Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 20, 2015

P. Saint-Andre
&yet
M. Miller
Cisco Systems, Inc.
P. Hancke
&yet

February 16, 2015

Domain Name Associations (DNA) in the Extensible Messaging and Presence Protocol (XMPP)

draft-ietf-xmpp-dna-09

Abstract

This document improves the security of the Extensible Messaging and Presence Protocol (XMPP) in two ways. First, it specifies how to establish a strong association between a domain name and an XML stream, using the concept of "prooftypes". Second, it describes how to securely delegate a service domain name (e.g., example.com) to a target server host name (e.g., hosting.example.net), which is especially important in multi-tenanted environments where the same target server hosts a large number of domains.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

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."

This Internet-Draft will expire on August 20, 2015.

Table of Contents

1. Introduction

In systems that use the Extensible Messaging and Presence Protocol (XMPP) [RFC6120], it is important to establish a strong association between the DNS domain name of an XMPP service (e.g., example.com) and the XML stream that a client or peer server initiates with that service. In other words, the client or peer server needs to verify the identity of the server to which it connects. Additionally, servers need to verify incoming connections from other servers.

To date, such verification has been established based on information obtained from the Domain Name System (DNS), the Public Key Infrastructure (PKI), or similar sources. In relation to such associations, this document does the following:

1. Generalizes the model currently in use so that additional prooftypes can be defined if needed.

2. Provides a basis for modernizing some prooftypes to reflect progress in underlying technologies such as DNS Security [RFC4033].

3. Describes the flow of operations for establishing a domain name association (DNA).

This document also provides guidelines for secure delegation of a service domain name (e.g., example.com) to a target server host name (e.g., hosting.example.net). The need for secure delegation arises because the process for resolving the domain name of an XMPP service into the IP address at which an XML stream will be negotiated (see [RFC6120]) can involve delegation of a service domain name to a target server host name using technologies such as DNS SRV records [RFC2782]. A more detailed description of the delegation problem can be found in [I-D.ietf-xmpp-posh]. If such delegation is not done in a secure manner, then the domain name association cannot be verified.

2. Terminology

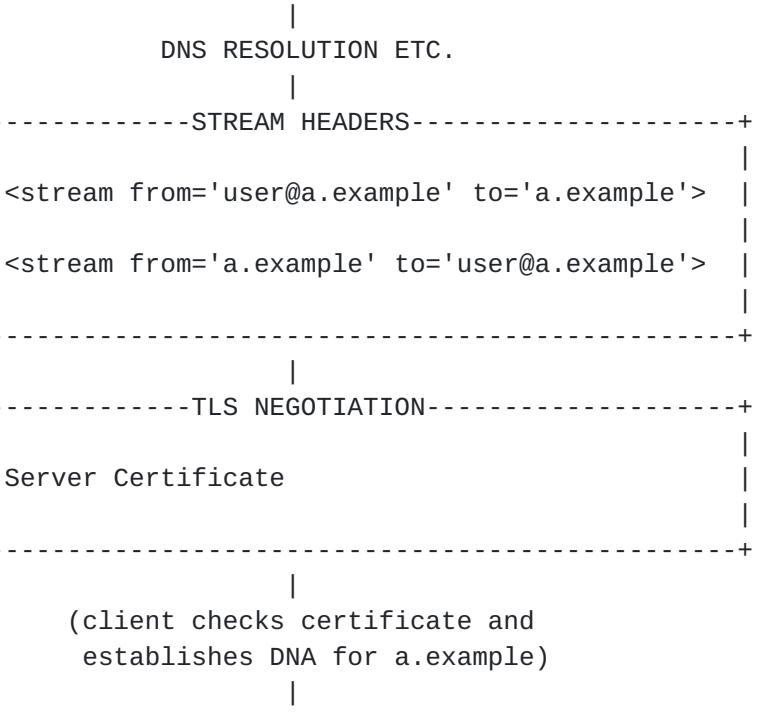
This document inherits XMPP terminology from [RFC6120] and [XEP-0220], DNS terminology from [RFC1034], [RFC1035], [RFC2782] and [RFC4033], and security terminology from [RFC4949] and [RFC5280]. The terms "reference identity", and "presented identity" are used as defined in the "CertID" specification [RFC6125]. For the sake of consistency with [I-D.ietf-dane-srv], this document uses the terms "service domain name" and "target server host name" to refer to the same entities identified by the terms "source domain" and "derived domain" from [RFC6125].

3. Client-to-Server (C2S) DNA

The client-to-server case is much simpler than the server-to-server case because the client does not assert a domain name, which means verification happens in only one direction. Therefore we describe this case first to help the reader understand domain name associations in XMPP.

3.1. C2S Flow

The following flow chart illustrates the protocol flow for establishing a domain name association for an XML stream from a client (C) to a server (S) using the standard PKIX prooftype specified in [RFC6120].



