XMPP

Florian “Florob” Zeitz

2012-09-27

Gliederung

1. Ich
2. XMPP 101: Basics
   - Geschichte
   - Netzstruktur
   - JID
   - Stanzas
3. XMPP 102: Sicherheit
   - SSL/TLS
   - Dialback
   - SASL
   - End-to-End Encryption

Gliederung

1. Ich
2. XMPP 101: Basics
   - Geschichte
   - Netzstruktur
   - JID
   - Stanzas
3. XMPP 102: Sicherheit
   - SSL/TLS
   - Dialback
   - SASL
   - End-to-End Encryption

Ich

- Florian Zeitz
- Elektrotechnik Student
- XSF Member seit Februar 2010
- aktiv in der XMPP IETF WG
- Mitarbeit am Prosody XMPP server (mod_adhoc, ...)
- Einer der Autoren von ObjXMPP
- Kleinere Features und Bugfixes in Gajim

Ich

- Florian Zeitz
- Elektrotechnik Student
- XSF Member seit Februar 2010
- aktiv in der XMPP IETF WG
- Mitarbeit am Prosody XMPP server (mod_adhoc, ...)
- Einer der Autoren von ObjXMPP
- Kleinere Features und Bugfixes in Gajim

Ich

- Florian Zeitz
- Elektrotechnik Student
- XSF Member seit Februar 2010
- aktiv in der XMPP IETF WG
- Mitarbeit am Prosody XMPP server (mod_adhoc, ...)
- Einer der Autoren von ObjXMPP
- Kleinere Features und Bugfixes in Gajim

Ich

- Florian Zeitz
- Elektrotechnik Student
- XSF Member seit Februar 2010
- aktiv in der XMPP IETF WG
- Mitarbeit am Prosody XMPP server (mod_adhoc, ...)
- Einer der Autoren von ObjXMPP
- Kleinere Features und Bugfixes in Gajim

Ich

- Florian Zeitz
- Elektrotechnik Student
- XSF Member seit Februar 2010
- aktiv in der XMPP IETF WG
- Mitarbeit am Prosody XMPP server (mod_adhoc, ...)
- Einer der Autoren von ObjXMPP
- Kleinere Features und Bugfixes in Gajim

Ich

- Florian Zeitz
- Elektrotechnik Student
- XSF Member seit Februar 2010
- aktiv in der XMPP IETF WG
- Mitarbeit am Prosody XMPP server (mod_adhoc, ...)
- Einer der Autoren von ObjXMPP
- Kleinere Features und Bugfixes in Gajim

Ich

- Florian Zeitz
- Elektrotechnik Student
- XSF Member seit Februar 2010
- aktiv in der XMPP IETF WG
- Mitarbeit am Prosody XMPP server (mod_adhoc, ...)
- Einer der Autoren von ObjXMPP
- Kleinere Features und Bugfixes in Gajim

Gliederung

1. Ich
2. XMPP 101: Basics
   - Geschichte
   - Netzstruktur
   - JID
   - Stanzas
3. XMPP 102: Sicherheit
   - SSL/TLS
   - Dialback
   - SASL
   - End-to-End Encryption

Überblick

- eXtensible Messaging and Presence Protocol
- Real time messaging and presence system
- verteilt
- XML basiert
- offener Standard

Überblick

- eXtensible Messaging and Presence Protocol
- Real time messaging and presence system
- verteilt
- XML basiert
- offener Standard

Überblick

- eXtensible Messaging and Presence Protocol
- Real time messaging and presence system
- verteilt
- XML basiert
- offener Standard

Überblick

- eXtensible Messaging and Presence Protocol
- Real time messaging and presence system
- verteilt
- XML basiert
- offener Standard

Überblick

- eXtensible Messaging and Presence Protocol
- Real time messaging and presence system
- verteilt
- XML basiert
- offener Standard

Warum

- Keine zentrale Instanz
- Von jedem verwend- und implementierbar
- Erweiterbarkeit

Warum

- Keine zentrale Instanz
- Von jedem verwend- und implementierbar
- Erweiterbarkeit

Warum

- Keine zentrale Instanz
- Von jedem verwend- und implementierbar
- Erweiterbarkeit

Gliederung

1. Ich
2. XMPP 101: Basics
   - Geschichte
   - Netzstruktur
   - JID
   - Stanzas
3. XMPP 102: Sicherheit
   - SSL/TLS
   - Dialback
   - SASL
   - End-to-End Encryption

Damals

- 1999:
  - Jeremie Miller startet das Jabber Projekt
  - Ziel: Freiheit von Gesprächen und offene Standards fördern
  - erster jabberd
  - erster Kontakt mit der IETF (IMPP WG)
- 2000:
  - jabberd 1.0 $\Rightarrow$ stabiles Protokol
- 2001:
  - XSF (damals noch JSF) wird gegründet
- 2002:
  - erste XMPP WG bei der IETF
- 2004:
  - RFC3920 XMPP Core und RFC3921 XMPP IM werden veröffentlicht

