

Network Working Group  
Internet-Draft  
Expires: June 15, 2006

S. Kelly  
Talari Networks  
E. Rescorla  
Network Resonance  
December 12, 2005

Securing LWAPP with DTLS  
draft-kelly-capwap-lwapp-dtls-01

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/lid-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 June 15, 2006.

Copyright Notice

Copyright (C) The Internet Society (2005).

Abstract

The LWAPP protocol defines interactions between wireless termination points and wireless access controllers. Communications between these components must be secured, and the current specification provides for transport security using proprietary mechanisms which are embedded in the protocol. This document describes an alternative approach which eliminates the embedded security, and instead uses DTLS as a secure, tightly-integrated wrapper.

Internet-Draft

Securing LWAPP with DTLS

December 2005

## Table of Contents

<a href="#">1.</a>	Introduction . . . . .	<a href="#">3</a>
<a href="#">2.</a>	Inserting DTLS . . . . .	<a href="#">4</a>
<a href="#">2.1.</a>	Control/Data Channel Considerations . . . . .	<a href="#">7</a>
<a href="#">2.1.1.</a>	Separate Control/Data Channel Ports . . . . .	<a href="#">8</a>
<a href="#">2.1.2.</a>	Adding a Protocol Mux . . . . .	<a href="#">8</a>
<a href="#">3.</a>	Endpoint Authentication using DTLS . . . . .	<a href="#">8</a>
<a href="#">3.1.</a>	Authenticating with Certificates . . . . .	<a href="#">9</a>
<a href="#">3.2.</a>	Authenticating with Preshared Keys . . . . .	<a href="#">9</a>
<a href="#">4.</a>	Conclusions . . . . .	<a href="#">10</a>
<a href="#">5.</a>	Security Considerations . . . . .	<a href="#">10</a>
<a href="#">6.</a>	IANA Considerations . . . . .	<a href="#">11</a>
<a href="#">7.</a>	References . . . . .	<a href="#">11</a>
<a href="#">7.1.</a>	Normative References . . . . .	<a href="#">11</a>
<a href="#">7.2.</a>	Informative References . . . . .	<a href="#">11</a>
	Authors' Addresses . . . . .	<a href="#">13</a>
	Intellectual Property and Copyright Statements . . . . .	<a href="#">14</a>

## 1. Introduction

The Light Weight Access Point Protocol (LWAPP) provides for centralized control and management of Wireless Termination Points (WTPs) by devices referred to as Access Controllers (ACs). For more detail on this protocol and/or these components, see [[LWAPP](#)]. The CAPWAP working group is currently considering using LWAPP as the basis for a standardized AC-WTP control protocol (recommended in [[CAPWAP-EVAL](#)]).

LWAPP currently includes security elements which provide for the following capabilities:

- o Endpoint Authentication - AC and WTP are strongly authenticated using either public key certificates or shared secrets (also known as "pre-shared keys").
- o Data Confidentiality - (AC-WTP control channel) data is encrypted using the 128-bit AES-CBC algorithm.
- o Data Integrity/Origin Authenticity - an Integrity Check Value (ICV) is computed using 128-bit AES-CBC-MAC (a keyed MAC).

The current LWAPP security scheme has been through at least one security review [[LWAPP-SEC](#)], the results of which were favorable. Still, the protocol evaluation team concluded that LWAPP would benefit from replacement of its proprietary security scheme with a standardized, more widely deployed facility such as DTLS [[DTLS](#)].

Why replace LWAPP's security mechanism, when so far, security evaluations have not found it wanting? There are at least two good reasons:

- o Industry experience/review - to the chagrin of many protocol designers, it has been often demonstrated that subtle security flaws may escape the most diligent reviewer. As a result, the

cryptographic community invests significant effort in the ongoing analysis of deployed (and proposed) security mechanisms. Sometimes problems are found very quickly, but in other cases issues may not be discovered for years. Thus, security protocols and mechanisms which have been extensively deployed and analyzed are almost always preferable to those which have not.