3.2. C2S Description

The simplified order of events (see [RFC6120] for details) in establishing an XML stream from a client (user@a.exmaple) to a server (a.example) is as follows:

1. The client resolves via DNS the service _xmpp-client._tcp.a.example.

2. The client opens a TCP connection to the resolved IP address.

3. The client sends an initial stream header to the server:

   <stream:stream from='user@a.example' to='a.example'>

4. The server sends a response stream header to the client, asserting that it is a.example:

   <stream:stream from='a.example' to='user@a.example'>

5. The parties attempt TLS negotiation, during which the XMPP server (acting as a TLS server) presents a PKIX certificate proving that it is a.example.

6. The client checks the PKIX certificate that the server provided; if the proof is consistent with the XMPP profile of the matching rules from [RFC6125] and the certificate is otherwise valid according to [RFC5280], the client accepts that there is a strong domain name association between its stream to the target server and the DNS domain name of the XMPP service.

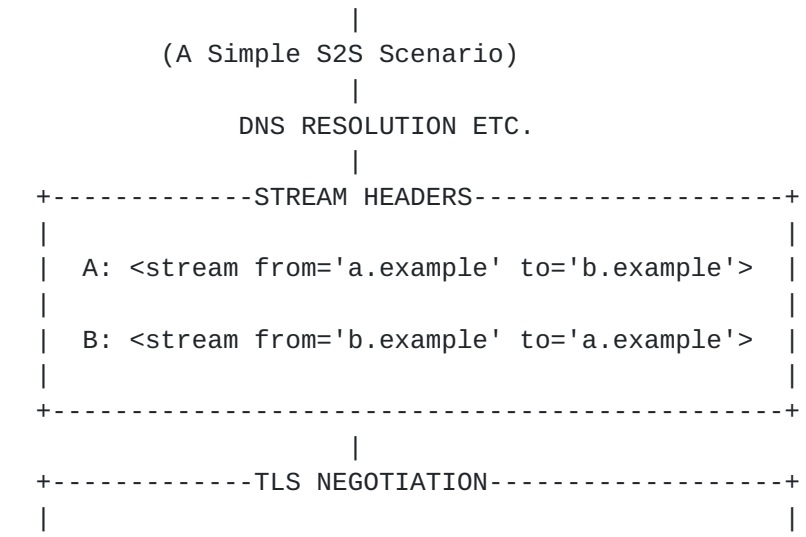
The certificate that the server presents might not be trusted by the client. As one example, the server might be hosting multiple domains and secure delegation as described in Section 6 is necessary. As another example, the server might present a self-signed certificate, which requires the client to apply either the fallback process described in section 6.6.4 of [RFC6125] or prompt the user to accept an unauthenticated connection as described in [I-D.ietf-uta-xmpp].