Damals

- 1999:
  - Jeremie Miller startet das Jabber Projekt
  - Ziel: Freiheit von Gesprächen und offene Standards fördern
  - erster jabberd
  - erster Kontakt mit der IETF (IMPP WG)
- 2000:
  - jabberd 1.0 $\Rightarrow$ stabiles Protokol
- 2001:
  - XSF (damals noch JSF) wird gegründet
- 2002:
  - erste XMPP WG bei der IETF
- 2004:
  - RFC3920 XMPP Core und RFC3921 XMPP IM werden veröffentlicht

Damals

- 1999:
  - Jeremie Miller startet das Jabber Projekt
  - Ziel: Freiheit von Gesprächen und offene Standards fördern
  - erster jabberd
  - erster Kontakt mit der IETF (IMPP WG)
- 2000:
  - jabberd 1.0 $\Rightarrow$ stabiles Protokol
- 2001:
  - XSF (damals noch JSF) wird gegründet
- 2002:
  - erste XMPP WG bei der IETF
- 2004:
  - RFC3920 XMPP Core und RFC3921 XMPP IM werden veröffentlicht

Damals

- 1999:
  - Jeremie Miller startet das Jabber Projekt
  - Ziel: Freiheit von Gesprächen und offene Standards fördern
  - erster jabberd
  - erster Kontakt mit der IETF (IMPP WG)
- 2000:
  - jabberd 1.0 $\Rightarrow$ stabiles Protokol
- 2001:
  - XSF (damals noch JSF) wird gegründet
- 2002:
  - erste XMPP WG bei der IETF
- 2004:
  - RFC3920 XMPP Core und RFC3921 XMPP IM werden veröffentlicht

Damals

- 1999:
  - Jeremie Miller startet das Jabber Projekt
  - Ziel: Freiheit von Gesprächen und offene Standards fördern
  - erster jabberd
  - erster Kontakt mit der IETF (IMPP WG)
- 2000:
  - jabberd 1.0 $\Rightarrow$ stabiles Protokol
- 2001:
  - XSF (damals noch JSF) wird gegründet
- 2002:
  - erste XMPP WG bei der IETF
- 2004:
  - RFC3920 XMPP Core und RFC3921 XMPP IM werden veröffentlicht

Heute

- Seit 2009 zweite XMPP WG der IETF
- 2011:
  - “bis” versionen veröffentlicht als RFC 6120-6122
- Goals and Milestones (Auszug):
  - Bis August 2012: Lösung für Ende-zu-Ende Verschlüsselung definieren
  - Bis September 2012: RFC 6122bis (Stringprep → PRECIS)

Heute

- Seit 2009 zweite XMPP WG der IETF
- 2011:
  - “bis” versionen veröffentlicht als RFC 6120-6122
- Goals and Milestones (Auszug):
  - Bis August 2012: Lösung für Ende-zu-Ende Verschlüsselung definieren
  - Bis September 2012: RFC 6122bis (Stringprep → PRECIS)

