XEP-0219: Hop Check

Peter Saint-Andre
mailto:xsf@stpeter.im
xmpp:peter@jabber.org
http://stpeter.im/

2008-06-12
Version 0.3

Status	Type	Short Name
Retracted	Standards Track	TO BE ASSIGNED

This document defines an XMPP protocol extension that enables an entity to check the security of client-to-server and server-to-server hops between it and another entity. Note: This specification has been retracted by the author because the problem is not compelling and a real solution would be too complicated.

Legal

Copyright

Permissions

Warranty

Liability

Conformance

Contents

1 Introduction

Note: This specification has been retracted by the author because the problem is not compelling and a real solution would be too complicated.

1.1 Motivation

When two XMPP users communicate, one or both of them may want to know the status of the XMPP communications path (i.e., of all the hops) between them. While the primary motivation is to determine if all the hops are secured via Transport Layer Security (see RFC 5246 [1]) as specified for XMPP in XMPP Core [2], more general information about the communications path may also be of interest.
This document describes a protocol for discovering such information, similar in spirit to the traceroute protocol specified in RFC 1393 [3] but specific to XMPP.

1.2 Approach

As a user, I may want to know three things:

1. If my connection to my server is encrypted.
2. If my server's connection to my contact's server is encrypted.
3. If my contact's connection to their server is encrypted.

If the answer to all three of these questions is "yes" and I have some level of trust in my XMPP server (see Security Considerations) then I have some assurance that communications between me and my contact will not be subject to interception by the likes of "Eve" (a passive attacker who can eavesdrop on messages) and "Mallory" (an active attacker who can maliciously modify messages in transit).
Note: By "encrypted" is meant channel encryption with Transport Layer Security (or, for client-to-server connections, legacy SSL) using something other than the null cipher. This protocol does not include more sophisticated information about the connection, such as the cipher suite used, the SASL authentication mechanism used (if any), the certificate identity (if any), the trusted roots involved in certificate authenticate, etc.
The answer to the first question is straightforward, since my client knows if its connection to my server is encrypted (as well as the certificates involved).
If I trust my server, I can ask it to report on the security of its connection to my contact's server. (Note: Because server-to-server connections are unidirectional and it is possible for the connection in one direction to be encrypted but for the connection in the opposite

[1] RFC 5246: The Transport Layer Security (TLS) Protocol Version 1.2 <http://tools.ietf.org/html/rfc5246>.
[2] RFC 6120: Extensible Messaging and Presence Protocol (XMPP): Core <http://tools.ietf.org/html/rfc6120>.
[3] RFC 1393: Traceroute Using an IP Option <http://tools.ietf.org/html/rfc1393>.

direction to be unencrypted, my server MUST NOT report the connection to my contact's server as encrypted unless the connection is encrypted in both directions.)
If the server-to-server connection is encrypted and my server trusts that connection, my server can ask my contact's server to report on the security of my contact's connection.
By feeling my way along the hops in this manner, I can learn if all three hops are encrypted and therefore can have some assurance that the entire communications path is encrypted.

1.3 How It Works

Because XMPP has a decentralized architecture, there may be multiple hops along the communications path. Consider two users Juliet at capulet.lit and Romeo at montague.lit. [4] There is one hop from Juliet's client to the capulet.lit server, one hop from the capulet.lit server to the montague.lit server, and one hop from the montague.lit server to Romeo's client. [5]
We assume that Juliet's connection to capulet.lit is encrypted and that she has some trust in her server. If she wants to check the security of the path between her and Romeo, she should would send the following request:

Listing 1: Hop check request

```
<iq type='get'
    from='juliet@capulet.lit/balcony'
    to='capulet.lit'
    id='check1'>
  <hopcheck xmlns='http://www.xmpp.org/extensions/xep-0219.html#ns'
            to='romeo@montague.lit/orchard'/>
</iq>
```

Naturally, capulet.lit knows about the security of Juliet's connection to capulet.lit and about the security of its connection to montague.lit. But it does not know about the security of Romeo's connection to montague.net; therefore on behalf of Juliet it needs to ask montague.lit about that hop before returning an answer to Juliet:

Listing 2: Intermediate hop check request

```
<iq type='get'
    from='capulet.lit'
    to='montague.lit'
    id='check2'>
  <hopcheck xmlns='http://www.xmpp.org/extensions/xep-0219.html#ns'
```

[4] This specification uses the example of two clients, but the protocol can be used to check hops between any two XMPP entities.