- o Algorithm Agility – Because most cryptographic algorithms are eventually either broken outright or rendered computationally insufficient by advancing technology, it is crucial to have the ability to easily replace outdated or compromised algorithms.

Note that LWAPP, while having gone through some security review, has not yet provided the opportunity for the sort of extensive public review and analysis that TLS [[TLS11](#)] has enjoyed. Also, LWAPP provides no facility for algorithm negotiation – changing security algorithms would require a change to the protocol standard, along with firmware upgrades for both WTP and AC. This is clearly undesirable.

DTLS, on the other hand, is a standards-track effort which is based upon TLS. The underlying security-related protocol mechanisms have been successfully deployed for many years now. The TLS protocol is well-understood from an operational perspective, and with the recent specification of its datagram-based variant, is an obvious choice for meeting the security requirements of LWAPP.

## [2.](#) Inserting DTLS

Note that for the time being, only the UDP transport mechanism for LWAPP is considered. Since the evaluation document recommends eliminating layer 2 encapsulation support, it is not addressed here. Should this change, the mechanism described below in [section 2.1.2](#) could be used to partially address that case.

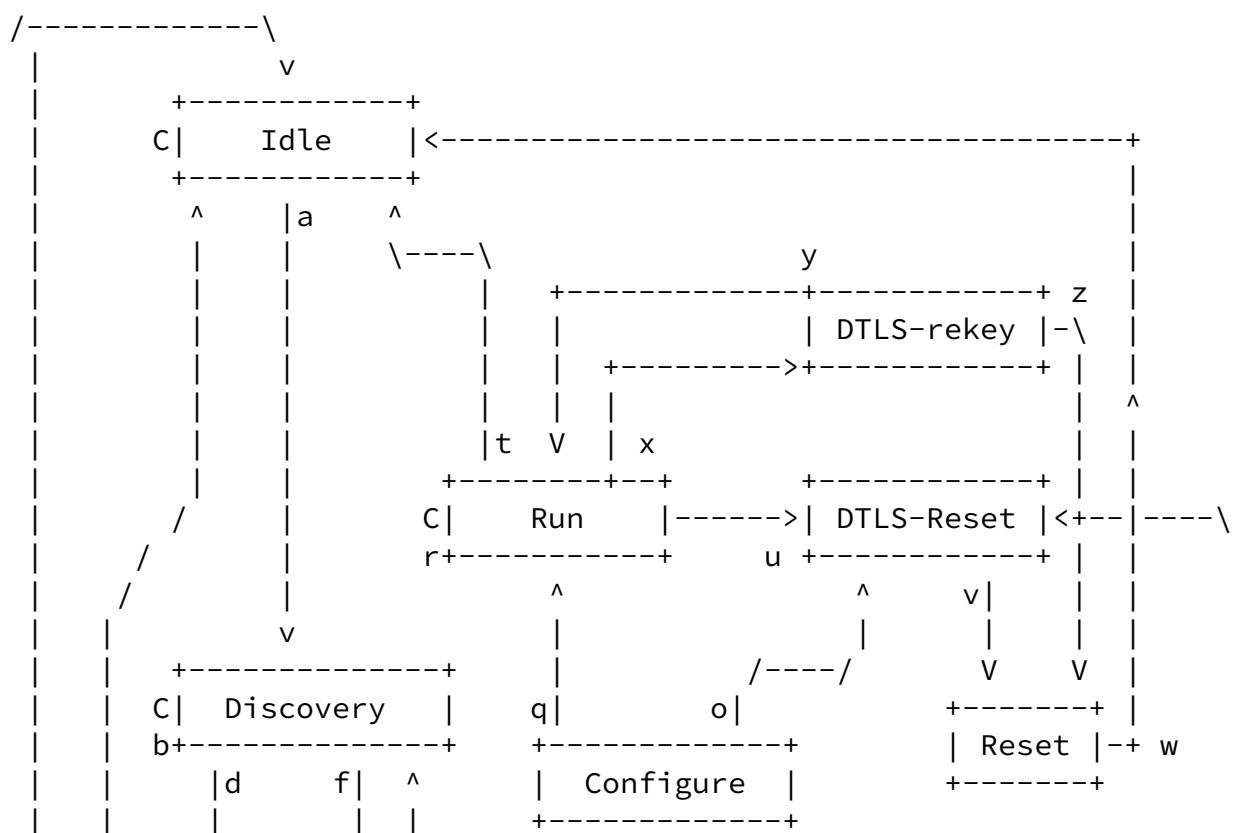
From a high level, simple replacement of the LWAPP security mechanisms with DTLS amounts to something like this:

1. Replace the JOIN phase with DTLS session establishment

- This amounts to employing DTLS as a tightly-integrated secure wrapper. Here is the resulting LWAPP state machine:

[Page 4]

December 2005



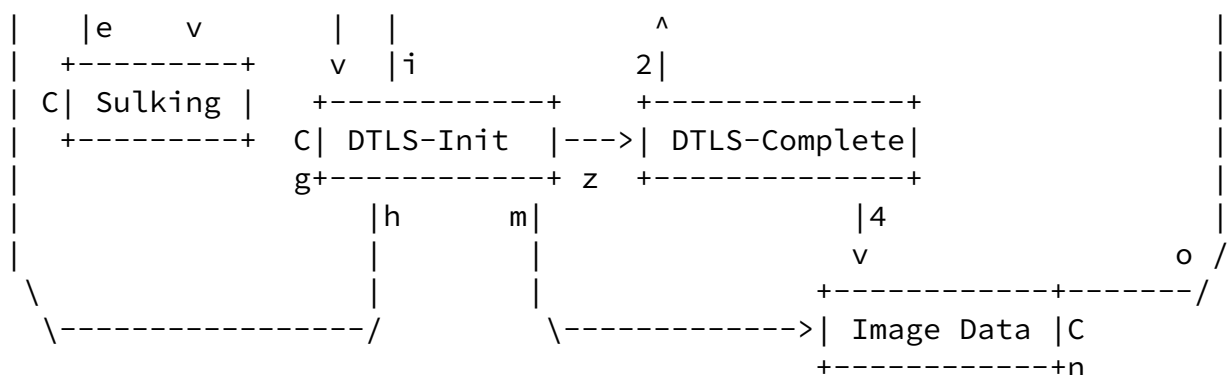


Figure 1: LWAPP State Machine w/DTLS Support

Following is a description of the associated state changes. Note that we only address those which are new:

Discovery to DTLS-Init (f): This state is used by the WTP to confirm its commitment to an AC that it wishes to be provided service, and to simultaneously establish a secure control channel.

WTP: The WTP selects the best AC based on the information it gathered during the Discovery Phase. It then initiates a DTLS connection with its preferred AC. The WTP starts the WaitJoin Timer.

AC: The AC enters this state for the given WTP upon reception of a DTLS initialization request. The AC processes the request and responds by engaging in DTLS negotiation with the WTP.

DTLS-Init to Discovery (i): This state is used to return the WTP to discovery mode when an unresponsive AC is encountered.

WTP: The WTP enters this state when the WaitJoin timer expires prior to successful completion of DTLS negotiation.

AC: This state transition is invalid.

DTLS-Init to DTLS-Complete (z): This state is used to indicate DTLS session establishment.

WTP: This state is entered when the WTP and AC complete DTLS negotiation.

AC: This state is entered when the WTP and AC complete DTLS negotiation.

Run to DTLS-Reset (u): This state is used to when the AC or WTP wish to tear down the connection.