Heute

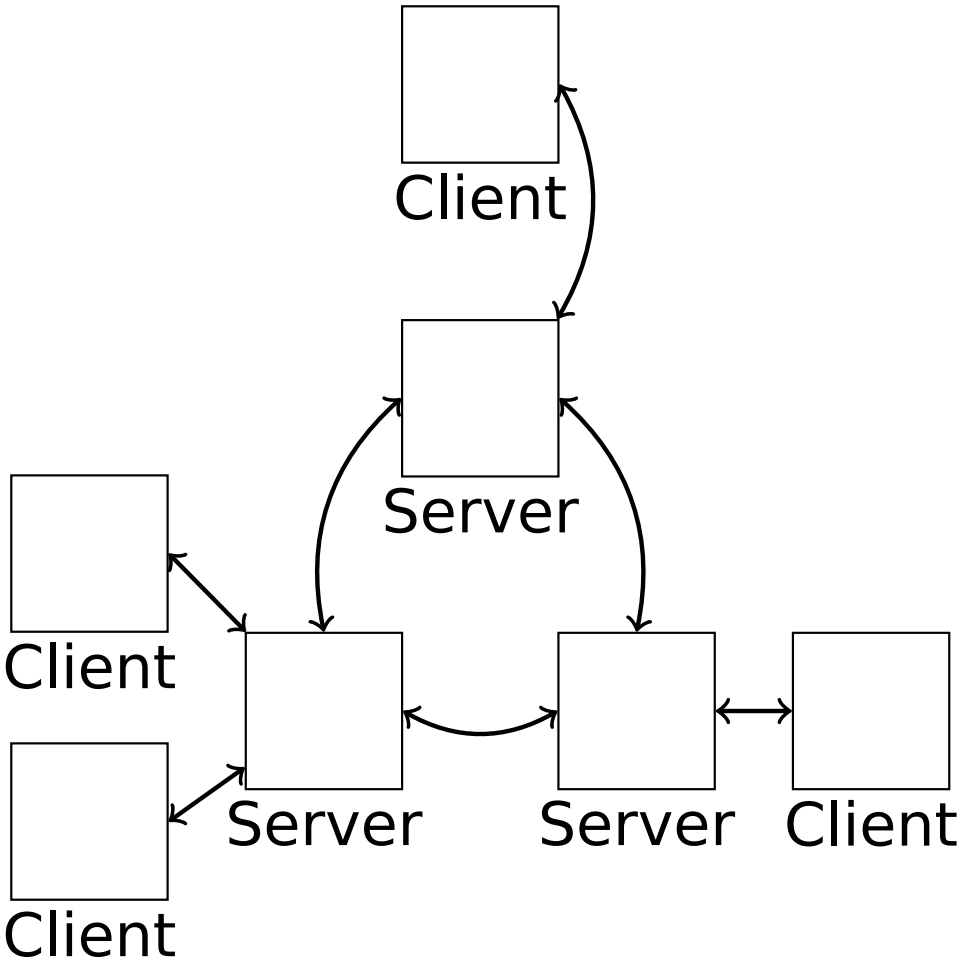
- Seit 2009 zweite XMPP WG der IETF
- 2011:
  - “bis” versionen veröffentlicht als RFC 6120-6122
- Goals and Milestones (Auszug):
  - Bis August 2012: Lösung für Ende-zu-Ende Verschlüsselung definieren
  - Bis September 2012: RFC 6122bis (Stringprep → PRECIS)

Gliederung

1. Ich
2. XMPP 101: Basics
   - Geschichte
   - Netzstruktur
   - JID
   - Stanzas
3. XMPP 102: Sicherheit
   - SSL/TLS
   - Dialback
   - SASL
   - End-to-End Encryption

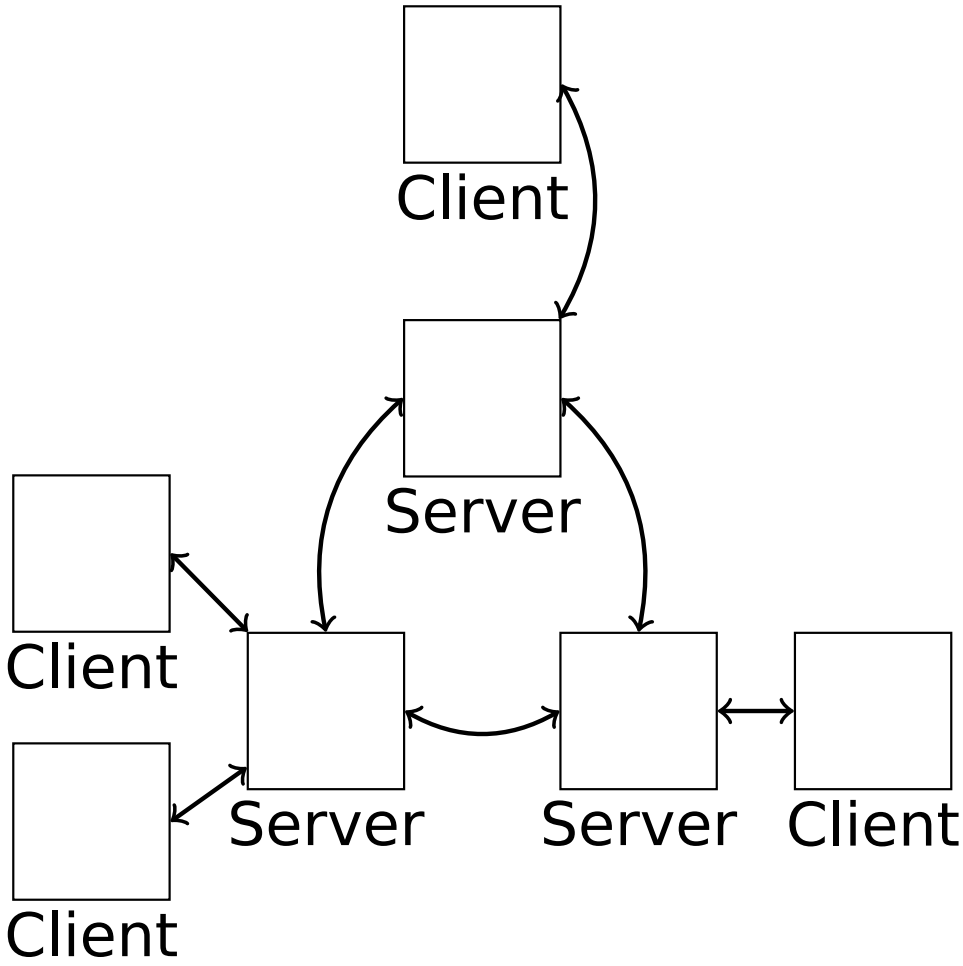
Netzstruktur

- Dezentralisiert
- c2s und s2s Verbindungen
- c2s bi-direktional
- s2s oft uni-directional (Dialback)
- s2s Verbindungen entstehen on-demand



