

XEP-0520: SASL2 Terms of Service Acceptance Task

Guus der Kinderen
mailto:guus.der.kinderen@gmail.com
xmpp:guus.der.kinderen@igniterealtime.org

2026-11-13
Version 0.1.0

Status	Type	Short Name
Experimental	Standards Track	NOT_YET_ASSIGNED

This specification defines a SASL2 task, per the extensibility mechanism of XEP-0388, that lets a server require a user to accept the current terms of service before authentication completes.

Legal

Copyright

This XMPP Extension Protocol is copyright © 1999 – 2024 by the XMPP Standards Foundation (XSF).

Permissions

Permission is hereby granted, free of charge, to any person obtaining a copy of this specification (the ”Specification”), to make use of the Specification without restriction, including without limitation the rights to implement the Specification in a software program, deploy the Specification in a network service, and copy, modify, merge, publish, translate, distribute, sublicense, or sell copies of the Specification, and to permit persons to whom the Specification is furnished to do so, subject to the condition that the foregoing copyright notice and this permission notice shall be included in all copies or substantial portions of the Specification. Unless separate permission is granted, modified works that are redistributed shall not contain misleading information regarding the authors, title, number, or publisher of the Specification, and shall not claim endorsement of the modified works by the authors, any organization or project to which the authors belong, or the XMPP Standards Foundation.

Warranty

NOTE WELL: This Specification is provided on an ”AS IS” BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, express or implied, including, without limitation, any warranties or conditions of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A PARTICULAR PURPOSE.

Liability

In no event and under no legal theory, whether in tort (including negligence), contract, or otherwise, unless required by applicable law (such as deliberate and grossly negligent acts) or agreed to in writing, shall the XMPP Standards Foundation or any author of this Specification be liable for damages, including any direct, indirect, special, incidental, or consequential damages of any character arising from, out of, or in connection with the Specification or the implementation, deployment, or other use of the Specification (including but not limited to damages for loss of goodwill, work stoppage, computer failure or malfunction, or any and all other commercial damages or losses), even if the XMPP Standards Foundation or such author has been advised of the possibility of such damages.

Conformance

This XMPP Extension Protocol has been contributed in full conformance with the XSF’s Intellectual Property Rights Policy (a copy of which can be found at <https://xmpp.org/about/xsf/ipr-policy> or obtained by writing to XMPP Standards Foundation, P.O. Box 787, Parker, CO 80134 USA).

Contents

1 Introduction 1

2 Requirements 1

3 Protocol 1

3.1 Determining Applicability . . . 2
3.2 Selecting the Task . . . 2
3.3 Presenting the Terms . . . 2
3.4 Responding to the Terms . . . 3
3.5 Outcome . . . 3

4 Business Rules 4

5 Security Considerations 4

6 IANA Considerations 5

7 XMPP Registrar Considerations 5

8 XML Schema 5

1 Introduction

Operators of XMPP services are sometimes required, for legal or policy reasons, to obtain a user's explicit acceptance of a terms-of-service document before letting that user access the service, and to be able to demonstrate that the currently-applicable version of that document was the one accepted.
Today this is typically handled outside of the XMPP authentication exchange itself, for example through an out-of-band web flow, or by nagging the user post-login with no guarantee they ever comply. Neither gives the server a reliable way to withhold a session from an account that has not accepted the current terms.
Extensible SASL Profile (XEP-0388) [1] defines a generic mechanism, the SASL2 "task" flow, by which a server can insert one or more additional steps between the completion of a SASL exchange and the completion of the SASL2 negotiation, using the <continue/>, <next/> and <task-data/> elements it defines. Nothing that marks a session as authenticated may occur until every such task has completed. This specification defines one concrete task, built on that mechanism, for the purpose of requiring acceptance of a versioned terms-of-service document.
This document defines only the task itself: its namespace, its elements, and the rules governing when it applies and how its outcome is determined. How a task is offered, selected, and driven to completion or failure in general is defined by Extensible SASL Profile (XEP-0388) [2].

2 Requirements

1. A server MUST be able to prevent an account from completing authentication until the current version of its terms of service has been accepted by that account.

2. A server SHOULD be able to determine, per account, whether the currently-applicable version has already been accepted, so that a user who has already accepted is not asked again.

3. A server MUST provide a human-readable explanation of why authentication has not yet completed, so that a client that does not implement this specific task, but does implement SASL2 authentication, can still inform its user rather than simply reporting a failed login.