[5] If both capulet.lit and montague.lit are virtual hosts of the same server, it is possible that there are only two hops in this situation rather than three. Moreover, in certain circumstances (e.g., if one of the clients connects to a third-party HTTP connection manager) it is possible for there to be more than three hops. However, this document assumes a standard XMPP architecture.

```
            for='juliet@capulet.lit/balcony'
            to='romeo@montague.lit/orchard'/>
</iq>
```

The montague.lit server would then report to capulet.lit regarding the hops from juliet@capulet.lit to romeo@montague.lit/orchard:

Listing 3: Intermediate hop check result

```
<iq type='result'
    from='montague.lit'
    to='juliet@capulet.lit'
    id='check2'>
  <hopcheck xmlns='http://www.xmpp.org/extensions/xep-0219.html#ns'
            for='juliet@capulet.lit/balcony'
            to='romeo@montague.lit/orchard'>
    <hop from='capulet.lit'
         to='montague.lit'
         ip='192.0.2.1'
         delay='7.178'
         auth='EXTERNAL'
         encrypted='1'/>
    <hop from='montague.lit'
         to='romeo@montague.lit/orchard'
         delay='15.734'
         auth='PLAIN'
         encrypted='1'/>
  </hopcheck>
</iq>
```

The capulet.lit server would then report to Juliet regarding the security of all the hops:

Listing 4: Hop check result

```
<iq type='result'
    from='capulet.lit'
    to='juliet@capulet.lit/balcony'
    id='check1'>
  <hopcheck xmlns='http://www.xmpp.org/extensions/xep-0219.html#ns'
            from='juliet@capulet.lit/balcony'
            to='romeo@montague.lit/orchard'>
    <hop from='juliet@capulet.lit/balcony'
         to='capulet.lit'
         auth='DIGEST-MD5'
         encrypted='true'/>
    <hop from='capulet.lit'
         to='montague.lit'
         ip='192.0.2.1'
         delay='11.602'
```

```
        auth='EXTERNAL'
        encrypted='true'/>
    <hop from='montague.lit'
        to='romeo@montague.lit/orchard'
        delay='15.734'
        auth='PLAIN'
        encrypted='1'/>
  </hopcheck>
</iq>
```

The foregoing shows only the "happy path"; many alternate flows and error conditions are possible, as described in the Protocol section of this document.

2 Protocol

2.1 Request

In order for an entity to check the communications path between itself and a target entity, the entity shall generate a hopcheck request of the following form:

```
<iq type='get'
    from='sender'
    to='respondent'
    id='foo'>
  <hopcheck xmlns='http://www.xmpp.org/extensions/xep-0219.html#ns'
            [for='original-sender']
            to='target'/>
</iq>
```

The sender MUST set the respondent to be the entity that the sender believes is the entity at the other end of the next hop in the communications path. (For a connected resource, this is the server to which the client is connected; for a server, this is the server corresponding to the domain portion of the JID in the triggering request received from a connected resource or other local entity.)
If the sender is a server but it is making the request on behalf of a connected resource, it MUST include the 'for' attribute and set it to the 'from' address of the triggering request.

2.2 Errors

The following error cases are defined:

- If the respondent does not support the hopcheck namespace, it MUST return a <service-unavailable/> error.

- If the <hopcheck/> element does not include a 'to' attribute, the respondent MUST return a <bad-request/> error.
- If the 'to' or 'for' attribute of the <hopcheck/> element contains a JID that is malformed, the respondent MUST return a <jid-malformed/> error.
- If the respondent is a server and the sender is a full JID <local-part@domain.tld/resource>, the server MUST verify that the JID is a connected resource of the server (per RFC 6120); if it is not, then the respondent MUST return a <forbidden/> error.
- If both the sender and the respondent are servers and the domain identifier of the <hopcheck/> element's 'to' attribute does not match a validated domain of the respondent, the respondent MUST return a <item-not-found/> error.
- If both the sender and the respondent are servers, the JID of the <hopcheck/> element's 'to' is a connected resource of the respondent server, and the JID of the <hopcheck/> element's 'for' is not authorized to know the presence of the target (either via presence subscription or temporary presence sharing via directed presence as defined in XMPP IM [6]), then the respondent MUST return a <forbidden/> error.