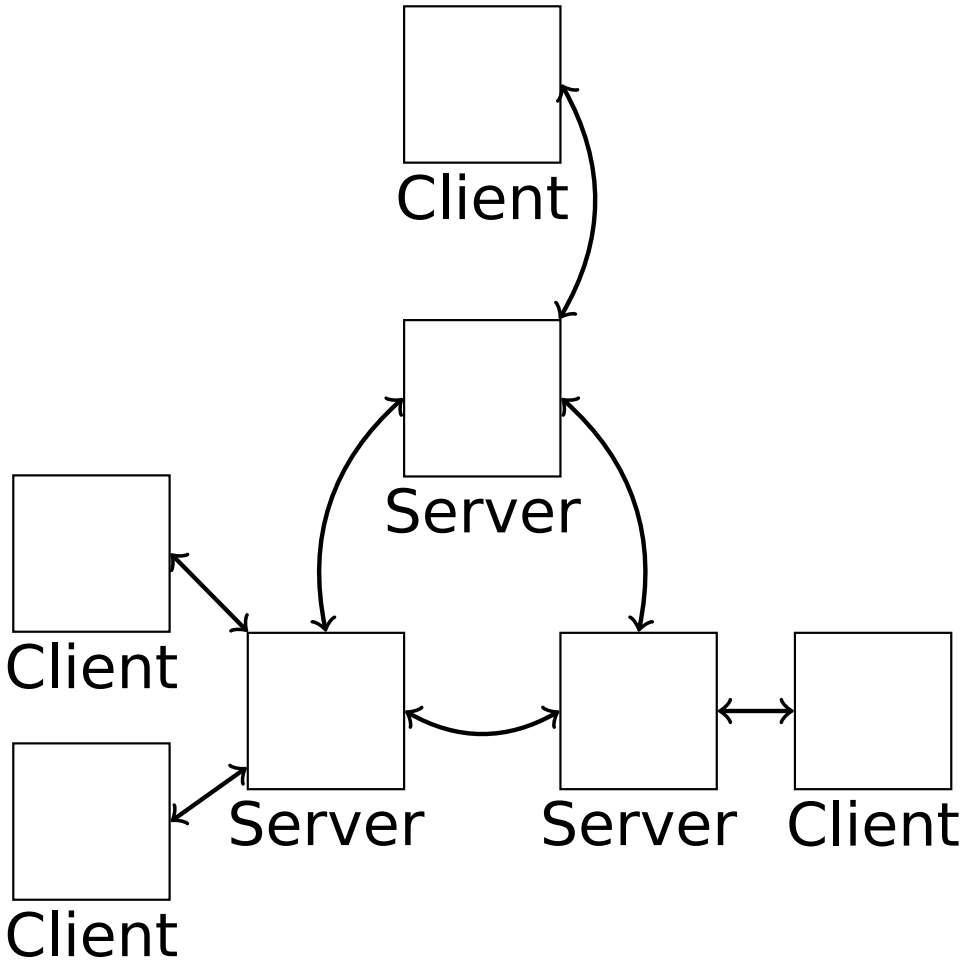
Netzstruktur

- Dezentralisiert
- c2s und s2s Verbindungen
- c2s bi-direktional
- s2s oft uni-directional (Dialback)
- s2s Verbindungen entstehen on-demand



