

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 26, 2013

M. Miller
P. Saint-Andre
Cisco Systems, Inc.
February 22, 2013

Using DNS Security Extensions (DNSSEC) and DNS-based Authentication of
Named Entities (DANE) as a Proofotype for XMPP Domain Name Associations
draft-miller-xmpp-dnssec-proofotype-04

Abstract

This document defines a proofotype that uses DNS-based Authentication of Named Entities (DANE) for associating a domain name with an XML stream in the Extensible Messaging and Presence Protocol (XMPP). It also defines a method that uses DNS Security (DNSSEC) for securely delegating a source domain to a derived domain in XMPP.

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 26, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terminology	2
3. Requirements	3
4. Secure Delegation using DNS SRV	3
5. DANE Proofotype	4
5.1. No Service Records	4
5.2. Insecure Delegation	4
5.3. Secure Delegation	4
6. Order of Operations	5
7. Internationalization Considerations	5
8. Security Considerations	5
9. IANA Considerations	6
10. References	6
10.1. Normative References	6
10.2. Informative References	7
Authors' Addresses	7

1. Introduction

The [XMPP-DNA] specification defines a framework for secure delegation and strong domain name associations (DNA) in the Extensible Messaging and Presence Protocol (XMPP). This document defines a secure delegation method that uses DNS Security (DNSSEC) [RFC4033] in conjunction with the standard DNS SRV records [RFC2782] employed in domain name resolution in XMPP, with the result that a client or peer server that initiates an XMPP stream can legitimately treat a derived domain as a reference identifier during stream negotiation. This document also defines a DNA proofotype that uses DNS-based Authentication of Named Entities [RFC6698] (DANE) to verify TLS certificates containing source domains or derived domains during stream negotiation.

2. Terminology

This document inherits XMPP terminology from [RFC6120], DNS terminology from [RFC1034], [RFC1035], [RFC2782] and [RFC4033], and

security terminology from [RFC4949] and [RFC5280]. The terms "source domain", "derived domain", "reference identifier", and "presented identifier" are used as defined in the "CertID" specification [RFC6125].

This document is applicable to connections made from an XMPP client to an XMPP server ("xmpp-client.tcp") or between XMPP servers ("xmpp-server.tcp"). In both cases, the XMPP initiating entity acts as a TLS client and the XMPP receiving entity acts as a TLS server. Therefore, to simplify discussion this document uses "xmpp-client.tcp" to describe to both cases, unless otherwise indicated.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Requirements

An XMPP initiating entity (TLS client) that wishes to use the DNSSEC proofotype MUST do so before exchanging stanzas addressed to the source domain. In general, this means that the proof MUST be completed before the XMPP stream is restarted following STARTTLS negotiation (as specified in [RFC6120]). However, connections between XMPP servers MAY also use this proofotype to verify the addition of new source domains onto an existing connection, such as multiplexing or "piggybacking" via [XEP-0220].

4. Secure Delegation using DNS SRV

In order to determine if delegation using DNS SRV records is secure, an XMPP initiating entity (TLS client) performs the following actions:

1. Query for the appropriate SRV resource record for the source domain (e.g., "xmpp-client.tcp.im.example.com").
2. If there is no SRV resource record, pursue the fallback methods described in [RFC6120].
3. If there is an SRV resource record, validate that the SRV record answer is secure according to [RFC4033]. If the answer is insecure, then delegation to the derived domain(s), as indicated by the "target host" field, is insecure and the TLS client MUST treat only the source domain as a reference identifier during certificate verification, as described in [RFC6120]; if the answer is bogus, the TLS client MUST abort.

4. If the answer is secure, the TLS client SHOULD consider any derived domain(s) in the answer as securely delegated; during certificate verification, the TLS client MUST treat both the source domain and the derived domain to which it has connected as reference identifiers.

The foregoing secure delegation method can be used with the DANE proofotype defined below, or with the PKIX proofotype specified in [RFC6120].

5. DANE Proofotype

DANE provides additional tools to verify the keys used in TLS connections. A TLS client MAY use DANE for TLS certificate verification; its use depends on the delegation status of the source domain, as described in the following sections.

5.1. No Service Records

If no SRV records are found for the source domain, then the TLS client MUST query for a TLSA resource record as described in [RFC6698], where the prepared domain name MUST contain the source domain and the IANA-registered port 5222 for client-to-server streams (e.g., "_5222._tcp.im.example.com") or the IANA-registered port 5269 for server-to-server streams (e.g., "_5269._tcp.im.example.com").

In this case, the TLS client MUST treat only the source domain as its reference identifier during certificate verification, as described in [RFC6120].

5.2. Insecure Delegation

If the delegation of a source domain to a derived domain is not secure, then the TLS client MUST NOT make a TLSA record query to the derived domain as described in [RFC6698]. Instead, the TLS client MUST treat only the source domain as its reference identifier during certificate verification, as described in [RFC6120], and MUST NOT use DANE.

5.3. Secure Delegation

If the source domain has been delegated to a derived domain in a secure manner as described under Section 4, then the TLS client MUST query for a TLSA resource record as described in [RFC6698], where the prepared domain name MUST contain the derived domain and a port obtained from the SRV answer (e.g., "_5555._tcp/hosting.example.net" for an SRV record such as "_xmpp-client._tcp.im.example.com IN TLSA 1 1 5555 hosting.example.net").

If no TLSA resource records exist for the specified service, then the TLS client MUST perform certificate verification as described under Section 4.

If TLSA resource records exist for the specified service, then the TLS client MUST treat the derived domain(s) as its reference identifier during certificate verification, using the information from the TLSA answer as the basis for verification as described in [RFC6698].

6. Order of Operations

The processes for the DANE proofotype MUST be complete before the TLS handshake over the XMPP connection finishes, so that the client can perform verification of reference identities. To that end, a TLS client SHOULD perform the processes for this proofotype as part of its normal DNS resolution of the source domain into a socket address. Validating secure delegation ought to be done immediately upon receiving the answers to the SRV and follow-up A/AAAA queries; queries for TLSA records ought to be done once the target service is determined (whether the source domain and IANA-registered port, or delegated domain and port).

Ideally a TLS client will perform the DNSSEC and DANE processes in parallel with other XMPP session establishment processes where possible (e.g., perform the TLSA resource queries as the socket connection is made to the server); this is sometimes called the "happy eyeballs" approach, similar to [RFC6555] for IPv4 and IPv6. However, a TLS client might delay as much of the XMPP session establishment as it needs to in order to gather all of the DNSSEC- and DANE-based verification material. For instance, a TLS client might not open the socket connection until it has validated the secure delegation, or it might delay beginning the TLS handshake until it has obtained the TLSA certificate verification material.

7. Internationalization Considerations

If the SRV, A/AAAA, and TLSA record queries are for an internationalized domain name, then they need to use the A-label form as defined in [RFC5890].

8. Security Considerations

This document supplements but does not supersede the security considerations provided in [RFC4033], [RFC6120], [RFC6125], and [RFC6698].

9. IANA Considerations

This document has no actions for the IANA.