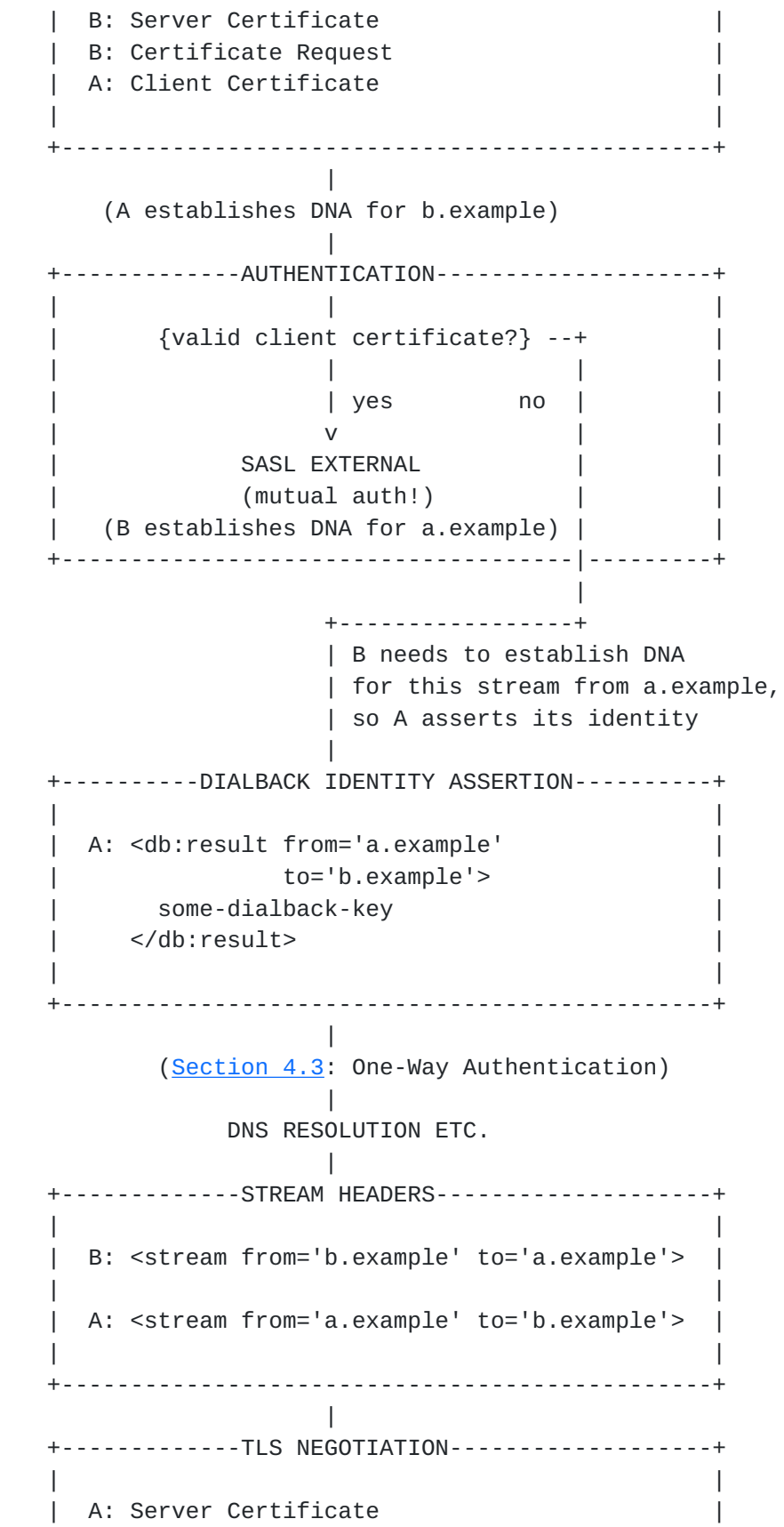
4. Server-to-Server (S2S) DNA

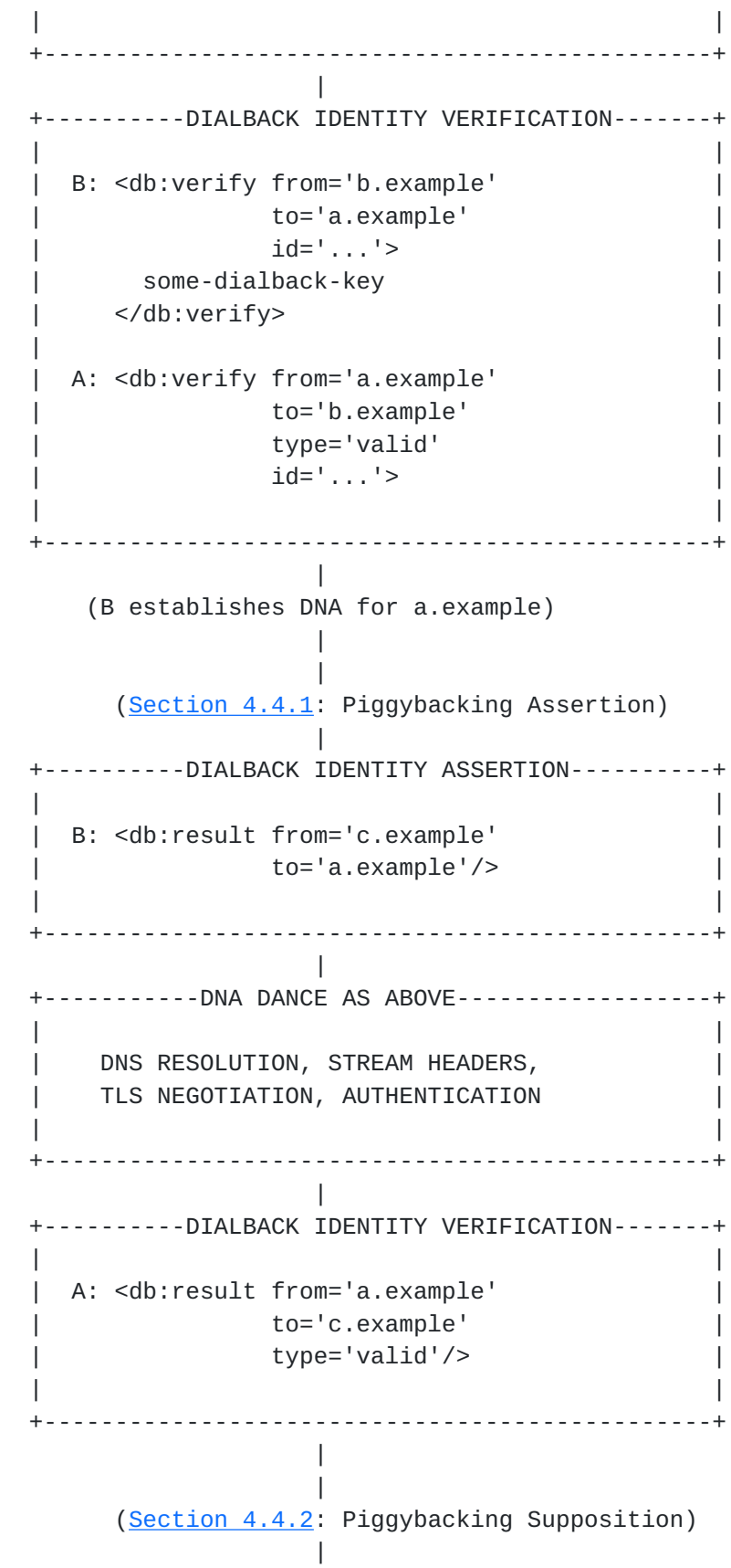
The server-to-server case is significantly more complex than the client-to-server case, and involves checking of domain name associations in both directions along with other "wrinkles" described in the following sections.