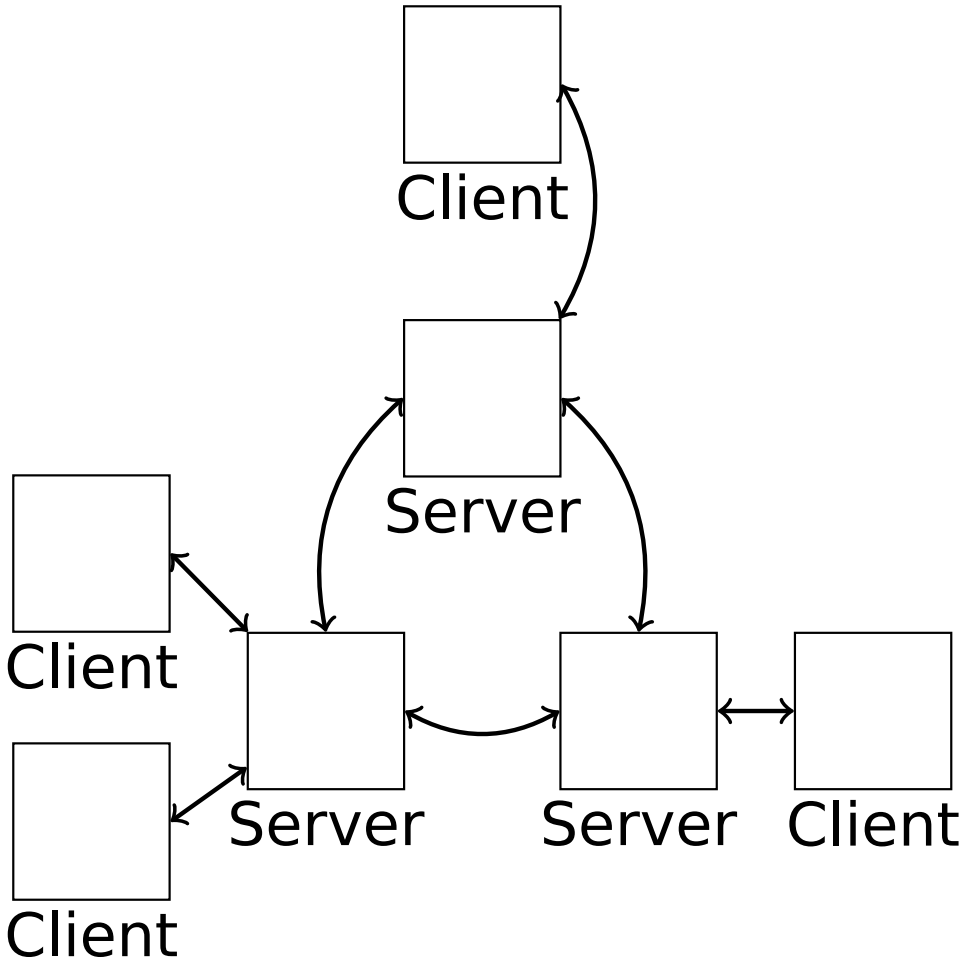
Netzstruktur

- Dezentralisiert
- c2s und s2s Verbindungen
- c2s bi-direktional
- s2s oft uni-directional (Dialback)
- s2s Verbindungen entstehen on-demand



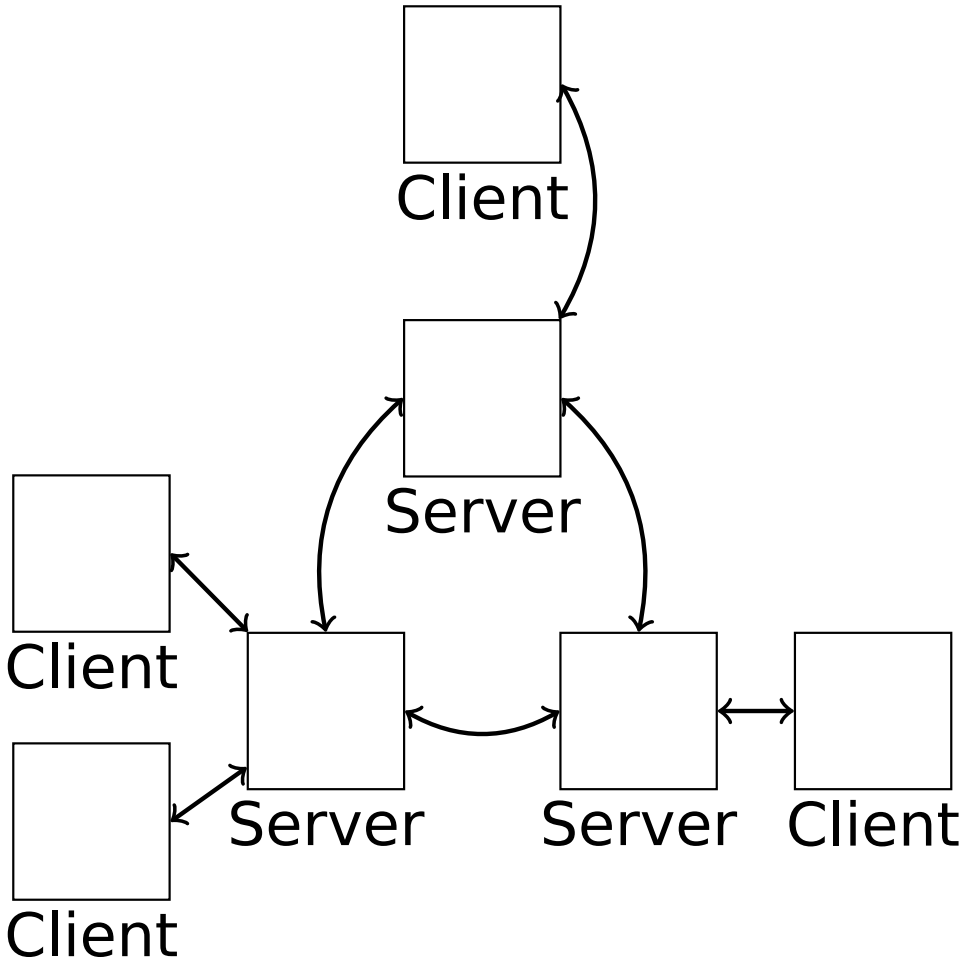
Netzstruktur

- Dezentralisiert
- c2s und s2s Verbindungen
- c2s bi-direktional
- s2s oft uni-directional (Dialback)
- s2s Verbindungen entstehen on-demand