10. References

10.1. Normative References

- [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.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [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, May 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.
- [RFC5890] Klensin, J., "Internationalized Domain Names for Applications (IDNA): Definitions and Document Framework", RFC 5890, August 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.
- [XEP-0220] Miller, J., Saint-Andre, P., and P. Hancke, "Server Dialback", XSF XEP 0220, August 2011.
- [XMPP-DNA] Saint-Andre, P. and M. Miller, "Domain Name Associations (DNA) in the Extensible Messaging and Presence Protocol (XMPP)", draft-saintandre-xmpp-dna-01 (work in progress), February 2013.

10.2. Informative References

- [RFC6555] Wing, D. and A. Yourtchenko, "Happy Eyeballs: Success with Dual-Stack Hosts", RFC 6555, April 2012.

Authors' Addresses

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

Email: mamille2@cisco.com

Peter Saint-Andre
Cisco Systems, Inc.
1899 Wynkoop Street, Suite 600
Denver, CO 80202
USA

Email: psaintan@cisco.com

XMPP
Internet-Draft
Intended status: Standards Track
Expires: August 26, 2013

M. Miller
P. Saint-Andre
Cisco Systems, Inc.
February 22, 2013

Using PKIX over Secure HTTP (POSH) as a Proofotype for XMPP Domain Name
Associations
draft-miller-xmpp-posh-proofotype-03

Abstract

This document defines a proofotype involving PKIX over Secure HTTP (POSH) for associating a domain name with an XML stream in the Extensible Messaging and Presence Protocol (XMPP). It also defines a method involving HTTPS redirects (appropriate for use with the POSH proofotype) for securely delegating a source domain to a derived domain in XMPP.

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 26, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terminology	3
3. Proofotype	3
4. Secure Delegation	4
4.1. Permanent versus Temporary Redirects	6
5. Order of Operations	6
6. Caching Results	6
7. Alternates and Roll-over	7
8. Security Considerations	8
9. IANA Considerations	9
9.1. The "posh._xmpp-client._tcp.json" Well-Known URI	9
9.2. The "posh._xmpp-server._tcp.json" Well-Known URI	9
10. References	9
10.1. Normative References	9
10.2. Informative References	10
10.3. Informative References	10
Authors' Addresses	10

1. Introduction

The [XMPP-DNA] specification defines a framework for secure delegation and strong domain name associations (DNA) in the Extensible Messaging and Presence Protocol (XMPP). This document defines a DNA proofotype using PKIX certificates obtained over secure HTTP ("POSH"), as well as a secure delegation method, based on HTTPS redirects, that is appropriate for use with the POSH proofotype.

The rationale for POSH is driven by current operational realities. It is effectively impossible for a hosting service to provide and maintain PKIX certificates [RFC5280] that include the appropriate identifiers [RFC6125] for each hosted domain. It is true that DNS-based technologies are emerging for secure delegation, in the form of DNS Security ([RFC4033] and [RFC6698]); however, these technologies are not yet widely deployed and might not be deployed in the near future for domains outside the most common top-level domains (e.g., ".COM", ".NET", ".EDU"). Because the XMPP community wishes to deploy secure delegation and strong domain name associations as widely and as quickly as possible, this document specifies how to use secure HTTP ([RFC2616] and [RFC2818]) and PKIX certificates [RFC5280] to verify that a domain is delegated to a hosting provider and also establish a strong association between a domain name and an XML stream.

2. Terminology

This document inherits XMPP terminology from [RFC6120] and security terminology from [RFC5280]. The terms "source domain", "derived domain", "reference identifier", and "presented identifier" are used as defined in the "CertID" specification [RFC6125].

This document is applicable to connections made from an XMPP client to an XMPP server ("`_xmpp-client._tcp`") or between XMPP servers ("`_xmpp-server._tcp`"). In both cases, the XMPP initiating entity acts as a TLS client and the XMPP receiving entity acts as a TLS server. Therefore, to simplify discussion this document uses "`_xmpp-client._tcp`" to describe both cases, unless otherwise indicated.

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

3. Proofotype

POSH stands for PKIX Over Secure HTTP: the server's proof consist of a PKIX certificate [RFC5280], the certificate is checked according to the rules from [RFC6120] and [RFC6125], the client obtains its verification material by retrieving the certificate over HTTPS ([RFC2616] and [RFC2818]) from a well-known URI [RFC5785], and secure DNS is not necessary since the HTTPS retrieval mechanism relies on the chain of trust based on the public key infrastructure.

The process for retrieving a PKIX certificate over secure HTTP is as follows.

1. The initiating entity performs an HTTPS GET at the source domain to the path "`/.well-known/posh._<service>._tcp.json`"; where "`_<service>`" MUST be either "`_xmpp-client`" for XMPP client-to-server connections or "`_xmpp-server`" for XMPP server-to-server connections. Here is an example:

```
HTTP GET /.well-known/posh._xmpp-server._tcp.json HTTP/1.1
Host: im.example.com
```

2. If the source domain HTTPS server has a certificate for the requested path, it MUST respond with a success status code, with the message body as a JSON Web Key Set (JWK Set) [JOSE-JWK], which itself contains at least one JWK of type "PKIX" [JOSE-PKIX-KEY] that the XMPP server at the source domain will

present during the TLS negotiation phase of XMPP stream setup (linebreaks and whitespace added for readability). Here is an example:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 806

{
  "keys": [
    {
      "kty": "PKIX",
      "x5c": [
        "MIICPTCCAaYCCQDDVeBaBmWC_jANBgkqhkiG9w0BAQUFADBjMQswCQYD
        VQQGEwJVUzERMA8GA1UECBMIQ29sb3JhZG8xDzANBgNVBACTBkRlbn
        ZlcjEXMBUGA1UEChMOaW0uZXBhbXBsZS5jb20xZzAVBgNVBAMTDm1tL
        mV4YW1wbGUuY29tMB4XDTEyMDYxMTI0NTQ0NFoXDTIyMDYwOTI0NTQ0
        NFowYzELMAKGA1UEBhMCVVMxETAPBgNVBAGTCENvbG9yYWRvMQ8wDQYD
        VQQHEwZlZW52ZXIxFzAVBgNVBAoTDm1tLmV4YW1wbGUuY29tMRcwFQ
        YDVQQDEw5pbS5leGFtcGxlLmNvbTCBnzANBgkqhkiG9w0BAQEFAAOBj
        QAwwYkCgYEA4hoKHi_B07eQH-1NB9gWiNFDT__AbTHQOEC0AOr4Gh_o
        9PU7kD0gklU4uv7rSAhAyCe4WaoiQ_HShzEryGfHiZmWht0BaYmj19
        iuPWRecZOXWqKZji9NtAxn9l3kdon_YLJcrPGyNTGK66-ggNaqy8LkQ
        QpI4rff60yHHZ_0XkCAwEAATANBgkqhkiG9w0BAQUFAAOBgQDcw3u30
        bSMlykWyZ-tTDSlQ3wLSVB9RsR8jXmJvMo7y7icXwg54a9M3xipjZtr
        fAhYM5I5iqUTQPkis26n9SQpRm5bonEFDA3WGwrwma35biP9-NSBWz
        SaDF8AztwFNKXXl6_U6hWwG05G_NdeS1lgpww9NUDraJgVoDpRK04tg"
      ]
    }
  ]
}
```

4. Secure Delegation

When PKIX Over Secure HTTP (POSH) is the DNA proofotype, it is possible to use HTTPS redirects in determining if a domain is securely delegated, as follows:

1. The initiating entity performs an HTTPS GET at the source domain to the path `"/.well-known/posh._<service>._tcp.json"`; where `"_<service>"` MUST be either `"_xmpp-client"` for XMPP client-to-server connections or `"_xmpp-server"` for XMPP server-to-server connections. Here is an example:

```
GET /.well-known/posh._xmpp-server._tcp.json HTTP/1.1
Host: im.example.com
```

2. If the source domain HTTPS server has delegated to a derived domain, it MUST respond with one of the redirect mechanisms provided by HTTP (e.g., using the 302, 303, 307, or 308 response). The 'Location' header MUST specify an HTTPS URL, where the hostname and port is the derived domain HTTPS server, and the path MUST match the pattern "_<service>._tcp.json"; where "_<service>" MUST be identical to the "_<service>" portion of the original request (line breaks added for readability). Here is an example:

```
HTTP/1.1 302 Found
Location: https://hosting.example.net/.well-known
        /posh._xmpp-server._tcp.json
```

3. The initiating entity performs an HTTPS GET to the URL specified in the 'Location' header. Here is an example:

```
GET /.well-known/posh._xmpp-server._tcp.json HTTP/1.1
Host: hosting.example.net
```

4. If the derived domain HTTPS server has a certificate, it MUST respond with a success status code, with the message body as a JSON Web Key Set (JWK Set) [JOSE-JWK], which itself contains at least one JWK of type "PKIX" [JOSE-PKIX-KEY] that the XMPP server at the derived domain will present during the TLS negotiation phase of XMPP stream setup. Here is an example:

```
HTTP/1.1 200 OK
Content-Type: application/json
Content-Length: 806
```

```
{
  "keys":[
    {
      "kty":"PKIX",
      "x5c":[
        "MIICPTCCAaYCCQDDVeBaBmWC_jANBgkqhkiG9w0BAQUFADBjMQswCQYD
        VQQGEwJVUzERMA8GA1UECBMIQ29sb3JhZG8xDzANBgNVBACTBkRlbn
        ZlcjEXMBUGA1UEChMOaW0uZXhhbXBsZS5jb20xFzAVBgNVBAMTDm1tL
```

```

mV4YW1wbGUuY29tMB4XDTEyMDYxMTIxNTQ0NFoXDTIyMDYwOTIxNTQ0
NFowYzELMAkGA1UEBhMCVVMxETAPBgNVBAGTCENvbG9yYWRvMQ8wDQY
DVQQHEwZEZw52ZXIxFzAVBgNVBAoTDm1tLmV4YW1wbGUuY29tMRcwFQ
YDVQQDEw5pbS5leGFtcGxlLmNvbTCBnzANBgkqhkiG9w0BAQEFAAOBj
QAwwYkCgYEA4hoKhi_B07eQH-1NB9gWiNFDT__AbTHQOEC0AOr4Gh_o
9PU7kD0gklU4uv7rSAhAyCe4WaOiQ_HShzEryGfHiZmWht0BaYmj19
iuPWRecZOXWqKZji9NtAxn9l3kdon_YLJcrPGyNTGK66-ggNaqy8LkQ
QpI4rff60yHHZ_0XkCAwEAATANBgkqhkiG9w0BAQUFAAOBgQDcw30
bSMlykWyZ-tTDSlQ3wLSVB9RsR8jXmJvMo7y7icXwg54a9M3xipjZtr
fAhYM5I5iqUTQPki6s26n9SQpRm5bonEFDA3WGwrwma35biP9-NSBWz
SaDF8AztwFNKXXl6_U6hWwG05G_NdeS1lgpww9NUDraJgVoDpRK04tg"

```

```

    ]
  }
]
}

```

4.1. Permanent versus Temporary Redirects

Care needs to be taken with which redirect mechanism is used for delegation. Clients might remember the redirected location in place of the original, which can lead to verification mismatches when a source domain is migrated to a different delegated domain.

To mitigate this concern, source domains SHOULD use only temporary redirect mechanisms, such as HTTP status codes 302 (Found) and 307 (Temporary Redirect). Clients MAY treat any redirect as temporary, ignoring the specific semantics for 301 (Moved Permanently) or 308 (Permanent Redirect) [HTTP-STATUS-308].

5. Order of Operations

The processes for the POSH proofotype MUST be complete before the TLS handshake over the XMPP connection finishes, so that the client can perform verification of reference identities. Ideally a TLS client ought to perform the POSH processes in parallel with other XMPP session establishment processes; this is sometimes called the "happy eyeballs" approach, similar to [RFC6555] for IPv4 and IPv6. However, a TLS client might delay as much of the XMPP session establishment as it needs to in order to gather all of the POSH-based verification material. For instance, a TLS client might not open the socket connection until it retrieves the PKIX certificates.

6. Caching Results

Ideally, the initiating entity relies on the expiration time of the certificate obtained via POSH, and not on HTTP caching mechanisms.

To that end, the HTTPS servers for source and derived domains SHOULD specify a 'Cache-Control' header indicating a short duration (e.g., max-age=60) or "no-cache" to indicate the response (redirect or content) is not appropriate to cache at the HTTP level.

7. Alternates and Roll-over

To indicate alternate PKIX certificates, such as when an existing certificate will soon expire, the returned JWK Set can contain multiple "PKIX" JWK objects. The JWK Set SHOULD be ordered with the most relevant certificate first as determined by the XMPP server operator (e.g., the certificate soonest to expire), followed by the next most relevant certificate (e.g., the renewed certificate). Here is an example:

```
{
  "keys":[
    {
      "kty":"PKIX",
      "x5c":[
        "MIICYTCCAcqgAwIBAgIJAK_Lh7cXMZvdMA0GCSqGSIb3DQEBBQUAME
        8xCzAJBgNVBAYTA1VTMREwDwYDVQQIEWhDb2xvcmFkbzEPMA0GA1UEB
        xMGRGVudmVyMRwwGgYDVQQDExNob3N0aW5nLmV4YW1wbGUubmV0MB4X
        DTEzMDIwNzE4MjY0MFoXDTEzMDIwNTE4MjY0MFowTzELMAKGA1UEBhM
        CVVMxETAPBgNVBAGTCENvbG9yYWRvMQ8wDQYDVQQHEwZEZW52ZXIxHD
        AaBgNVBAMTE2hvc3RpbmcuZXhhbXBsZS5uZXQwgZ8wDQYJKoZIhvcNA
        QEBBQADgY0AMIGJAoGBAOLjqQxacJ-DQNOuVxNzoBBRyLku7V_ZEpFY
        8SHPyrK38I7Q3lWnEpAyUanpMClDMV0B_EJQDeueJgWkyrgd6bDZLvi
        _UtGha9E4q-IpHO6cM_cSE9d_oZuCcdGV8HHjK9mlxHUEyeTGAm1tMA
        m7j_BNfdhETkUqTfFPggFdmhAXAgMBAAGjRTBDMEEGA1UdEQQ6MDigI
        QYIKwYBBQUHCAWgFQwTaG9zdGluZy5leGFtcGx1Lm5ldIITaG9zdGlu
        Zy5leGFtcGx1Lm5ldDANBgkqhkiG9w0BAQUFAAOBgQAaz8lgC5KqFQo
        WGF8mJz_mYx2pW6i-QeYw-BqpdAgdkrRvOHlJ4pYRhkaJkfdiauvHcM
        ZDPWuuSm7jzIEOPqZdzYXkffgfr4br5UOAmYqpiKpjlSsTLd5h_38p-
        3lz-1502wcs1xveBTYtIT13MAI844IBCZF-xDl-wpJG3kktTA"
      ]
    }
  ]
}
```