WTP: The WTP enters this state when it wishes to initiate orderly termination of the DTLS connection; the WTP sends the a TLS Finished message.

AC: The AC enters this state upon receipt of TLS Finished message from the WTP.

Image-data to DTLS-Reset (o): This state is used to reset the connection prior to restarting the WTP after an image download.

WTP: The WTP enters this state when image download completes

AC: The AC enters this state upon receipt of TLS Finished message from the WTP.

DTLS-Reset to Reset (v): This state is used to complete DTLS session tear-down

WTP: The WTP enters this state when it has completed DTLS session cleanup, and it is ready to finish LWAPP session clean-up.

AC: The AC enters this state when it has completed DTLS session cleanup, and it is ready to finish LWAPP session clean-up.

Run to DTLS-Rekey (x): This state is used to initiate a new DTLS handshake. Either the WTP or AC may initiate the state transition. It is important to note that this might more accurately be termed a "meta-state", as the DTLS re-handshake is transparent to the LWAPP protocol, and may even be interspersed with other LWAPP control

messages.

WTP: The WTP enters this state when either (1) a rekey is required, or (2) the AC initiates a DTLS handshake.

AC: The AC enters this state when either (1) a rekey is required, or (2) the WTP initiates a DTLS handshake.

DTLS-Rekey to Reset (z): This state is used to clean up when a DTLS handshake fails.

WTP: The WTP enters this state when a DTLS handshake fails.

AC: The AC enters this state when a DTLS handshake fails.

## [2.1.](#) Control/Data Channel Considerations

Note that while this scheme seems quite simple at first glance, there is one complication. Currently, LWAPP only applies security to control channel communications, and relies upon external facilities for securing user data. In order to preserve this convention, we must be able to distinguish between control and data packets, forwarding only control packets to the DTLS engine.

This task is complicated by the fact that LWAPP currently distinguishes between control and data traffic using the 'C' bit in the LWAPP header. This is possible even on the encrypted control channel because the LWAPP header is not encrypted - in the case of the control channel, it is only authenticated:

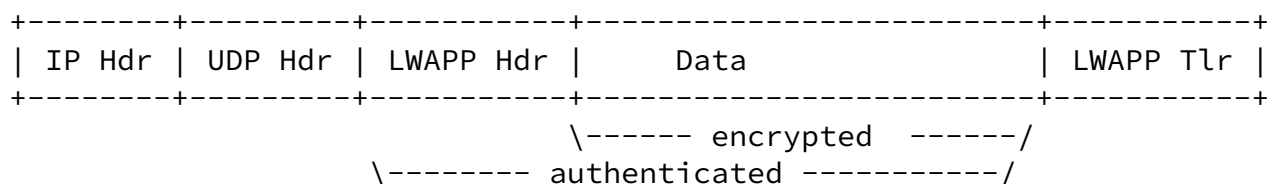


Figure 2: Current LWAPP Packet Security

DTLS, on the other hand, provides for securing the entire channel. If the LWAPP packets are encapsulated within DTLS, the LWAPP header

will be encrypted:



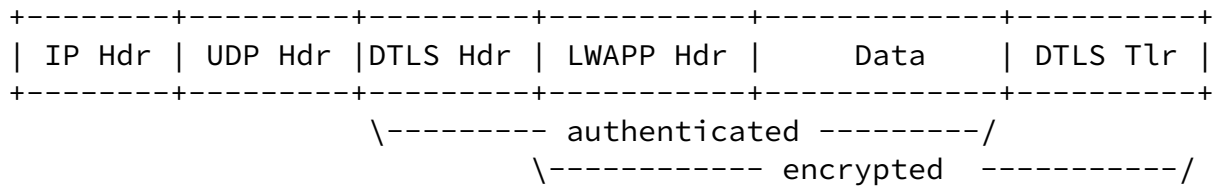


Figure 3: LWAPP+DTLS Packet Security

A direct consequence of this is that with DTLS encapsulation, we cannot distinguish between control traffic and data without first decrypting the packet – this means we must establish separate channels if we do not wish to encrypt data channel traffic. Two methods for accomplishing this are discussed below.