Netzstruktur

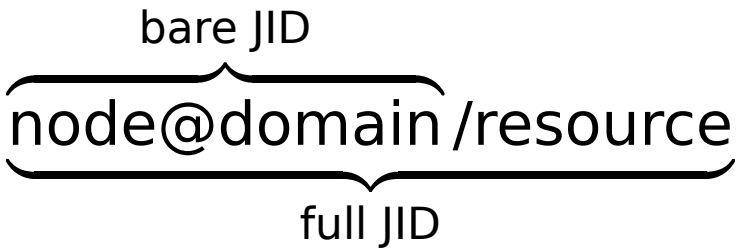
- Dezentralisiert
- c2s und s2s Verbindungen
- c2s bi-direktional
- s2s oft uni-directional (Dialback)
- s2s Verbindungen entstehen on-demand



Gliederung

1. Ich
2. XMPP 101: Basics
   - Geschichte
   - Netzstruktur
   - JID
   - Stanzas
3. XMPP 102: Sicherheit
   - SSL/TLS
   - Dialback
   - SASL
   - End-to-End Encryption

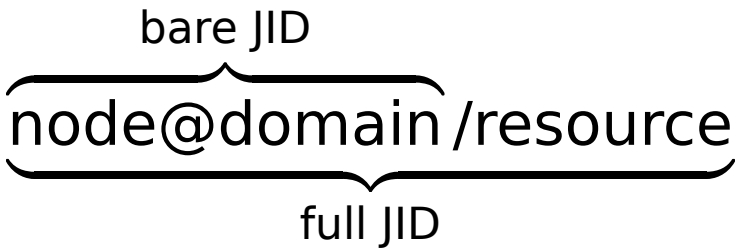
Adressen (heute)



Sieht fast aus wie eine E-mail Adresse, aber:

- UTF-8
- beschränkt auf Unicode 3.2 (Stringprep)
- domain nach IDNA2003
- node erlaubt fast alle Unicode Zeichen, außer Whitespace, Steuerzeichen und: ”, &, ’, /, :, <, >, @
- resource erlaubt nur non-ASCII Whitespace und Steuerzeichen nicht

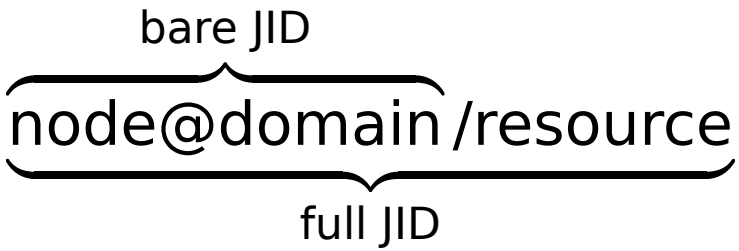
Adressen (heute)



Sieht fast aus wie eine E-mail Adresse, aber:

- UTF-8
- beschränkt auf Unicode 3.2 (Stringprep)
- domain nach IDNA2003
- node erlaubt fast alle Unicode Zeichen, außer Whitespace, Steuerzeichen und: ”, &, ’, /, :, <, >, @
- resource erlaubt nur non-ASCII Whitespace und Steuerzeichen nicht

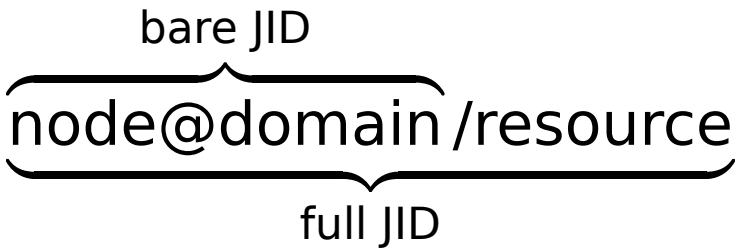
Adressen (heute)



Sieht fast aus wie eine E-mail Adresse, aber:

- UTF-8
- beschränkt auf Unicode 3.2 (Stringprep)
- domain nach IDNA2003
- node erlaubt fast alle Unicode Zeichen, außer Whitespace, Steuerzeichen und: ”, &, ’, /, :, <, >, @
- resource erlaubt nur non-ASCII Whitespace und Steuerzeichen nicht