```
{
  "kty":"PKIX",
  "x5c":[
    "MIIC-zCCAeOgAwIBAgIBAJANBgkqhkiG9w0BAQUFAADBGMQswCQYDVQ
    QGEwJVUzERMA8GA1UECBMIQ29sb3JhZG8xZDZANBgNVBACTBkRlbnZlc
    jETMBEGA1UEAxMKRXhhbXBsZSBDQTAeFw0xMzAyMTIyMTI5MDBaFw0x
    NDAyMTIyMTI5MDBaME8xCzAJBgNVBAYTA1VTMREwDwYDVQQIEWhDb2x
    vcmFkbzEPMA0GA1UEBxMGRGVudmVyMRwwGgYDVQQDExNob3N0aW5nLm
    V4YW1wbGUubmV0MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDi4
    6kMWnCfg0DTrlcTc6AQUCi5Lulf2RKRWPEhz8qyt_CO0N5VpxKQm1Gp
    6TApQzFdAfxCUA3rniYFpMq4Hemw2S74v1LRowvR0KviKRzunDP3EhP
```

```

Xf6GbgmHRlfBx4yvZtcRlBMnkxgJtbTAJu4_wTRXYRE5FKk3xT4IBXT
IQFwIDAQABo28wbTAMBgNVHRMBAf8EAjAAMB0GA1UdDgQWBBrGaaG6v
5py2KwjT-X-ToLKTEIqeVTALBgNVHQ8EBAMCBeAwEQYJYIZIAyb4QgEB
BAQDAGZAMB4GCWCGSAGG-EIBDQQRFG94Y2EgY2VydGlmawNhdGUwDQY
JKoZIhvcNAQEFBQADggEBAE6Vhvd0OuMHJjyi8F8NoFSCRYOJXOry5B
lmU6eVwEcUQSakHaC4Q2isWCIES58Wm5P2VVQTYBUn58H7ZR9-7looj
YVykweIQmE_aaVsMM-8AwTMJ7qj7aGhXFlKT2xwiPMVq9JF_Gv43qSy
V9GJ3Uw5Jz6AN4WawXm1IVD0eKhPoHSD00wfnFc8KM8mHPu7JXqIriX
18w4jffj3ySuHIkXeOjdbDWqZWJ7akBVf8McbB05tXP5T7sDTV-t8qh5
6fdnSQC-qO-sQgmWlKLfTKybT6Fa6J7ChEd_sOJNqB9SoMar5sRYyfS
foV0D7m_IFlMI6X95rLlYnKIGxDYWBq4ck",
"MIIDEtCCAmGgAwIBAgIBATANBgkqhkiG9w0BAQUFADBGMQswCQYDVQ
QGEwJUVuZERMA8GA1UECBMIQ29sb3JhZG8xZDZANBgNVBAcTBkRlbnZlc
jETMBEAGALUEAxMKRXhhbXBsZSBBDQTAeFw0xMzAyMTIyMTI4MDBaFw0y
MzAyMTIyMTI4MDBaMEYxChAJBgNVBAYTAlVTMREwDwYDVQQIEWhDb2x
vcnFkbzEPMA0GA1UEBxMGRGRGVudmVyMRMwEQYDVQQDEwpFeGFtcGxlIE
NBMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEazNQ30X7uX
Tg-4jKadtRO5uQEMRMnkZvDnptbWAtx0dlPsufQ2kfvog0gDhigjPEZ
DV9S-zm63Ia-eqJ3ROT9jDXjtF6s_IawITf5cPSNxn8qP8w-vbiy0rB
4W4Nk1Dwji7KJ_wKNo0mwOx_qWNjSk3yoaU4sUEuIypizgLxKAr25vV
vAJAXf6HafdQoVAIdCZ_7qbBPI7aurdU_NdmabbKBK0lp8aV1MYLzz8D
I0hWcBQa2-gOSUcd_yTlaz7UpMjGllbnVlUDxyJeCzbBaHny5NlWWHs
GnsbucbM-9yeAMbRes_z0KeHxcRtomd8bh7As12RIXKrk5GRoNVKAoi
wLQIDAQABo3IwcDAPBgNVHRMBAf8EBTADAQH_MB0GA1UdDgQWBBSyie
t77RfWpH3X8NMwGFVu2ldJPTALBgNVHQ8EBAMCAQYwEQYJYIZIAyb4Q
gEBBAQDAGAHMB4GCWCGSAGG-EIBDQQRFG94Y2EgY2VydGlmawNhdGUw
DQYJKoZIhvcNAQEFBQADggEBAIE-gvYX-2MOAmL3qOraIYUbleDeUyC
rxroqrI1xX3jDapMPltCxuZr8VklLjHaNpe7sLJlFWSaQHkZe4snxWL
SdINLrgFhxskclAlSLutPVTA4xPwo60t0hBJE0NJ8kC8gVvvlWXWaiI
IVszG3vLBcfxZeuOS4JsVwGbTt5uKsVIJ2VkrIBG4ey5lsS508u0vRf
ei7HFr1NzZ8y5BHoix9VLN2--n1lSNicwDOo2V618B8GQnPqM2dsaDa
AlwIrMZeEyoRtIN25jcW-as4sS9dPJlueNIzrSuzlXtKYGjflaTcEfD
-_kImTw9tHzS57iBXHqgQTQo61pYzAZMlk9wA"
]
}
]
}

```

8. Security Considerations

This document supplements but does not supersede the security considerations provided in [RFC2616], [RFC2818], [RFC6120], and [RFC6125].

Specifically, communication via HTTPS depends on checking the identity of the HTTP server in accordance with [RFC2818].

9. IANA Considerations

9.1. The "posh._xmpp-client._tcp.json" Well-Known URI

This specification registers the "posh._xmpp-client._tcp.json" well-known URI 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]]

9.2. The "posh._xmpp-server._tcp.json" Well-Known URI

This specification registers the "posh._xmpp-server._tcp.json" well-known URI 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]]

10. References

10.1. Normative References

[JOSE-JWK]

Jones, M., "JSON Web Key (JWK)", draft-ietf-jose-json-web-key-08 (work in progress), December 2012.

[JOSE-PKIX-KEY]

Miller, M., "JSON Web Key (JWK) for PKIX Certificates", draft-miller-jose-pkix-key-01 (work in progress), February 2013.

[XMPP-DNA]

Saint-Andre, P. and M. Miller, "Domain Name Associations (DNA) in the Extensible Messaging and Presence Protocol (XMPP)", draft-saintandre-xmpp-dna-01 (work in progress), February 2013.

[RFC2119]

Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

- [RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.
- [RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000.
- [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.

10.2. Informative References

- [HTTP-STATUS-308] Reschke, J., "The Hypertext Transfer Protocol (HTTP) Status Code 308 (Permanent Redirect)", draft-reschke-http-status-308-07 (work in progress), March 2012.
- [RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, May 2005.
- [RFC6698] Hoffman, P. and J. Schlyter, "The DNS-Based Authentication of Named Entities (DANE) Transport Layer Security (TLS) Protocol: TLSA", RFC 6698, August 2012.

10.3. Informative References

- [RFC6555] Wing, D. and A. Yourtchenko, "Happy Eyeballs: Success with Dual-Stack Hosts", RFC 6555, April 2012.

Authors' Addresses

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

Email: mamille2@cisco.com

Peter Saint-Andre
Cisco Systems, Inc.
1899 Wynkoop Street, Suite 600
Denver, CO 80202
USA

Email: psaintan@cisco.com

HyBi Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 29, 2013

L. Stout, Ed.
&yet
J. Moffitt
E. Cestari
ProcessOne
February 25, 2013