2.3 Response

If an error does not occur, the response shall be of the following form:

```
<iq type='result'
    from='respondent'
    to='sender'
    id='foo'>
  <hopcheck xmlns='http://www.xmpp.org/extensions/xep-0219.html#ns'
            [for='original-sender']
            to='target'>
    <hop auth='some-defined-value'
         [delay='float']
         from='sender-or-respondent'
         [ip='some-IP-address']
         encrypted='boolean'
         to='respondent-or-target'/>
  </hopcheck>
</iq>
```

The 'delay' attribute is OPTIONAL. If included, it represents the time in milliseconds that it took for the respondent to receive a reply from the target or intermediate respondent.
The 'ip' attribute is OPTIONAL. If included, it represents the IPv4 or IPv6 address of the target or intermediate respondent.

[6] RFC 6121: Extensible Messaging and Presence Protocol (XMPP): Instant Messaging and Presence <http://tools.ietf.org/html/rfc6121>.

The 'encrypted' attribute is REQUIRED. It is a boolean [7] that represents whether the hop in question is encrypted via Transport Layer Security (see RFC 5246 [8]) or at a legacy SSL port.
The 'auth' attribute is REQUIRED. It represents whether and how the hop in question is authenticated, in particular via one of the following means:

- A Simple Authentication and Security Layer mechanism (see RFC 4422 [9] for the specification and IANA SASL Mechanisms Registry [10] for a list of registered SASL mechanisms; the names for standardized, non-obsolete mechanisms are used as values of the 'auth' attribute).
- The obsolete Non-SASL Authentication (XEP-0078) [11] protocol using the "plaintext" or "digest" method.
- The server dialback protocol ("dialback") as specified in Server Dialback (XEP-0220) [12].

3 Discovering Support

If an entity supports the Hop Check protocol, it MUST report that by including a service discovery feature of "http://www.xmpp.org/extensions/xep-0219.html#ns" (see Protocol Namespaces regarding issuance of one or more permanent namespaces) in response to a Service Discovery (XEP-0030) [13] information request:

Listing 5: Service Discovery information request

```
<iq type='get'
    from='juliet@capulet.lit/balcony'
    from='capulet.lit'
    id='disco1'>
  <query xmlns='http://jabber.org/protocol/disco#info'/>
</iq>
```

Listing 6: Service Discovery information response

```
<iq type='result'
    from='capulet.lit'
    to='juliet@capulet.lit/balcony'
```

[7] In accordance with Section 3.2.2.1 of XML Schema Part 2: Datatypes, the allowable lexical representations for the xs:boolean datatype are the strings "0" and "false" for the concept 'false' and the strings "1" and "true" for the concept 'true'; implementations MUST support both styles of lexical representation.
[8] RFC 5246: The Transport Layer Security (TLS) Protocol Version 1.2 <http://tools.ietf.org/html/rfc5246>.
[9] RFC 4422: Simple Authentication and Security Layer (SASL) <http://tools.ietf.org/html/rfc4422>.
[10] IANA registry of mechanisms used in the Simple Authentication and Security Layer protocol <http://www.iana.org/assignments/sasl-mechanisms>.
[11] XEP-0078: Non-SASL Authentication <https://xmpp.org/extensions/xep-0078.html>.
[12] XEP-0220: Server Dialback <https://xmpp.org/extensions/xep-0220.html>.
[13] XEP-0030: Service Discovery <https://xmpp.org/extensions/xep-0030.html>.

```
    id='disco1'>
  <query xmlns='http://jabber.org/protocol/disco#info'>
    ...
    <feature var='http://www.xmpp.org/extensions/xep-0219.html#ns'/>
    ...
  </query>
</iq>
```

4 Security Considerations

4.1 Server Trust

Trust levels vary. The administrator of a server may have complete trust in the server they run. A user of a friend's server may have significant trust in the server. A user of a server in a military environment may have no choice of server and may trust that their superiors are doing the right thing. And so on. While the trust that a user places in a server may or may not be warranted, this document uses the level of trust that exists to bootstrap greater trust in the entire communications path.