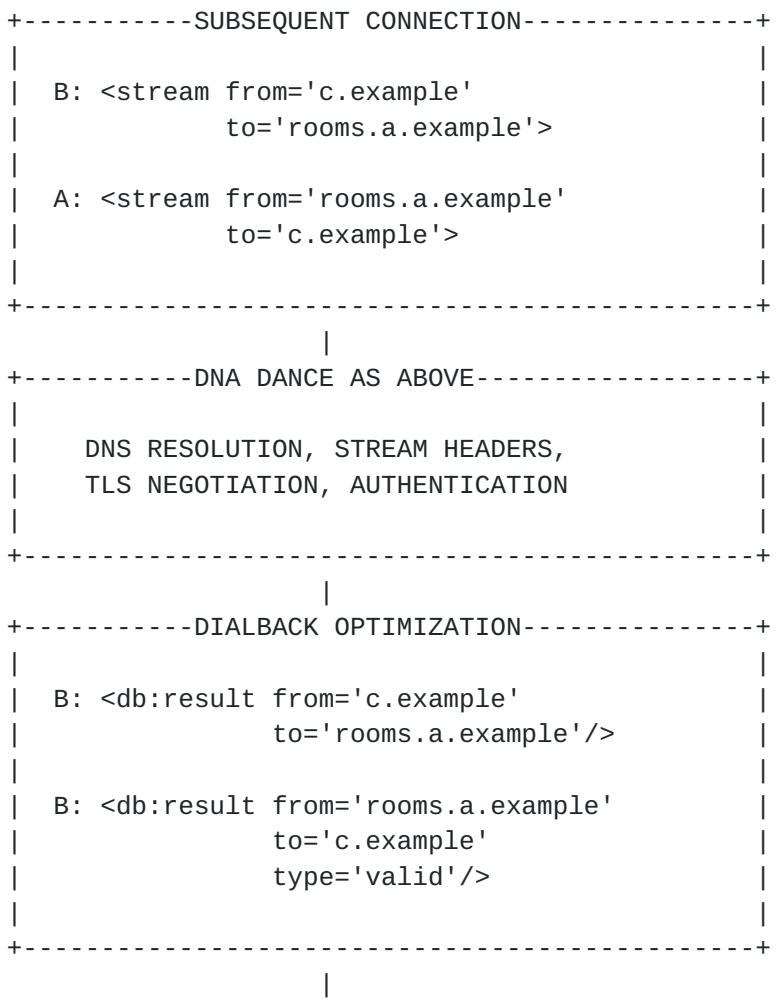
4.1. S2S Flow

The following flow chart illustrates the protocol flow for establishing domain name associations between Server A (the initiating entity) and Server B (the receiving entity), as described in the remaining sections of this document.




B: Server Certificate
B: Certificate Request
A: Client Certificate
(A establishes DNA for b.example)
AUTHENTICATION
{valid client certificate?}
yes
no
SASL EXTERNAL
(mutual auth!)
(B establishes DNA for a.example)
B needs to establish DNA
for this stream from a.example,
so A asserts its identity
DIALBACK IDENTITY ASSERTION
A: <db:result from='a.example'
to='b.example'>
some-dialback-key
</db:result>
(Section 4.3: One-Way Authentication)
DNS RESOLUTION ETC.
STREAM HEADERS
B: <stream from='b.example' to='a.example'>
A: <stream from='a.example' to='b.example'>
TLS NEGOTIATION
A: Server Certificate