#### [2.1.1.1.](#) Separate Control/Data Channel Ports

The simplest solution entails using separate ports for LWAPP control and data traffic, with DTLS securing only the control channel. The control traffic could continue to utilize the current well-known LWAPP port. For the data channel, a new port could be assigned by IANA, or it could instead be specified by the AC after the DTLS session is established, providing some additional flexibility. Note that this solution will not work for layer 2 LWAPP encapsulation. However, if L2 support is to be removed from LWAPP, this point is moot.

#### [2.1.1.2.](#) Adding a Protocol Mux

An alternative solution entails adding a protocol multiplexer module between the packet input/output and the DTLS modules, and adding an additional small associated LWAPP header between the UDP header and the DTLS record layer header. While this LWAPP header need only contain a single bit to differentiate between control/data traffic, alignment concerns suggest the header would most likely be either 32 or 64 bits in length.

### [3.](#) Endpoint Authentication using DTLS

Currently, LWAPP supports authentication using either public key certificates or shared secrets (pre-shared keys). DTLS support implies no changes in this regard. Certificate-based authentication is natively supported, and support for preshared keys is currently progressing toward standardization (see [\[TLS-PSK\]](#)). Below we describe supported TLS algorithm suites for each endpoint

authentication method.

### [3.1.](#) Authenticating with Certificates

Note that only block ciphers are currently recommended for use with DTLS. To understand the reasoning behind this, see [[DTLS-DESIGN](#)]. The following algorithms MUST be supported when using certificates for LWAPP authentication:

- o TLS\_RSA\_WITH\_AES\_128\_CBC\_SHA
- o TLS\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA

The following algorithms SHOULD be supported when using certificates:

- o TLS\_DH\_RSA\_WITH\_AES\_128\_CBC\_SHA
- o TLS\_DH\_RSA\_WITH\_3DES\_EDE\_CBC\_SHA

The following algorithms MAY be supported when using certificates:

- o TLS\_RSA\_WITH\_AES\_256\_CBC\_SHA
- o TLS\_DH\_RSA\_WITH\_AES\_256\_CBC\_SHA

Certificates should be verified in the same manner as currently specified in LWAPP.

### [3.2.](#) Authenticating with Preshared Keys

Pre-shared keys present significant challenges from a security perspective, and for that reason, their use is strongly discouraged. However, [[TLS-PSK](#)] defines 3 different methods for authenticating with preshared keys:

- o PSK key exchange algorithm - simplest method, ciphersuites use only symmetric key algorithms
- o DHE\_PSK key exchange algorithm - use a PSK to authenticate a Diffie-Hellman exchange. These ciphersuites give some additional protection against dictionary attacks and also provide Perfect Forward Secrecy (PFS).
- o RSA\_PSK key exchange algorithm - use RSA and certificates to authenticate the server, in addition to using a PSK. Not susceptible to passive attacks.

The first approach (PSK) is susceptible to passive dictionary

attacks; hence, that method MUST NOT be used. If support for pre-shared keys is desired, then DHE\_PSK MUST be supported, and RSA\_PSK MAY be supported.

The following cryptographic algorithms MUST be supported when using preshared keys:

- o TLS\_PSK\_WITH\_AES\_128\_CBC\_SHA
- o TLS\_PSK\_WITH\_3DES\_EDE\_CBC\_SHA

The following algorithms SHOULD be supported when using preshared keys:

- o TLS\_PSK\_WITH\_AES\_256\_CBC\_SHA

The following algorithms MAY be supported when using preshared keys:

- o TLS\_RSA\_PSK\_WITH\_AES\_128\_CBC\_SHA
- o TLS\_RSA\_PSK\_WITH\_AES\_256\_CBC\_SHA
- o TLS\_RSA\_PSK\_WITH\_3DES\_EDE\_CBC\_SHA

#### [4.](#) Conclusions

DTLS represents a strong replacement candidate for the existing LWAPP security scheme. In addition to being a known quantity which has received and will continue to receive a healthy dose of ongoing analysis and review from the cryptographic community, it supports all required LWAPP security functionality, and also provides for algorithm agility should the need arise. Further, its negotiation capability provides for a measure of implementation flexibility not possible with the current LWAPP scheme.