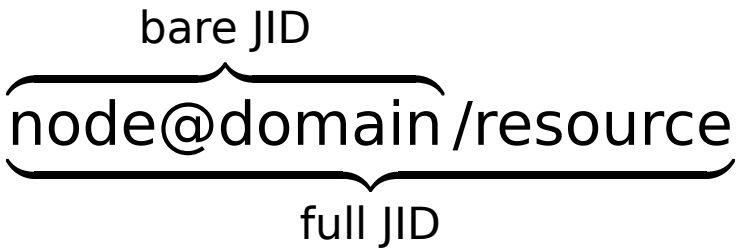
Adressen (heute)



Sieht fast aus wie eine E-mail Adresse, aber:

- UTF-8
- beschränkt auf Unicode 3.2 (Stringprep)
- domain nach IDNA2003
- node erlaubt fast alle Unicode Zeichen, außer Whitespace, Steuerzeichen und: ”, &, ’, /, :, <, >, @
- resource erlaubt nur non-ASCII Whitespace und Steuerzeichen nicht

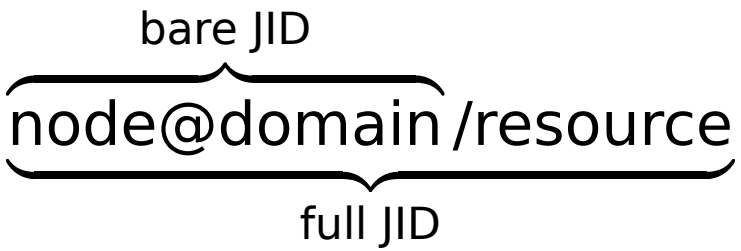
Adressen (heute)



Sieht fast aus wie eine E-mail Adresse, aber:

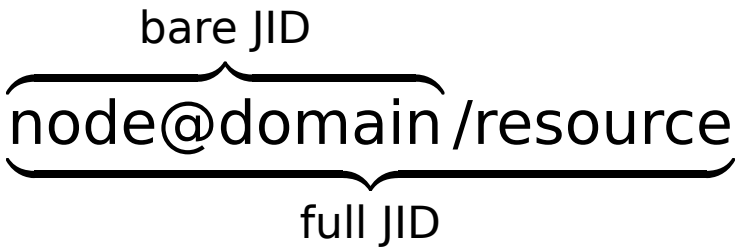
- UTF-8
- beschränkt auf Unicode 3.2 (Stringprep)
- domain nach IDNA2003
- node erlaubt fast alle Unicode Zeichen, außer Whitespace, Steuerzeichen und: ”, &, ’, /, :, <, >, @
- resource erlaubt nur non-ASCII Whitespace und Steuerzeichen nicht

Adressen (demnächst)



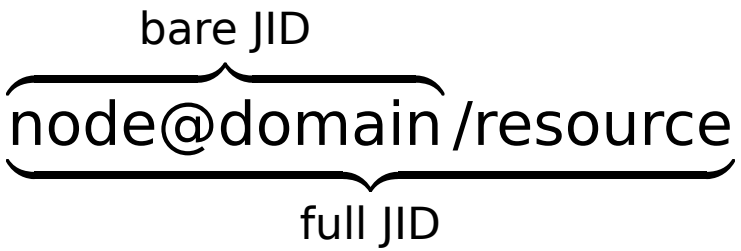
- flexible bezüglich der Unicode version (PRECIS)
- domain nach IDNA2008, nur U- und NR-LDH-label
- node basiert auf PRECIS NameClass, verbietet z. B. non-ASCII Symbole und Interpunktion
- resource basiert auf PRECIS FreeClass, ähnlich freizügig

Adressen (demnächst)



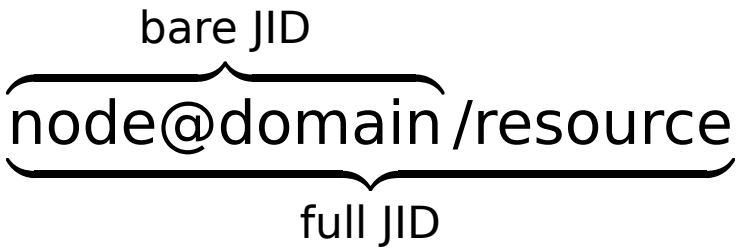
- flexible bezüglich der Unicode version (PRECIS)
- domain nach IDNA2008, nur U- und NR-LDH-label
- node basiert auf PRECIS NameClass, verbietet z. B. non-ASCII Symbole und Interpunktion
- resource basiert auf PRECIS FreeClass, ähnlich freizügig

Adressen (demnächst)



