Network Working Group
INTERNET-DRAFT
Category: Proposed Standard
<draft-dekok-radext-dtls-00.txt>
26 February 2007

DeKok, Alan
FreeRADIUS

DTLS as a Transport Layer for RADIUS

Copyright Notice

Abstract

The RADIUS protocol [RFC2865] has limited support for protocol level authentication and encryption. RADIUS packets contain attributes sent "in the clear", although some attributes can have "hidden" content. Packets may be replayed verbatim by an attacker, and the client-server authentication could be better. This document proposes DTLS as the solution to these problems, and details how this proposal is backwards-compatible wih existing RADIUS solutions.

Table of Contents

1. Introduction

RADIUS security is bad. TLS is good. TCP is often bad as a transport protocol for AAA. [RFC3539]. DTLS [RFC4347] seems to be a good idea.

Note that we choose DTLS rather than invent our own crypto protocols, for the following reasons:

- Cryptography is hard.
- Re-inventing the wheel is bad.
- DTLS exists, is implemented, and deployed.
- DTLS appears to fulfill all of the RADEXT crypto-agility requirements
- crypto updates to TLS can be done independently of RADIUS, and RADIUS will gain the benefits.
- DTLS is just a wrapper on RADIUS, and involves minimal changes to existing implementations.

1.1. Terminology

Network Access Server (NAS)
: The device providing access to the network. Also known as the Authenticator (IEEE 802.1X or EAP terminology) or RADIUS client.

Home Server
: A RADIUS server that is authoritative for user authorization and authentication.

Proxy Server
: A RADIUS server that acts as a Home Server to the NAS, but in turn proxies the request to another Proxy Server, or to a Home Server.

silently discard
: This means the implementation discards the packet without further processing. The implementation SHOULD provide the capability of logging the error, including the contents of the silently discarded packet, and SHOULD record the event in a statistics counter.

1.2. Requirements Language

In this document, several words are used to signify the requirements of the specification. 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].

2. DTLS Negotiation.

2.1. NAS requirements

When a NAS desires to initiate a DTLS session with a RADIUS server, it sends an Access-Request packet containing Service-Type = Start-DTLS. The request packet has no User-Name or Password attribute, but MUST have a Message-Authenticator attribute.

Note that the lack of User-Name and User-Password ensures that servers not supporting DTLS will will respond with an Access-Reject. [RFC2865] permits Access-Request packets to not contain a User-Name.

The lack of a response within a time period (we suggest 5 seconds), or an Access-Reject MUST be interpreted by the NAS as an indication that the server does not support DTLS. In that case, the NAS MAY revert to normal RADIUS, although this is subject to "bidding down" attacks.

The NAS SHOULD be configurable to require DTLS on a per-server basis. If a server is marked as requiring DTLS, the NAS MUST use DTLS to transport RADIUS traffic. The NAS MUST NOT send normal RADIUS traffic to servers marked as requiring DTLS. If the server is unresponsive, or rejects the DTLS request, the NAS MUST consider the server to be "dead".

2.2. Server requirements

When server receives an Access-Request with Service-Type = Start-DTLS, it SHOULD respond with an Access-Request, ack'ing the Service-Type = Start-DTLS. Later packets are handled as per the DTLS specification. [RFC4347]

A server SHOULD be configurable to require DLTS on a per-NAS basis. If a NAS is marked as requiring DTLS, the server MUST respond to all normal RADIUS Access-Request packets with an Access-Reject.

2.3. Cryptographic Negotiations

Servers and NASes MUST support a minimum cipher suite ZZZ.

2.4. Accounting-Requests

Similar stuff here... Accounting-Request packets [RFC2866] contain Service-Type = Start-DTLS, and maybe Acct-Status-Type, but not Acct-Session-Id. Accounting-Response packets ack it. Note that

Accounting-Request packets MUST contain a nonce, and SHOULD contain Event-Timestamp, in order to prevent replay attacks.

Note that this breaks the requirements of [RFC2866] Section 5.13. It may be possible to add Acct-Session-Id, etc. with "well known" values, in order to satisfy the requirements of [RFC2866] while still not affecting this proposal.

2.5. CoA and Disconnect-Request.

It looks to be pretty much the same here. [RFC3576]

3. Issues and Benefits

DTLS imposes ordering of request (Section 3.2.2), which is not currently required in RADIUS. This may be beneficial, however.

DTLS has replay protection, which RADIUS does not. Encryption means that certain attacks requiring access to the Request Authenticator and User-Password attribute are no longer possible.