DIALBACK IDENTITY VERIFICATION
B: <db:verify from='b.example'
to='a.example'
id='...'>
some-dialback-key
</db:verify>
A: <db:verify from='a.example'
to='b.example'
type='valid'
id='...'>
(B establishes DNA for a.example)
(Section 4.4.1: Piggybacking Assertion)
DIALBACK IDENTITY ASSERTION
B: <db:result from='c.example'
to='a.example'/>
DNA DANCE AS ABOVE
DNS RESOLUTION, STREAM HEADERS,
TLS NEGOTIATION, AUTHENTICATION
DIALBACK IDENTITY VERIFICATION
A: <db:result from='a.example'
to='c.example'
type='valid'/>
(Section 4.4.2: Piggybacking Supposition)



4.2. A Simple S2S Scenario

To illustrate the problem, consider the simplified order of events (see [RFC6120] for details) in establishing an XML stream between Server A (a.example) and Server B (b.example):

1. Server A resolves via DNS the service _xmpp-server._tcp.b.example.

2. Server A opens a TCP connection to the resolved IP address.

3. Server A sends an initial stream header to Server B, asserting that it is a.example:

   <stream:stream from='a.example' to='b.example'>

4. Server B sends a response stream header to Server A, asserting that it is b.example:

<stream:stream from='b.example' to='a.example'>

5. The servers attempt TLS negotiation, during which Server B (acting as a TLS server) presents a PKIX certificate proving that it is b.example and Server A (acting as a TLS client) presents a PKIX certificate proving that it is a.example.

6. Server A checks the PKIX certificate that Server B provided and Server B checks the PKIX certificate that Server A provided; if these proofs are consistent with the XMPP profile of the matching rules from [RFC6125] and is otherwise valid according to [RFC5280], each server accepts that there is a strong domain name association between its stream to the other party and the DNS domain name of the other party.

Several simplifying assumptions underlie the happy scenario just outlined:

- The PKIX certificate presented by Server B during TLS negotiation is trusted by Server A and matches the expected identity.
- The PKIX certificate presented by Server A during TLS negotiation is trusted by Server B, which enables the parties to complete mutual authentication.
- There are no additional domains associated with Server A and Server B (say, a subdomain rooms.a.example on Server A or a second domain c.example on Server B).
- The server administrators are able to obtain PKIX certificates issued by a widely-accepted certificate authority (CA) in the first place.
- The server administrators are running their own XMPP servers, rather than using hosting services.

Let's consider each of these "wrinkles" in turn. Since Server A is acting as a S2S client the behaviour is same as in the C2S case described in Section 3.2.

4.3. One-Way Authentication

If the PKIX certificate presented by Server A during TLS negotiation is not trusted by Server B, Server B is unable to mutually authenticate Server A. Therefore, Server B needs to verify the asserted identity of Server A by other means.

1. Server A asserts it is a.example using the Server Dialback protocol:

   <db:result from='a.example' to='b.example' id='...'>some-dialback-key</db:result>

2. Server B resolves via DNS the service _xmpp-server._tcp.a.example.

3. Server B opens a TCP connection to the resolved IP address.

4. Server B sends an initial stream header to Server A, asserting that it is b.example:

   <stream:stream from='b.example' to='a.example'>

5. Server A sends a response stream header to Server B, asserting that it is a.example:

   <stream:stream from='a.example' to='b.example'>

6. The servers attempt TLS negotiation, during which Server A (acting as a TLS server) presents a PKIX certificate proving that it is a.example.

7. Server B checks the PKIX certificate that Server A provided. This might be the same certificate presented by Server A as a client certificate in the initial connection. Even if this certificate is not signed by a trusted CA (for example it could be self-signed) Server B can verify that there is an association between the incoming connection and the domain name a.example. Note that this may be insecure unless DNSSEC [RFC4033] is used.

8. If the certificate provided by Server A is different from the one presented in the initial connection, Server B proceeds with Server Dialback in order to establish the domain name association. In order to do this it sends a request for verification as described in [XEP-0220]:

   <db:verify from='b.example' to='a.example' id='...'>some-dialback-key</db:verify>

9. Server A responds to this:

   <db:verify from='a.example' to='b.example' id='...' type='valid/>

   allowing Server B to establish the domain name association.

At this point the servers are using two TCP connections instead of one, which is somewhat wasteful. However, there are ways to tie the authentication achieved on the second TCP connection to the first TCP connection; see [XEP-0288] for further discussion.

4.4. Piggybacking

4.4.1. Assertion

Consider the common scenario in which Server B hosts not only b.example but also a second domain c.example (often called a "multi-tenanted" environment). If a user of Server B associated with c.example wishes to communicate with a friend at a.example, Server B needs to send XMPP stanzas from the domain c.example rather than b.example. Although Server B could open a new TCP connection and negotiate new XML streams for the domain pair of c.example and a.example, that too is wasteful (especially if Server B hosts a large number of domains). Server B already has a connection to a.example, so how can it assert that it would like to add a new domain pair to the existing connection?