3 Protocol

This specification defines the urn:xmpp:tos:0 namespace, used to qualify the elements defined below, and one SASL2 task, identified in <task/> and <next/> elements by the token TOS-ACCEPT.

[1] XEP-0388: Extensible SASL Profile <https://xmpp.org/extensions/xep-0388.html>.
[2] XEP-0388: Extensible SASL Profile <https://xmpp.org/extensions/xep-0388.html>.

3.1 Determining Applicability

Once the SASL exchange has succeeded, the server determines whether this task applies to the authenticating account, per the following rules (see also Business Rules):

1. There is no account against which an anonymous authentication's acceptance can be reliably recorded. A server that requires this task for anonymous authentication MUST therefore offer it on every anonymous authentication attempt, rather than only the first.
2. The task does not apply if it has already completed earlier in this same negotiation. This matters because eligibility is re-evaluated after every task completes, not just once, so a task that already ran is not offered again merely because a later round is reached.
3. Otherwise, the task applies if the account has not accepted the version that is currently applicable. Acceptance of a prior version does not count: what is tracked is acceptance of a specific version, not a bare yes/no flag.

When the task applies, the server offers it as it would any SASL2 task, in a <continue/> element. It MUST include human-readable <text/> explaining why authentication has not yet completed.

Listing 1: Server offers the terms task

```
<continue xmlns='urn:xmpp:sasl:2'>
  <tasks>
    <task>TOS-ACCEPT</task>
  </tasks>
  <text>The terms of service have changed and must be accepted before
     you can sign in.</text>
</continue>
```

3.2 Selecting the Task

Listing 2: Client selects the terms task

```
<next xmlns='urn:xmpp:sasl:2' task='TOS-ACCEPT'/>
```

3.3 Presenting the Terms

On selection, the server sends the current terms in a <task-data/> element, carrying a <terms/> element. Its content is the URL at which the terms can be read, and its version attribute identifies the version being offered for acceptance.

Listing 3: Server presents the current terms

```
<task-data xmlns='urn:xmpp:sasl:2'>
  <terms xmlns='urn:xmpp:tos:0' version='2026-01'>https://example.org/
      terms</terms>
</task-data>
```

3.4 Responding to the Terms

The client responds with a <task-data/> element of its own. To accept, it includes an <accept/> element whose version attribute repeats the version it was offered.

Listing 4: Client accepts the current terms

```
<task-data xmlns='urn:xmpp:sasl:2'>
  <accept xmlns='urn:xmpp:tos:0' version='2026-01'/>
</task-data>
```

A peer that does not wish to accept, or that does not implement this task at all, has no way to decline while continuing the negotiation: per Extensible SASL Profile (XEP-0388) [3], it can only abort, which fails the entire SASL2 negotiation and leaves the session unauthenticated.

Listing 5: Client aborts rather than accept

```
<abort xmlns='urn:xmpp:sasl:2'/>
```

3.5 Outcome

On receiving the client's response, the server determines the outcome per the following rules:

1. If the response contains no <accept/> element, the server MUST fail the SASL2 negotiation with a <not-authorized/> condition.
2. If the response contains an <accept/> element whose version attribute does not match the version that was offered in this round, the server MUST fail the SASL2 negotiation with a <malformed-request/> condition (rejecting, for example, a value cached from an earlier attempt).
3. Otherwise, the task completes for this negotiation. The server SHOULD persist the acceptance against the account, so that it is not asked again on a future connection. If the account is authenticating anonymously, or the acceptance cannot be persisted (for example due to a transient failure of the server's own backing store), no durable record results; the task still completes, but the server SHOULD offer it again on the account's next authentication attempt, since there is nothing on record to rely on next time.

[3] XEP-0388: Extensible SASL Profile <https://xmpp.org/extensions/xep-0388.html>.

Once every eligible task, including this one, has completed, the SASL2 negotiation concludes as usual with <success/>.

4 Business Rules

1. A server MUST NOT complete SASL2 authentication for an account for which this task applies (see Determining Applicability) unless the task has completed successfully during the same negotiation.

2. A server MUST offer this task on every anonymous authentication attempt for which it applies (see Determining Applicability), since acceptance cannot be reliably recorded against an account that does not exist.

3. A server SHOULD treat "currently accepted version" as a per-account fact rather than a per-session one: once durably recorded, the same account is not asked again on a later connection unless the applicable version changes or no durable record was actually made (see Outcome).