While it is not a drop-in replacement, it requires a reasonably bounded amount of change to the existing state machine and packet formats. As noted, since DTLS does not provide for unequal

encryption vs authentication lengths within a packet, it requires adding either a secondary data port or a short demux header.

## [5.](#) Security Considerations

The security of LWAPP over DTLS is completely dependent on the security of DTLS. Any flaws in DTLS compromise the security of LWAPP. In particular, it is critical that the communicating parties

verify their peer's credentials. In the case of pre-shared keys, this happens automatically via the key. In the case of certificates, the parties must check the peer's certificate. The appropriate checks are described in the current LWAPP document

The use of parallel protected and unprotected channels deserves special consideration, but does not create a threat. There are two potential concerns: attempting to convert protected data into unprotected data and attempting to convert un-protected data into protected data. The use of the MAC makes it impossible for the attacker to forge protected records. The attacker can easily remove protected records from the stream (this is a consequence of unreliability), though not undetectably so. If a non-encrypted cipher suite is in use, the attacker can turn such a record into an un-protected record. However, this attack is really no different from simple injection into the unprotected stream.

## [6.](#) IANA Considerations

Should a separate UDP port for data channel communications be the selected demultiplexing mechanism, a port must be assigned for this purpose. Should a demultiplexing header be used instead, there may be additional IANA requirements (we'll cross that bridge if we come to it).

## [7.](#) References

### [7.1.](#) Normative References

[DTLS] Rescorla et al, E., "Datagram Transport Layer Security",

June 2004.

[LWAPP] Calhoun et al, P., "Light Weight Access Point Protocol", June 2005, <<http://www.ietf.org>>.

[TLS-PSK] Eronen et al, P., "Pre-Shared Key Ciphersuites for Transport Layer Security (TLS)", June 2005.

## 7.2. Informative References

[CAPWAP-EVAL]

Lohrer et al, D., "Evaluation of Candidate CAPWAP Protocols", August 2005, <<http://www.ietf.org>>.

[DTLS-DESIGN]

Modadugu et al, N., "The Design and Implementation of

Kelly & Rescorla

Expires June 15, 2006

[Page 11]

---

Internet-Draft

Securing LWAPP with DTLS

December 2005

Datagram TLS", Feb 2004.

[LWAPP-SEC]

Clancy, C., "Security Review of the Light Weight Access Point Protocol", May 2005.

[TLS11]

Dierks et al, T., "The TLS Protocol Version 1.1", June 2005.

Authors' Addresses

Scott G. Kelly  
Talari Networks  
150 W. Iowa Ave Ste 208  
Sunnyvale, CA 94086  
US

Email: [scott@hyperthought.com](mailto:scott@hyperthought.com)

Eric Rescorla  
Network Resonance  
2483 El Camino Real, #212  
Palo Alto, CA 94303  
US

Email: [ekr@networkresonance.com](mailto:ekr@networkresonance.com)

#### Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in [BCP 78](#) and [BCP 79](#).

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at <http://www.ietf.org/ipr>.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at [ietf-ipr@ietf.org](mailto:ietf-ipr@ietf.org).

#### Disclaimer of Validity

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

#### Copyright Statement

Copyright (C) The Internet Society (2005). This document is subject to the rights, licenses and restrictions contained in [BCP 78](#), and except as set forth therein, the authors retain all their rights.

#### Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.