The traditional method for doing so is the Server Dialback protocol, first specified in (the now obsolete) [RFC3920] and since moved to [XEP-0220]. Here, Server B can send a <db:result/> element for the new domain pair over the existing stream.

```
<db:result from='c.example' to='a.example'>
  some-dialback-key
</db:result>
```

This element functions as Server B's assertion that it is (also) c.example, and thus is functionally equivalent to the 'from' address of an initial stream header as previously described.

In response to this assertion, Server A needs to obtain some kind of proof that Server B really is also c.example. If the certificate presented by Server B is also valid for c.example then no further action is necessary. However, if not then Server A needs to do a bit more work. Specifically, Server A can pursue the same strategy it used before:

1. Server A resolves via DNS the service _xmpp-server._tcp.c.example.

2. Server A opens a TCP connection to the resolved IP address (which might be the same IP address as for b.example).

3. Server A sends an initial stream header to Server B, asserting that it is a.example:

   <stream:stream from='a.example' to='c.example'>

4. Server B sends a response stream header to Server A, asserting that it is c.example:

   <stream:stream from='c.example' to='a.example'>

5. The servers attempt TLS negotiation, during which Server B (acting as a TLS server) presents a PKIX certificate proving that it is c.example.

6. At this point, Server A needs to establish that, despite different certificates, c.example is associated with the origin of the request. This is done using Server Dialback [XEP-0220]:

   <db:verify from='a.example' to='c.example' id='...'>some-dialback-key</db:verify>

7. Server B responds to this:

   <db:verify from='c.example' to='a.example' id='...' type='valid/>

   allowing Server A to establish the domain name association.

Now that Server A accepts the domain name association, it informs Server B of that fact:

   <db:result from='a.example' to='c.example' type='valid'/>

The parties can then terminate the second connection, since it was used only for Server A to associate a stream with the domain name c.example (the dialback key links the original stream to the new association).

4.4.2. Supposition

Piggybacking can also occur in the other direction. Consider the common scenario in which Server A provides XMPP services not only for a.example but also for a subdomain such as a groupchat service (e.g., Multi-User Chat [XEP-0045]) at rooms.a.example. If a user from c.example at Server B wishes to join a room on the groupchat sevice, Server B needs to send XMPP stanzas from the domain c.example to the domain rooms.a.example rather than a.example. First, Server B needs to determine whether it can piggyback the domain rooms.a.example on the connection to a.example:

1. Server B resolves vua DNS the service _xmpp-server._tcp.rooms.a.example.

2. Server B determines this resolves to an IP address and port that it is already connected to.

3. Server B determines that the PKIX certificate for that active connection would also be valid for the rooms.a.example domain and that Server A has announced support for dialback errors.

Server B sends a dialback key to Server A over the existing connection.

```
<db:result from='c.example' to='rooms.a.example'>
  some-dialback-key
</db:result>
```

Server A then informs Server B that it accepts the domain name association:

```
<db:result from='rooms.a.example' to='c.example' type='valid'/>
```

5. Alternative Prooftypes

The foregoing protocol flows assumed that domain name associations were proved using the PKI prooftype specified in [RFC6120]: that is, the server's proof consists of a PKIX certificate that is checked according to the XMPP profile [RFC6120] of the matching rules from [RFC6125] (and the overall validation rules from [RFC5280]), the client's verification material is obtained out of band in the form of a trusted root, and secure DNS is not necessary.

However, sometimes XMPP server administrators are unable or unwilling to obtain valid PKIX certificates for all of the domains they host at their servers. For example:

- In order to issue a PKIX certificate, a CA might try to send email messages to authoritative mailbox names [RFC2142], but the administrator of a subsidiary service such as im.cs.podunk.example cannot receive email sent to mailto:hostmaster@podunk.example.

- A hosting provider such as hosting.example.net might not want to take on the liability of holding the certificate and private key for a tenant such as example.com (or the tenant might not want the hosting provider to hold its certificate and private key).

- Even if PKIX certificates for each tenant can be obtained, the management of so many certificates can introduce a large administrative load.

(Additional discussion can be found in [I-D.ietf-xmpp-posh].)

In these circumstances, prooftypes other than PKIX are desirable or necessary. As described below, two alternatives have been defined so far: DNS-Based Authentication of Named Entities (DANE) and PKIX Over Secure HTTP (POSH).

5.1. DANE

The DANE prooftype can be defined as follows:

1. The server's proof consists of either a service certificate or domain-issued certificate (TLSA usage PKIX-EE or DANE-EE, see [RFC6698] and [RFC7218]).

2. The proof is checked by verifying an exact match or a hash of either the SubjectPublicKeyInfo or the full certificate.

3. The client's verification material is obtained via secure DNS as described in [I-D.ietf-dane-srv].

4. Secure DNS is necessary in order to effectively establish an alternative chain of trust from the service certificate or domain-issued certificate to the DNS root.