4. A server MAY treat acceptance obtained through a channel other than this task (for example a support interaction or a signed agreement) as satisfying acceptance of a version, provided it is recorded per-account in the same way. This task exists to enforce acceptance for accounts that have not already accepted through such a channel, not to be the sole route by which acceptance can be recorded.

5. A server SHOULD treat the version identifiers it issues as opaque and SHOULD compare them for exact equality, both when checking whether a previously-recorded acceptance is still current and when validating a peer's <accept/> against the version it was offered in the same round.

5 Security Considerations

This task is a policy and record-keeping mechanism, not an authentication factor. It MUST NOT be treated as contributing to the strength of the authentication that preceded it, and does not protect against an attacker who already holds valid credentials: by the time this task runs, the SASL exchange has already succeeded.
This task, like any SASL2 task, is only evaluated as part of a SASL2 negotiation. If a server also offers an authentication path that bypasses SASL2 entirely, for example the legacy SASL profile of RFC 6120, an account can authenticate through that path without this, or any other, SASL2 task ever being evaluated. A server that needs to guarantee acceptance before granting access MUST NOT offer such a path to accounts this task applies to, or MUST enforce the same requirement by some other means there. This limitation applies equally to any mandatory SASL2 task, not just this one.
A server that cannot durably persist an acceptance, for example due to a transient failure of

its own backing store, MAY still let the negotiation continue rather than fail it outright (see Outcome): the peer did explicitly accept during this negotiation, which is what the task exists to establish. Since no durable record exists in that case, the server SHOULD offer the task again on the account's next authentication attempt, so that a persistence failure results in the user being asked again rather than the requirement being silently and permanently bypassed. Because completion of this task changes an account's compliance record, a server SHOULD only offer, and only accept a response to, this task over a channel that satisfies its usual requirements for authenticated sessions, per whatever channel-binding or TLS policy already governs the SASL2 negotiation in which it is offered; this specification does not introduce a separate requirement.
The terms URL communicated in the <terms/> element is typically dereferenced by a human outside of the XMPP session (for example by opening it in a browser). Server operators should serve it over a secure and integrity-protected transport, but validating that is outside the scope of this specification.

6 IANA Considerations

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

7 XMPP Registrar Considerations

This specification defines the following XML namespace, which the XMPP Registrar shall add to its registry of protocol namespaces (XMPP Registrar Function (XEP-0053) [4]):

- urn:xmpp:tos:0

This specification also defines the SASL2 task token TOS-ACCEPT. Extensible SASL Profile (XEP-0388) [5] defines no central registry of task tokens; as with the UPGR-* tokens defined by SASL Upgrade Tasks (XEP-0480)[6], uniqueness is expected to follow from the token being sufficiently distinctive, checked against existing task-defining specifications during the XSF's normal review process.

8 XML Schema

```
<xs:schema
    xmlns:xs='http://www.w3.org/2001/XMLSchema'
```

[4] XEP-0053: XMPP Registrar Function <https://xmpp.org/extensions/xep-0053.html>.
[5] XEP-0388: Extensible SASL Profile <https://xmpp.org/extensions/xep-0388.html>.
[6] XEP-0480: SASL Upgrade Tasks <https://xmpp.org/extensions/xep-0480.html>.

```
    targetNamespace='urn:xmpp:tos:0'
    xmlns='urn:xmpp:tos:0'
    elementFormDefault='qualified'>

  <xs:annotation>
    <xs:documentation>
      The protocol documented by this schema is defined in
      XEP-xxxx: http://www.xmpp.org/extensions/xep-xxxx.html
    </xs:documentation>
  </xs:annotation>

  <xs:element name='terms'>
    <xs:annotation>
      <xs:documentation>
        Identifies the current terms of service. Sent by the server,
           as the
        content of a <task-data/> element, after this task has been
           selected.
      </xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:simpleContent>
        <xs:extension base='xs:anyURI'>
          <xs:attribute name='version' type='xs:string' use='required'
             />
        </xs:extension>
      </xs:simpleContent>
    </xs:complexType>
  </xs:element>

  <xs:element name='accept'>
    <xs:annotation>
      <xs:documentation>
        Sent by the client, as the content of a <task-data/> element,
           to accept
        the identified version of the terms of service.
      </xs:documentation>
    </xs:annotation>
    <xs:complexType>
      <xs:attribute name='version' type='xs:string' use='required'/>
    </xs:complexType>
  </xs:element>

</xs:schema>
```