An XMPP Sub-protocol for WebSocket
draft-moffitt-xmpp-over-websocket-02

Abstract

This document defines a binding for the XMPP protocol over a WebSocket transport layer. A WebSocket binding for XMPP provides higher performance than the current HTTP binding for XMPP.

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 29, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terminology	3
3. XMPP Sub-Protocol	3
3.1. Handshake	3
3.2. Messages	3
3.3. XMPP Stream Setup	3
3.4. Stream Errors	4
3.5. Closing the Connection	4
3.6. Stanzas	4
3.7. Stream Restarts	4
3.8. Pings and Keepalives	5
3.9. Use of TLS	5
3.10. Stream Management	5
4. Examples	5
5. Security Considerations	5
6. IANA Considerations	6
7. Informative References	6
Authors' Addresses	7

1. Introduction

Applications using XMPP (see [RFC6120] and [RFC6121]) on the Web currently make use of BOSH (see [XEP-0124] and [XEP-0206]), an XMPP binding to HTTP. BOSH is based on the HTTP long polling technique, and it suffers from high transport overhead compared to XMPP's native binding to TCP. In addition, there are a number of other known issues with long polling [RFC6202], which have an impact on BOSH-based systems.

It would be much better in most circumstances to avoid tunneling XMPP over HTTP long polled connections and instead use the XMPP protocol directly. However, the APIs and sandbox that browsers have provided do not allow this. The WebSocket protocol [RFC6455] now exists to solve these kinds of problems. The WebSocket protocol is a bi-directional protocol that provides a simple message-based framing layer over raw sockets and allows for more robust and efficient communication in web applications.

The WebSocket protocol enables two-way communication between a client and a server, effectively emulating TCP at the application layer and therefore overcoming many of the problems with existing long-polling techniques for bidirectional HTTP. This document defines a WebSocket sub-protocol for the Extensible Messaging and Presence Protocol (XMPP).

2. Terminology

The basic unit of framing in the WebSocket protocol is called a message. In XMPP, the basic unit is the stanza, which is a subset of the first-level children of each document in an XMPP stream (see Section 9 of [RFC6120]). XMPP also has a concept of messages, which are stanzas whose top-level element name is message. In this document, the word "message" will mean a WebSocket message, not an XMPP message stanza (see Section 3.2).

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].

3. XMPP Sub-Protocol

3.1. Handshake

The XMPP sub-protocol is used to transport XMPP over a WebSocket connection. The client and server agree to this protocol during the WebSocket handshake (see Section 1.3 of [RFC6455]).

During the WebSocket handshake, the client MUST include the |Sec-WebSocket-Protocol| header in its handshake, and the value |xmpp| MUST be included in the list of protocols. The reply from the server MUST also contain |xmpp| in its own |Sec-WebSocket-Protocol| header in order for an XMPP sub-protocol connection to be established.

Once the handshake is complete, WebSocket messages sent or received will conform to the protocol defined in the rest of this document.

3.2. Messages

Data frame messages in the XMPP sub-protocol MUST be of the text type and contain UTF-8 encoded data. The close control frame's contents are specified in Section 3.5. Control frames other than close are not restricted.

Unless noted in text, the word "message" will mean a WebSocket message containing a text data frame.

3.3. XMPP Stream Setup

The first message sent after the handshake is complete MUST be an XMPP opening stream tag as defined in XMPP [RFC6120] or an XML text declaration (see Section 4.3.1 of [W3C.REC-xml-20081126]) followed by an XMPP opening stream tag. The stream tag MUST NOT be closed (i.e. the closing </stream:stream> tag should not appear in the message) as

it is the start of the client's outgoing XML. The '<' character of the tag or text declaration MUST be the first character of the text payload.

The server MUST respond with a message containing an error (see Section 3.4), its own opening stream tag, or an XML text declaration followed by an opening stream tag.

Except in the case of certain stream errors (see Section 3.4), the opening stream tag, <stream:stream>, MUST appear in a message by itself.

3.4. Stream Errors

Stream level errors in XMPP are terminal. Should such an error occur, the server MUST send the stream error as a complete element in a message to the client.

If the error occurs during the opening of a stream, the stream error message MUST start with an opening stream tag (see Section 4.7.1 of [RFC6120]) and end with a closing stream tag.