The DANE prooftype makes use of the DNS-Based Authentication of Named Entities [RFC6698], specifically the use of DANE with DNS SRV records [I-D.ietf-dane-srv]. For XMPP purposes, the following rules apply:

- If there is no SRV resource record, pursue the fallback methods described in [RFC6120].

- Use the 'to' address of the initial stream header to determine the domain name of the TLS client's reference identifier (since use of the Server Name Indication extension (TLS SNI) [RFC6066] is purely discretionary in XMPP, as mentioned in [RFC6120]).

5.2. POSH

The POSH prooftype can be defined as follows:

1. The server's proof consists of a PKIX certificate.

2. The proof is checked according to the rules from [RFC6120] and [RFC6125].

3. The client's verification material is obtained by retrieving a hash of the PKIX certificate over HTTPS at a well-known URI [RFC5785].

4. Secure DNS is not necessary since the HTTPS retrieval mechanism relies on the chain of trust from the public key infrastructure.

POSH is defined in [I-D.ietf-xmpp-posh]. For XMPP purposes, the following rules apply:

- If no verification materials are found via POSH, pursue the fallback methods described in [RFC6120].

- Use the 'to' address of the initial stream header to determine the domain name of the TLS client's reference identifier (since use of the Server Name Indication extension (TLS SNI) [RFC6066] is purely discretionary in XMPP, as mentioned in [RFC6120]).

The well-known URIs [RFC5785] to be used for POSH are:

- "/.well-known/posh._xmpp-client._tcp.json" for client-to-server connections

- "/.well-known/posh._xmpp-server._tcp.json" for server-to-server connections

6. Secure Delegation and Multi-Tenancy

One common method for deploying XMPP services is multi-tenancy: e.g., XMPP services for the service domain example.com are actually hosted at the target server hosting.example.net. Such an arrangement is relatively convenient in XMPP given the use of DNS SRV records [RFC2782], such as the following delegation from example.com to hosting.example.net:

```
_xmpp-server._tcp.example.com. 0 IN SRV 0 0 5269 hosting.example.net
```

Secure connections with multi-tenancy can work using the PKIX prooftype on a small scale if the provider itself wishes to host several domains (e.g., related domains such as jabber-de.example and jabber-ch.example). However, in practice the security of multi-tenancy has been found to be unwieldy when the provider hosts large numbers of XMPP services on behalf of multiple tenants (see [I-D.ietf-xmpp-posh] for a detailed description). Typically there are two main reasons for this state of affairs: the service provider

(say, hosting.example.net) wishes to limit its liability and therefore does not wish to hold the certificate and private key for the tenant (say, example.com) and the tenant wishes to improve the security of the service and therefore does not wish to share its certificate and private key with the service provider. As a result, server-to-server communications to example.com go unencrypted or the communications are TLS-encrypted but the certificates are not checked (which is functionally equivalent to a connection using an anonymous key exchange). This is also true of client-to-server communications, forcing end users to override certificate warnings or configure their clients to accept or "pin" certificates for hosting.example.net instead of example.com. The fundamental problem here is that if DNSSEC is not used then the act of delegation via DNS SRV records is inherently insecure.

The specification for use of SRV records with DANE [I-D.ietf-dane-srv] explains how to use DNSSEC for secure delegation with the DANE prooftype, and the POSH specification [I-D.ietf-xmpp-posh] explains how to use HTTPS redirects for secure delegation with the POSH prooftype.

7. Prooftype Model

In general, a domain name association (DNA) prooftype conforms to the following definition:

prooftype: A mechanism for proving an association between a domain name and an XML stream, where the mechanism defines (1) the nature of the server's proof, (2) the matching rules for comparing the client's verification material against the server's proof, (3) how the client obtains its verification material, and (4) whether the mechanism depends on secure DNS.

The PKIX, DANE, and POSH prooftypes adhere to this model. (Some prooftypes depend on, or are enhanced by, secure DNS and thus also need to describe how they ensure secure delegation.)

Other prooftypes are possible; examples might include TLS with PGP keys [RFC6091], a token mechanism such as Kerberos [RFC4120] or OAuth [RFC6749], and Server Dialback keys [XEP-0220].

Although the PKIX prooftype reuses the syntax of the XMPP Server Dialback protocol [XEP-0220] for signalling between servers, this framework document does not define how the generation and validation of Server Dialback keys (also specified in [XEP-0220]) is a DNA prooftype. However, nothing in this document prevents the continued use of Server Dialback for signaling, and a future specification (or

an updated version of [XEP-0220]) might define a DNA prooftype for Server Dialback keys in a way that is consistent with this framework.

8. IANA Considerations

The POSH specification [I-D.ietf-xmpp-posh] provides guidelines for registering the well-known URIs [RFC5785] of protocols that make use of POSH. This specification registers two such URIs, for which the completed registration templates follow.

