication provides more transition benefits than [XAUTH], which can only be safely deployed with both server and client machine certificates. However, as noted earlier, once full server certificate support is put in place, there may not be that much extra work involved in supporting client machine certificates. Thus the hybrid approach may provide only limited transition benefits above what is already supported in IKE.

If [HYBRID] were made independent of [XAUTH], it would be substantially enhanced. However, there would still be questions as to its necessity and utility. For [HYBRID] to provide meaningful security, the client must be able to validate the security gateway's certificate, or else session is susceptible to a man-in-the-middle attack. That is, this functionality relies upon the presence of a PKI infrastructure for its security. So, [HYBRID] requires a meaningful PKI capability in both the client and the server, yet it stops short of simply taking one more step and enrolling the client for a certificate of its own. If the client had its own certificate, the need for the hybrid framework would disappear entirely.

On the other hand, and especially in view of the desire for legacy-to-PKI transition mechanisms, there may yet be some limited usefulness for the hybrid approach, especially if it is freed from dependence upon xauth. The authors of [HYBRID] are invited to consider how its general principles might be integrated into the framework presented below.

2.6 Reasonable design goals

Before proceeding to propose a solution to the various problems associated with ULA in the IPsec context, we should first assert some design goals. These might include the following:

- o Provide support for existing legacy components, and make it transparent inasmuch as this is possible

- o Permit a transition to Public Key infrastructure beginning at step a in the transition process, support for server certificates.

- o Use the existing IPsec framework without modification if possible

- o Use existing standardized authentication framework(s)

- o Provide mechanisms which ultimately encourage migration toward newer, stronger authentication technologies, but which do not force such migration. Regardless, do not permit a failure to migrate to more appropriate technologies by some administrators to precipitate the weakening of security protocols as an IETF response.

3. An Alternative to Xauth and Hybrid

3.1 Overview

It is not difficult to arrive at an alternative to xauth/hybrid which does not suffer from many of the associated shortcomings, given the design goals enumerated above. Supporting existing legacy systems requires either that the client or the security gateway (sgw) speak with the legacy authentication server. If this is to be password based, it makes sense that the sgw implements a proxy server so that it is privy to the results of the exchange, and this is also supported by the desire to use an existing standardized authentication protocol. Also, if we are to use the existing Ipsec framework, we must somehow use the policy database on the sgw to control access. And if we are to encourage migration, we should strive to encourage the use of a PKI in place of or in addition to transmitted passwords or tokens. Given these considerations, a fairly straightforward scheme emerges, detailed below:

o the security gateway implements a co-resident authentication proxy application which uses a standard authentication protocol.

o the client establishes a securely authenticated IKE SA followed by an IPsec SA with selectors which indicate the authentication proxy application on the gateway.

o the client interacts with the authentication proxy, providing a password, token or whatever is required.

o if the client successfully authenticates, the authentication proxy adds selectors to the Security Policy Database (SPD) which permit access to the corporate network. These selectors MAY point to the same phase 2 SA as was used for authentication, or MAY require the instantiation of new phase 2 SA. In the case of a new phase 2 SA, the original selectors permitting access to the proxy application MAY remain as well, allowing for later re-authentication if needed. This selector addition results in a temporary selector entry in the same manner as name selectors described in [RFC2401, p19].

o If the client fails to authenticate after a (small) predetermined number of attempts, the client MUST be locked out. Failure to authenticate is an auditable event.

This mechanism is much simpler than those previously proposed, in that it relies almost entirely upon the existing framework, and requires no modifications to existing IPsec protocols. Compliant IPsec implementations should already support the addition of temporary policy selectors, and existing xauth/hybrid implementations already must support a co-resident authentication proxy.

3.2 The ULA Protocol

Of fundamental importance to the simplicity of the above described mechanism is the authentication proxy, and the protocol(s) it supports. Other proposed mechanisms advocate support for numerous individual mechanisms, leading to unacceptable implementation complexity. This problem has been addressed within existing IETF authentication frameworks such as EAP [RFC2284], GSS_API [GSS], and SASL [SASL].