After the stream error and closing stream tag have been sent, the server MUST close the connection as in Section 3.5.

3.5. Closing the Connection

Either the server or the client may close the connection at any time. Before closing the connection, the closing party MUST close the XMPP stream if it has been established. To initiate the close, the closing party MUST send a normal WebSocket close message with an empty body. The connection is considered closed when a matching close message is received (see Section 1.4 of [RFC6455]).

Except in the case of certain stream errors (see Section 3.4), the closing stream tag, </stream:stream>, MUST appear in a message by itself.

3.6. Stanzas

Each XMPP stanza MUST be sent in its own message. A stanza MUST NOT be split over multiple messages. All first level children of the <stream:stream> element MUST be treated the same as stanzas (e.g. <stream:features> and <stream:error>).

3.7. Stream Restarts

After successful SASL authentication, an XMPP stream needs to be restarted. In these cases, as soon as the message is sent (or received) containing the success indication, both the server and client streams are implicitly closed, and new streams needs to be opened. The client MUST open a new stream as in Section 3.3 and MUST NOT send a closing stream tag.

3.8. Pings and Keepalives

XMPP servers send whitespace pings as keepalives between stanzas, and XMPP clients can do the same thing. These extra whitespace characters are not significant in the protocol. Servers and clients SHOULD use WebSocket ping messages instead for this purpose.

The XMPP Ping extension [XEP-0199] allows entities to send and respond to ping requests. A client sending a WebSocket ping is equivalent to pinging the WebSocket server, which may also be the XMPP server. When the XMPP server is not also the WebSocket server, a WebSocket ping may be useful to check the health of the intermediary server.

3.9. Use of TLS

TLS cannot be used at the XMPP sub-protocol layer because the sub-protocol does not allow for raw binary data to be sent. Instead, enabling TLS SHOULD be done at the WebSocket layer using secure WebSocket connections via the `wss` URI scheme. (See Section 10.6 of [RFC6455]).

Because TLS is to be provided outside of the XMPP sub-protocol layer, a server MUST NOT advertise TLS as a stream feature (see Section 4.6 of [RFC6120]), and a client MUST ignore any advertised TLS stream feature, when using the XMPP sub-protocol.

3.10. Stream Management

Implications of, and recommendation to use, the XMPP Stream Management extension [XEP-0198] to be added.

4. Examples

Examples will be added as soon as the WebSocket protocol specification is more stable.

5. Security Considerations

Since application level TLS cannot be used (see Section 3.9), applications which need to protect the privacy of the XMPP traffic need to do so at the WebSocket or other appropriate layer.

The Security Considerations for both WebSocket (See Section 10 of [RFC6455] and XMPP (See Section 13 of [RFC6120]) apply to the WebSocket XMPP sub-protocol.

6. IANA Considerations

This specification requests IANA to register the WebSocket XMPP sub-protocol under the "WebSocket Subprotocol Name" Registry with the following data:

Subprotocol Identifier: xmpp

Subprotocol Common Name: WebSocket Transport for the Extensible Messaging and Presence Protocol (XMPP)

Subprotocol Definition: RFC XXXX

[[NOTE TO RFC EDITOR: Please change XXXX to the number assigned to this document upon publication.]]

7. Informative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC6120] Saint-Andre, P., "Extensible Messaging and Presence Protocol (XMPP): Core", RFC 6120, March 2011.
- [RFC6121] Saint-Andre, P., "Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence", RFC 6121, March 2011.
- [RFC6202] Loreto, S., Saint-Andre, P., Salsano, S., and G. Wilkins, "Known Issues and Best Practices for the Use of Long Polling and Streaming in Bidirectional HTTP", RFC 6202, April 2011.
- [RFC6455] Fette, I. and A. Melnikov, "The WebSocket Protocol", RFC 6455, December 2011.
- [W3C.REC-xml-20081126] Sperberg-McQueen, C., Yergeau, F., Paoli, J., Bray, T., and E. Maler, "Extensible Markup Language (XML) 1.0 (Fifth Edition)", World Wide Web Consortium Recommendation REC-

xml-20081126, November 2008,
<<http://www.w3.org/TR/2008/REC-xml-20081126>>.

[XEP-0124]

Paterson, I., Smith, D., Saint-Andre, P., and J. Moffitt,
"Bidirectional-streams Over Synchronous HTTP (BOSH)", XSF
XEP 0124, July 2010.

[XEP-0198]

Karneges, J., Saint-Andre, P., Hildebrand, J., Forno, F.,
Cridland, D., and M. Wild, "Stream Management", XSF XEP
0198, June 2011.

[XEP-0199]

Saint-Andre, P., "XMPP Ping", XSF XEP 0199, June 2009.

[XEP-0206]