4.2 Cleartext

Even if all the hops are encrypted, the traffic may still be available in cleartext to a representative of an Internet Service Provider ("Isaac"), a representative of the justice system ("Justin"), or a malicious attacker ("Mallory") that has gained access to any of the servers along the communications path. The protocol described herein is decidedly NOT a substitute for end-to-end encryption technologies such as Encrypted Session Negotiation (XEP-0116) [14].

5 IANA Considerations

This document requires no interaction with the Internet Assigned Numbers Authority (IANA) [15].

[14] XEP-0116: Encrypted Session Negotiation <https://xmpp.org/extensions/xep-0116.html>.

[15] The Internet Assigned Numbers Authority (IANA) is the central coordinator for the assignment of unique parameter values for Internet protocols, such as port numbers and URI schemes. For further information, see <http://www.iana.org/>.

6 XMPP Registrar Considerations

6.1 Protocol Namespaces

Until this specification advances to a status of Draft, its associated namespace shall be "http://www.xmpp.org/extensions/xep-0219.html#ns"; upon advancement of this specification, the XMPP Registrar [16] shall issue a permanent namespace in accordance with the process defined in Section 4 of XMPP Registrar Function (XEP-0053) [17].

7 XML Schema

```
<?xml version='1.0' encoding='UTF-8'?>

<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
    targetNamespace='http://www.xmpp.org/extensions/xep-0219.html#ns'
    xmlns='http://www.xmpp.org/extensions/xep-0219.html#ns'
    elementFormDefault='qualified'>

  <xs:element name='hopcheck'>
    <xs:complexType>
      <xs:sequence minOccurs='0' maxOccurs='unbounded'>
        <xs:element ref='hop'/>
      </xs:sequence>
      <xs:attribute name='for' type='xs:string' use='optional'/>
      <xs:attribute name='to' type='xs:string' use='required'/>
    </xs:complexType>
  </xs:element>

  <xs:element name='hop'>
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base='empty'>
          <xs:attribute name='auth' use='required'>
            <xs:simpleType>
              <xs:restriction base='xs:NCName'>
                <xs:enumeration value='ANONYMOUS'/>
                <xs:enumeration value='CRAM-MD5'/>
                <xs:enumeration value='DIGEST-MD5'/>
                <xs:enumeration value='EXTERNAL'/>
                <xs:enumeration value='GSSAPI'/>
                <xs:enumeration value='KERBEROS_V4'/>
```

[16] The XMPP Registrar maintains a list of reserved protocol namespaces as well as registries of parameters used in the context of XMPP extension protocols approved by the XMPP Standards Foundation. For further information, see <https://xmpp.org/registrar/>.

[17] XEP-0053: XMPP Registrar Function <https://xmpp.org/extensions/xep-0053.html>.

```
                <xs:enumeration value='OTP'/>
                <xs:enumeration value='PLAIN'/>
                <xs:enumeration value='SECURID'/>
                <xs:enumeration value='dialback'/>
                <xs:enumeration value='digest'/>
                <xs:enumeration value='plaintext'/>
              </xs:restriction>
            </xs:simpleType>
          </xs:attribute>
          <xs:attribute name='delay' type='xs:double' use='optional'/>
          <xs:attribute name='encrypted' type='xs:boolean' use='
              required'/>
          <xs:attribute name='from' type='xs:string' use='required'/>
          <xs:attribute name='ip' type='xs:string' use='optional'/>
          <xs:attribute name='to' type='xs:string' use='required'/>
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>

  <xs:simpleType name='empty'>
    <xs:restriction base='xs:string'>
      <xs:enumeration value=''/>
    </xs:restriction>
  </xs:simpleType>

</xs:schema>
```