Note that while these frameworks provide differing degrees of integrity protection taken on their own, when carried out under the protection of an IPSEC SA based on a fully authenticated IKE SA, they would inherit the authentication, integrity and replay protection services of IPSEC. This would be effectively realized by encapsulation of relevant authentication protocol exchanges within an IP transport protocol. Note that while GSS_API and EAP can be encapsulated within UDP, SASL support assumes TCP transport. Thus an additional port would be needed to serve as a UDP/TCP endpoint.

4. Shortcomings and Strengths compared to other mechanisms

4.1 Shortcomings

- this approach is somewhat susceptible to Denial of Service (DoS) attacks, due to the amount of processing necessary to generate both IKE and IPsec protocol Sas. However, such an attack requires that the phase 1 credential be compromised, and may be mitigated by lockout after some number of failures.

- There may be known plaintext in the authentication exchange. This may be mitigated by random ordering of TLV payloads within the exchange, and in any event, this mechanism provides far less known plaintext than xauth or hybrid.

4.2 Strengths

- no modifications to existing IPsec protocols

- significantly reduces implementation complexity

- provides clear migration path to PKI-based mechanisms

- utilizes standardized authentication protocols

5. Security Considerations

This document describes a mechanism for providing various degrees of user-level authentication prior to allowing general access via an IPsec protocol SA. It is assumed that the IKE (phase 1) SA is authenticated in a meaningful manner prior to engaging in user-level authentication. If preshared keys are used for such authentication, extreme care must be exercised in distributing and storing these keys, since compromise of the preshared key results in susceptibility to a man-in-the-middle attack. In cases where compromise could result in significant loss, preshared keys SHOULD NOT be used.

It must be emphasized that this mechanism adds nothing to the security of the underlying IKE/IPsec SAs, but simply serves instead to authenticate a user whose data is to be transported within the associated protocol SA.

6. Authors' Addresses

Scott Kelly
RedCreek Communications
3900 Newpark Mall Road
Newark, CA 94560
USA
email: skelly@redcreek.com
Telephone: +1 (510) 745-3969

Jim Knowles
RedCreek Communications
3900 Newpark Mall Road
Newark, CA 94560
USA
email: jknowles@redcreek.com
Telephone: +1 (510) 745-3977

Bernard Aboba
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052
USA
email: bernarda@microsoft.com
Telephone: +1 (425) 936-6605

Full Copyright Statement

7. References

[GSS]   Linn, J., "Generic Security Service Application Program Interface, Version 2", RFC 2078, January 1997.

[HYBRID]   M. Litvin, R. Shamir, and T. Zegman, "A Hybrid Authentication Mode for IKE", June 1999, draft-ietf-ipsec-isakmp-hybrid-auth-02.txt

[ISACFG]   R. Pereira, "The ISAKMP Configuration Method", draft-ietf-ipsec-isakmp-cfg-03, (expired)

[RADIUSEXT] Rigney, C., Willens, S., Calhoun, P., "RADIUS Extensions", draft-ietf-radius-ext-04.txt, Internet Draft (work in progress), May 1999.

[SASL]   Myers, J., "Simple Authentication and Security Layer (SASL)", RFC 2222, October 1997.

[RFC1661]   Simpson, W., Editor, "The Point-to-Point Protocol (PPP)." STD 51, RFC 1661, July 1994.

[RFC2138]   C. Rigney, A. Rubens, W. Simpson, S. Willens, "Remote Authentication Dial In User Service (RADIUS)", RFC2138

[RFC2284]   L. Blunk, J. Vollbrecht, "PPP Extensible Authentication Protocol (EAP)", RFC 2284, March 1998.

[RFC2401] Kent, S., and R. Atkinson, "Security Architecture for the Internet Protocol", RFC 2401, November 1998.

[RFC2409] Harkins, D., and D. Carrel, "The Internet Key Exchange (IKE)", RFC 2409, November 1998.

[RFC2661] Townsley, W., Valencia, A., Rubens, A., Pall, G., Zorn, G., and Palter, B., "Layer Two Tunneling Protocol L2TP", RFC 2661, August 1999.

[XAUTH] R. Pereira, S. Beaulieu, "Extended Authentication Within ISAKMP/Oakley", September 1999, draft-ietf-ipsec-isakmp-xauth-05.txt