Paterson, I. and P. Saint-Andre, "XMPP Over BOSH", XSF XEP
0206, July 2010.

Authors' Addresses

Lance Stout (editor)
&yet

Email: lance@andyet.net

Jack Moffitt

Email: jack@metajack.im

Eric Cestari
ProcessOne

Email: ecestari@process-one.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 26, 2013

P. Saint-Andre
M. Miller
Cisco Systems, Inc.
February 22, 2013

Domain Name Associations (DNA) in the Extensible Messaging and Presence
Protocol (XMPP)
draft-saintandre-xmpp-dna-01

Abstract

This document improves the security of the Extensible Messaging and Presence Protocol (XMPP) in two ways. First, it specifies how "prooftypes" can establish a strong association between a domain name and an XML stream. Second, it describes how to securely delegate a source domain to a derived domain, which is especially important in virtual hosting environments.

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 26, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terminology	3
3. Flow Chart	3
4. A Simple Scenario	6
5. One-Way Authentication	7
6. Piggybacking	7
6.1. Assertion	7
6.2. Supposition	9
7. Alternate Proofotypes	10
8. Virtual Hosting	11
9. Proofotype Model	11
10. Security Considerations	12
11. IANA Considerations	12
12. References	12
12.1. Normative References	12
12.2. Informative References	13
Authors' Addresses	14

1. Introduction

The need to establish a strong association between a domain name and an XML stream arises in both client-to-server and server-to-server communication using the Extensible Messaging and Presence Protocol (XMPP), because XMPP servers are typically identified by DNS domain names. However, a client or peer server needs to verify the identity of a server to which it connects. 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. This document (1) generalizes the model currently in use so that additional proofotypes can be defined, (2) provides a basis for modernizing some proofotypes to reflect progress in underlying technologies such as DNS Security [RFC4033], and (3) describes the flow of operations for establishing a domain name association.

Furthermore, the process for resolving the domain name of an XMPP service into the IP address at which an XML stream will be negotiated (defined in [RFC6120]) can involve delegation of a source domain (say, example.com) to a derived domain (say, hosting.example.net). If such delegation is not done in a secure manner, then the domain name association cannot be authenticated. Therefore, this document provides guidelines for defining secure delegation methods.

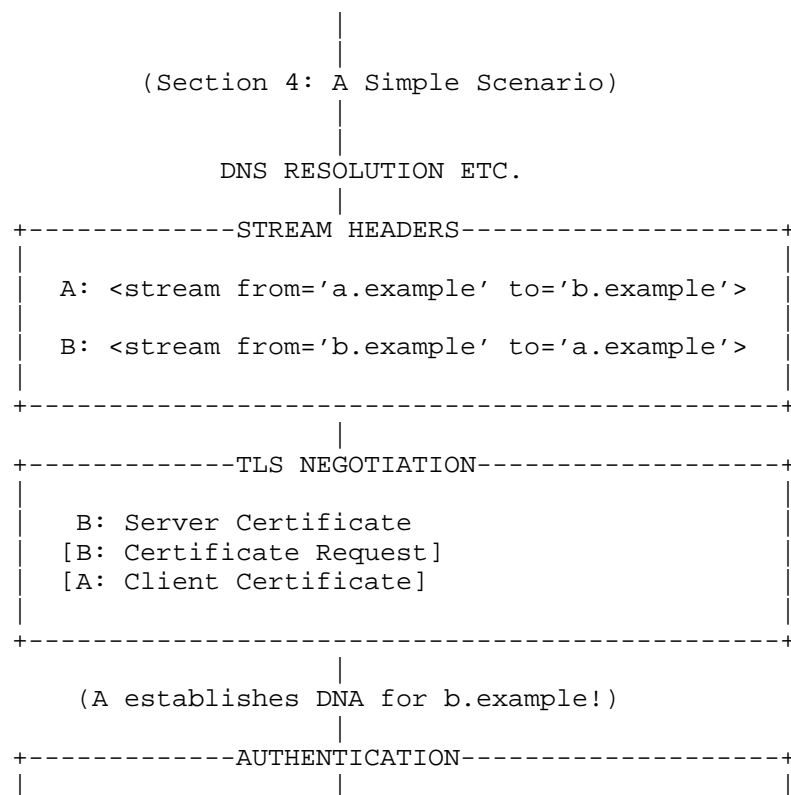
This document does not define any DNA proofypes or secure delegation methods; such technologies are defined in companion documents.

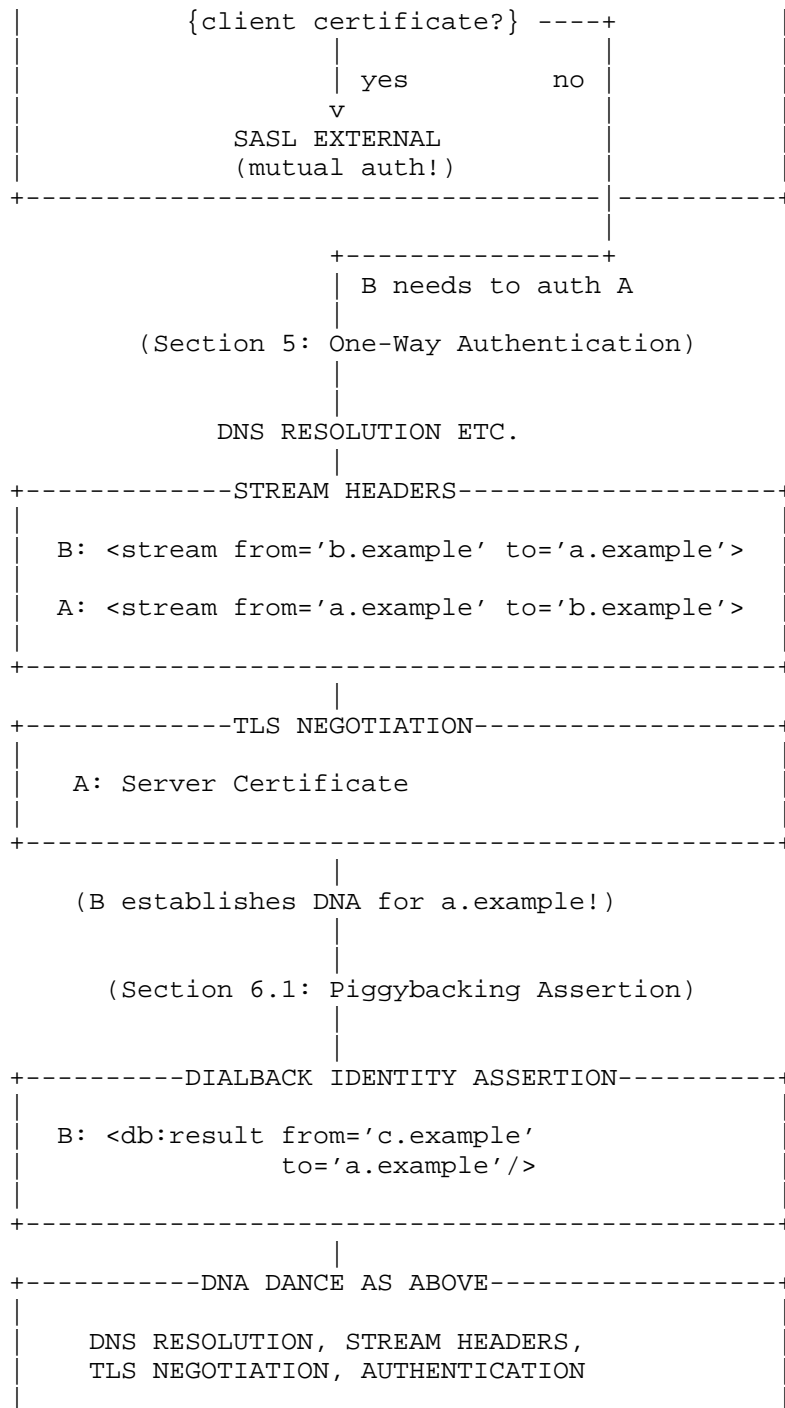
2. Terminology

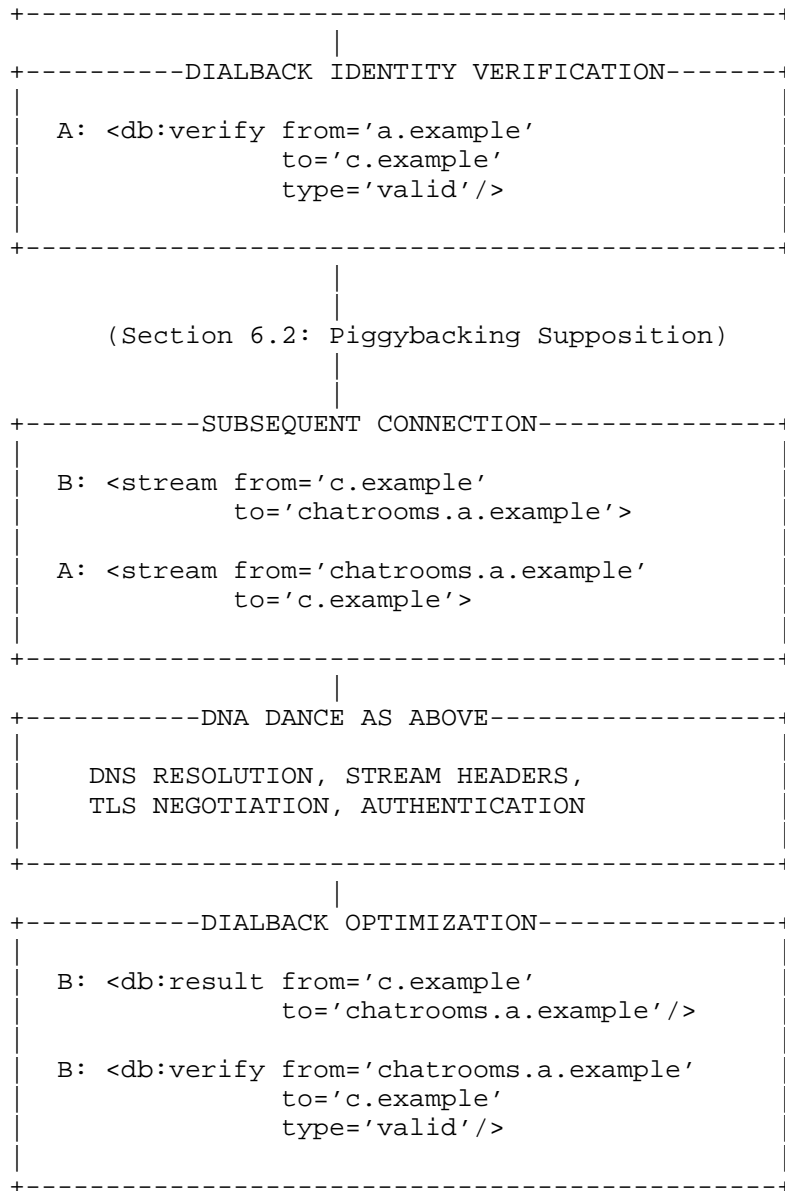
This document inherits XMPP terminology from [RFC6120] and [XEP-0220], DNS terminology from [RFC1034], [RFC1035], [RFC2782] and [RFC4033], and security terminology from [RFC4949] and [RFC5280]. The terms "source domain", "derived domain", "reference identity", and "presented identity" are used as defined in the "CertID" specification [RFC6125]. The terms "permissive federation", "verified federation", and "encrypted federation" are derived from [XEP-0238], although we substitute the term "authenticated federation" for the term "trusted federation" from that document.

3. Flow Chart

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







4. A Simple 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 the DNS domain name 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], 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:

- o Server A presents a PKIX certificate during TLS negotiation, which enables the parties to complete mutual authentication.
- o There are no additional domains associated with Server A and Server B (say, a subdomain chatrooms.a.example on Server A or a second domain c.example on Server B).
- o The server administrators are able to obtain PKIX certificates in the first place.
- o The server administrators are running their own XMPP servers, rather than using hosting services.

Let's consider each of these "wrinkles" in turn.

5. One-Way Authentication

If Server A does not present its PKIX certificate during TLS negotiation (perhaps because it wishes to verify the identity of Server B before presenting its own credentials), Server B is unable to mutually authenticate Server A. Therefore, Server B needs to negotiate and authenticate a stream to Server A, just as Server A has done:

1. Server B resolves the DNS domain name a.example.
2. Server B opens a TCP connection to the resolved IP address.
3. Server B sends an initial stream header to Server A, asserting that it is b.example:

```
<stream:stream from='b.example' to='a.example'>
```
4. Server A sends a response stream header to Server B, asserting that it is a.example:

```
<stream:stream from='a.example' to='b.example'>
```
5. The servers attempt TLS negotiation, during which Server A (acting as a TLS server) presents a PKIX certificate proving that it is a.example.
6. Server B checks the PKIX certificate that Server A provided; if it is consistent with the XMPP profile of the matching rules from [RFC6125], Server B accepts that there is a strong domain name association between its stream to Server A and the DNS domain name a.example.

Unfortunately, now 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.

6. Piggybacking

6.1. Assertion

Consider the common scenario in which Server B hosts not only b.example but also a second domain c.example. 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. 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 [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. It can do the same thing that it did before:

1. Server A resolves the DNS domain name 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. Server A checks the PKIX certificate that Server B provided; if it is consistent with the XMPP profile of the matching rules from [RFC6125], Server A accepts that there is a strong domain name association between its stream to Server B and the DNS domain name c.example.

Now that Server A accepts the domain name association, it informs Server B of that fact by sending verification of the Server Dialback key over the original connection:

```
<db:verify 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 over the same IP:port combination with the domain name c.example (dialback key links the original stream to the new association).

6.2. 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 at chatrooms.a.example (see [XEP-0045]). If a user from c.example at Server B wishes to join a room on the groupchat service, Server B needs to send XMPP stanzas from the domain c.example to the domain chatrooms.a.example rather than a.example. Therefore, Server B needs to negotiate and authenticate a stream to chatrooms.a.example:

1. Server B resolves the DNS domain name chatrooms.a.example.
2. Server B opens a TCP connection to the resolved IP address.
3. Server B sends an initial stream header to Server A acting as chatrooms.a.example, asserting that it is b.example:

```
<stream:stream from='b.example' to='chatrooms.a.example'>
```
4. Server A sends a response stream header to Server B, asserting that it is chatrooms.a.example:

```
<stream:stream from='chatrooms.a.example' to='b.example'>
```
5. The servers attempt TLS negotiation, during which Server A (acting as a TLS server) presents a PKIX certificate proving that it is chatrooms.a.example.
6. Server B checks the PKIX certificate that Server A provided; if it is consistent with the XMPP profile of the matching rules from [RFC6125], Server B accepts that there is a strong domain name association between its stream to Server A and the DNS domain name chatrooms.a.example.

As before, the parties now have two TCP connections open. So that they can close the now-redundant connection, Server B sends a dialback key to Server A over the new connection.

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

Server A then informs Server B that it accepts the domain name association by sending verification of the dialback key over the original connection:

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

Server B can now close the connection over which it tested the domain name association for chatrooms.a.example.

7. Alternate Proofypes

The foregoing protocol flows assumed that domain name associations were proved using the standard PKI proofype specified in [RFC6120]: that is, the server's proof consists of a PKIX certificate that is checked according to a profile of the matching rules from [RFC6125], 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 their servers (e.g., the administrator of im.cs.podunk.example can't receive certification authority verification messages sent to `mailto:hostmaster@podunk.example`, or `hosting.example.net` does not want to take on the liability of holding the certificate and private key for `example.com`). In these circumstances, proofypes other than PKIX are desirable. Two companion documents, [XMPP-DANE] and [XMPP-POSH], define alternate proofypes:

- o In the DANE proofype, the server's proof consists of a PKIX certificate that is compared as an exact match or a hash of either the `SubjectPublicKeyInfo` or the full certificate, and the client's verification material is obtained via secure DNS. See the accompanying [XMPP-DANE] spec for discussion and examples.
- o In the POSH (PKIX Over Secure HTTP) proofype, the server's proof consists of a PKIX certificate that is checked according to the rules from [RFC6120] and [RFC6125], the client's verification material is obtained by retrieving the PKIX certificate over HTTPS

at a well-known URI [RFC5785], and secure DNS is not necessary since the HTTPS retrieval mechanism relies on the chain of trust from the public key infrastructure. See the accompanying [XMPP-POSH] spec for discussion and examples.

8. Virtual Hosting

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

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

To improve security and limit liability, in typical deployments the administrators of hosting.example.net do not wish to hold the certificate and private key for example.com and the owners of example.com do not wish to share their certificate and private key with the administrators of hosting.example.net. In practice this means that 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 an unencrypted connection). This is also true of client-to-server communications, forcing end users to override certificate warnings or configure their clients to accept certificates for hosting.example.net instead of example.com. The fundamental problem here is that if DNSSEC is not used the act of delegation is inherently insecure.

This document does not describe how to achieve secure delegation. However, [XMPP-DANE] explains how to use DNSSEC for secure delegation in the PKI and DANE proofypes and [XMPP-POSH] explains how to use HTTPS redirects for secure delegation in the POSH proofotype.

9. Proofype Model

In general, a DNA proofype conforms to the following definition:

proofype: 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 PKI, DANE, and POSH prooftypes adhere to this model. In addition, other prooftypes are possible (examples might include PGP keys rather than PKIX certificates, or a token mechanism such as Kerberos or OAuth).

Some prooftypes depend on (or are enhanced by) secure DNS and therefore also need to describe how secure delegation occurs for that proftype.

10. Security Considerations

This document supplements but does not supersede the security considerations provided in [RFC6120] and [RFC6125].

11. IANA Considerations

This document has no actions for the IANA.

12. References

12.1. Normative References

- [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, May 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.
- [XEP-0220] Miller, J., Saint-Andre, P., and P. Hancke, "Server Dialback", XSF XEP 0220, August 2011.
- [XMPP-DANE] Miller, M. and P. Saint-Andre, "Using DNS Security Extensions (DNSSEC) and DNS-based Authentication of Named Entities (DANE) as a Proofotype for XMPP Domain Name Associations", draft-miller-xmpp-dnssec-proofotype-04 (work in progress), February 2013.
- [XMPP-POSH] Miller, M. and P. Saint-Andre, "Using PKIX over Secure HTTP (POSH) as a Proofotype for XMPP Domain Name Associations", draft-miller-xmpp-posh-proofotype-03 (work in progress), February 2013.

12.2. Informative References

- [RFC3920] Saint-Andre, P., Ed., "Extensible Messaging and Presence Protocol (XMPP): Core", RFC 3920, October 2004.
- [XEP-0045] Saint-Andre, P., "Multi-User Chat", XSF XEP 0045, February 2012.
- [XEP-0238] Saint-Andre, P., "XMPP Protocol Flows for Inter-Domain Federation", XSF XEP 0238, March 2008.
- [XEP-0288] Hancke, P. and D. Cridland, "Bidirectional Server-to-Server Connections", XSF XEP 0288, August 2012.

Authors' Addresses

Peter Saint-Andre
Cisco Systems, Inc.
1899 Wynkoop Street, Suite 600
Denver, CO 80202
USA

Email: psaintan@cisco.com

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

Email: mamille2@cisco.com