8.1. Well-Known URI for xmpp-client Service

This specification registers "posh._xmpp-client._tcp.json" in the Well-Known URI Registry as defined by [RFC5785].

URI suffix: posh._xmpp-client._tcp.json

Change controller: IETF

Specification document(s): [[this document]]

8.2. Well-Known URI for xmpp-server Service

This specification registers "posh._xmpp-server._tcp.json" in the Well-Known URI Registry as defined by [RFC5785].

URI suffix: posh._xmpp-server._tcp.json

Change controller: IETF

Specification document(s): [[this document]]

9. Security Considerations

With regard to the PKIX prooftype, this document supplements but does not supersede the security considerations of [RFC6120] and [RFC6125].

With regard to the DANE and PKIX prooftypes, the reader is referred to [I-D.ietf-dane-srv] and [I-D.ietf-xmpp-posh], respectively.

Any future prooftypes need to thoroughly describe how they conform to the prooftype model specified in Section 7 of this document.

10. References

10.1. Normative References

[I-D.ietf-dane-srv]
Finch, T., Miller, M., and P. Saint-Andre, "Using DNS-Based Authentication of Named Entities (DANE) TLSA Records with SRV Records", draft-ietf-dane-srv-06 (work in progress), June 2014.

[I-D.ietf-xmpp-posh]
Miller, M. and P. Saint-Andre, "PKIX over Secure HTTP (POSH)", draft-ietf-xmpp-posh-03 (work in progress), January 2015.

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, November 1987.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

[RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, February 2000.

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, March 2005.

[RFC4949] Shirey, R., "Internet Security Glossary, Version 2", RFC 4949, August 2007.

[RFC5280] Cooper, D., Santesson, S., Farrell, S., Boeyen, S., Housley, R., and W. Polk, "Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile", RFC 5280, May 2008.

[RFC5785] Nottingham, M. and E. Hammer-Lahav, "Defining Well-Known Uniform Resource Identifiers (URIs)", RFC 5785, April 2010.

[RFC6120] Saint-Andre, P., "Extensible Messaging and Presence Protocol (XMPP): Core", RFC 6120, March 2011.

[RFC6125] Saint-Andre, P. and J. Hodges, "Representation and Verification of Domain-Based Application Service Identity within Internet Public Key Infrastructure Using X.509 (PKIX) Certificates in the Context of Transport Layer Security (TLS)", RFC 6125, March 2011.

[RFC6698] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, August 2012.

[RFC7218] Gudmundsson, O., "Adding Acronyms to Simplify Conversations about DNS-Based Authentication of Named Entities (DANE)", RFC 7218, April 2014.

[XEP-0220] Miller, J., Saint-Andre, P., and P. Hancke, "Server Dialback", XSF XEP 0220, August 2014.

10.2. Informative References

[I-D.ietf-uta-xmpp] Saint-Andre, P. and a. alkemade, "Use of Transport Layer Security (TLS) in the Extensible Messaging and Presence Protocol (XMPP)", draft-ietf-uta-xmpp-05 (work in progress), January 2015.

[RFC2142] Crocker, D., "MAILBOX NAMES FOR COMMON SERVICES, ROLES AND FUNCTIONS", RFC 2142, May 1997.

[RFC3920] Saint-Andre, P., Ed., "Extensible Messaging and Presence Protocol (XMPP): Core", RFC 3920, October 2004.

[RFC4120] Neuman, C., Yu, T., Hartman, S., and K. Raeburn, "The Kerberos Network Authentication Service (V5)", RFC 4120, July 2005.

[RFC6066] Eastlake, D., "Transport Layer Security (TLS) Extensions: Extension Definitions", RFC 6066, January 2011.

[RFC6091] Mavrogiannopoulos, N. and D. Gillmor, "Using OpenPGP Keys for Transport Layer Security (TLS) Authentication", RFC 6091, February 2011.

[RFC6749] Hardt, D., "The OAuth 2.0 Authorization Framework", RFC 6749, October 2012.

[XEP-0045] Saint-Andre, P., "Multi-User Chat", XSF XEP 0045, February 2012.

[XEP-0288] Hancke, P. and D. Cridland, "Bidirectional Server-to-Server Connections", XSF XEP 0288, September 2013.

Appendix A. Acknowledgements

Thanks to Richard Barnes, Stephen Farrell, and Jonas Lindberg for contributing to earlier versions of this document.

Authors' Addresses

Peter Saint-Andre
&yet

Email: peter@andyet.com
URI: https://andyet.com/

Matthew Miller
Cisco Systems, Inc.
1899 Wynkoop Street, Suite 600
Denver, CO 80202
USA

Email: mamille2@cisco.com

Philipp Hancke
&yet

Email: fippo@andyet.com
URI: https://andyet.com/