DTLS SHOULD NOT negotiate mechanisms that yield integrity protection without encryption. The use of "well-known" shared secrets means that attackers may be able to observe the traffic and decode user passwords.

Packet integrity means that the whole packet can be authenticated using a stronger mechanism than the existing MD5 hacks.

Certificates could be used in addition to, or along with a default shared secret. NASes could be configured with a local site root certificate, and automatically connect to any number of local RADIUS servers for load balancing and failover, with minimal administrator interaction.

Backwards compatibility is implemented by bidding down to RADIUS, where that is permitted by NAS/server configuration.

DTLS is connection-based, so it only affects a local client to server conversation. It does not affect other clients known by that server, or other servers known by that client, or requests that are proxied. That is, if a client and server support DTLS, nothing else in the larger network supporting RADIUS needs to change.

DTLS works through NAT gateways, so long as they don't perform inspection and/or validation of the packets (such as is done by an application-aware proxy or load balancer).

Even if RADIUS security (MD5, etc.) is completely compromised, certificate-based authentication can be performed. All that is required is a request/response packet to negotiate DTLS. Those packets contain no secret information, so they don't have to be authenticated, but maybe rate-limited. i.e. If we were doing RADIUS today, we might just start with DTLS negotiation, and skip the Service-Type = Start-DTLS stage.

Attackers MAY DoS a DTLS-aware server by repeatedly requesting DTLS negotiations. Servers that implement DTLS SHOULD NOT initiate DTLS if the client (src IP/port) is currently using normal RADIUS. Instead, those requests SHOULD be silently dropped. That is, clients should signal DTLS support with an Accounting-Request containing Acct-Status-Type = On.

Packets with Service-Type = Start-DTLS MUST NOT be proxied. Backwards compatibility here is helped with the lack of a User-Name, which is the source of most proxying decisions. Proxy load balancers may be affected, if they are application-level (as opposed to simple UDP load balancers), and are unaware of DTLS. In this situation, home servers in the load-balanced configuration SHOULD respond to requests for DTLS with Access-Reject. Or, the proxy load balancer should be upgraded to be DTLS aware.

The RADIUS server must maintain transport-layer state for DTLS in addition to what it does now. Since many RADIUS servers already maintain TLS state for EAP sessions, we believe that this requirement is not onerous.

The RADIUS Identifier field is only 8 bits, so if more than 256 packets are outstanding to a server, a NAS must start another DTLS session.

3.1. Implementation notes

RADSEC (Radiator) has implemented RADIUS over TLS over TCP, and it has been deployed for a few years. So there do not appear to be any problems with implementing ot deploying RADIUS + TLS.

RADSEC has also been allocated a port (2083) for RADIUS over TLS over TCP. We note that the UDP side of the port is currently unused. We could therefore use port 2083 as RADIUS + DTLS, and skip the Service-Type = Start-DTLS portion of the conversation.

4. Diameter compatibility.

Packets with Service-Type = Start-DTLS MUST NOT be proxied across a RADIUS to Diameter, or Diameter to RADIUS gateway. Packets with

Service-Type = Start-DTLS MUST NOT appear in a Diameter packet.

Other than that, this proposal is just RADIUS, with a wrapper layer between a RADIUS client and server. Diameter is not affected, and no new RADIUS attributes or commands are allocated.

5. IANA Considerations

A new value for Service-Type (Start-DTLS) has to be allocated.

New ports may be allocated for RADIUS + DTLS.

6. Security Considerations

The entire content of this proposal is devoted to discussing security considerations related to RADIUS. No additional comments are noted here.

7. References

7.1. Informative references

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

7.2. Normative references

[RFC2865] Rigney, C., Willens, S., Rubens, A. and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, June 2000.

[RFC2866] Rigney, C., "RADIUS Accounting", RFC 2866, June 2000.

[RFC3539] Aboba, B., Wood, J., "Authentication, Authorization, and Accounting (AAA) Transport Profile", RFC 3539, June 2003.

[RFC3576] Chiba, M., Dommety, G., Eklund, M., Mitton, D. and B. Aboba, "Dynamic Authorization Extensions to Remote Authentication Dial In User Service (RADIUS)", RFC 3576, July 2003.

[RFC4347] Rescorla E., and Modadugu, N., "Datagram Transport Layer Security", RFC 4347, April 2006.

Acknowledgments

None as yet.

Authors' Addresses

Alan DeKok
The FreeRADIUS Server Project

Email: aland@freeradius.org

Intellectual Property Statement

Disclaimer of Validity

Full Copyright Statement

Acknowledgment

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