- flexible bezüglich der Unicode version (PRECIS)
- domain nach IDNA2008, nur U- und NR-LDH-label
- node basiert auf PRECIS NameClass, verbietet z. B. non-ASCII Symbole und Interpunktion
- resource basiert auf PRECIS FreeClass, ähnlich freizügig

Adressen (demnächst)



- flexible bezüglich der Unicode version (PRECIS)
- domain nach IDNA2008, nur U- und NR-LDH-label
- node basiert auf PRECIS NameClass, verbietet z. B. non-ASCII Symbole und Interpunktion
- resource basiert auf PRECIS FreeClass, ähnlich freizügig

Gliederung

1. Ich
2. XMPP 101: Basics
   - Geschichte
   - Netzstruktur
   - JID
   - Stanzas
3. XMPP 102: Sicherheit
   - SSL/TLS
   - Dialback
   - SASL
   - End-to-End Encryption

Überblick

- Kleine XML Stücke
- 3 Typen: Message, Presence, und IQ
- “Basic unit of meaning in XMPP”
- alle verfügen über die Attribute type, id, from und to

Überblick

- Kleine XML Stücke
- 3 Typen: Message, Presence, und IQ
- “Basic unit of meaning in XMPP”
- alle verfügen über die Attribute type, id, from und to

Überblick

- Kleine XML Stücke
- 3 Typen: Message, Presence, und IQ
- “Basic unit of meaning in XMPP”
- alle verfügen über die Attribute type, id, from und to

Überblick

- Kleine XML Stücke
- 3 Typen: Message, Presence, und IQ
- “Basic unit of meaning in XMPP”
- alle verfügen über die Attribute type, id, from und to

<message/>

```
<message from='juliet@capulet.lit/balcony'
    to='romeo@montague.lit' type='chat'>
  <body>Wherefore art thou romeo</body>
</message
```

- Fire and forget
- Push Mechanismus
- Verschiedene Typen: chat, groupchat, headline, normal
- Zustellungsregeln je nach Typ verschieden, z.B. für chat: Zustellung an die “anwesendste”, oder alle Resourcen

<message/>

```
<message from='juliet@capulet.lit/balcony'
    to='romeo@montague.lit' type='chat'>
  <body>Wherefore art thou romeo</body>
</message
```

- Fire and forget
- Push Mechanismus
- Verschiedene Typen: chat, groupchat, headline, normal
- Zustellungsregeln je nach Typ verschieden, z.B. für chat: Zustellung an die “anwesendste”, oder alle Resourcen

<message/>

```
<message from='juliet@capulet.lit/balcony'
    to='romeo@montague.lit' type='chat'>
  <body>Wherefore art thou romeo</body>
</message
```

- Fire and forget
- Push Mechanismus
- Verschiedene Typen: chat, groupchat, headline, normal
- Zustellungsregeln je nach Typ verschieden, z.B. für chat: Zustellung an die “anwesendste”, oder alle Resourcen

<message/>

```
<message from='juliet@capulet.lit/balcony'
    to='romeo@montague.lit' type='chat'>
  <body>Wherefore art thou romeo</body>
</message
```

- Fire and forget
- Push Mechanismus
- Verschiedene Typen: chat, groupchat, headline, normal
- Zustellungsregeln je nach Typ verschieden, z.B. für chat: Zustellung an die “anwesendste”, oder alle Resourcen

<presence/>

```
<presence from='romeo@example.net/orchard'>
  <show>dnd</show>
  <status>Wooing Juliet</status>
</presence>
```

- Übermittelt “Anwesenheit”/“Erreichbarkeit”
- Dient auch zur Verwaltung von Presence Subscriptions
- type Attribut hat keine Beziehung zum Status. Nur entweder da (kein Attribut), oder nicht (type='unavailable')

<presence/>

```
<presence from='romeo@example.net/orchard'>
  <show>dnd</show>
  <status>Wooing Juliet</status>
</presence>
```

- Übermittelt “Anwesenheit”/“Erreichbarkeit”
- Dient auch zur Verwaltung von Presence Subscriptions
- type Attribut hat keine Beziehung zum Status. Nur entweder da (kein Attribut), oder nicht (type='unavailable')

<presence/>

```
<presence from='romeo@example.net/orchard'>
  <show>dnd</show>
  <status>Wooing Juliet</status>
</presence>
```

- Übermittelt “Anwesenheit”/“Erreichbarkeit”
- Dient auch zur Verwaltung von Presence Subscriptions
- type Attribut hat keine Beziehung zum Status. Nur entweder da (kein Attribut), oder nicht (type='unavailable')

<iq/>

```
R: <iq to='juliet@capulet.lit/balcony'
     type='get' id='1'>
       <ping xmlns='urn:xmpp:ping'/>
   </iq>

J: <iq id='1' type='result'
     to='romeo@example.net/orchard'
     from='juliet@capulet.lit/balcony'/>
```

- Query/Response Mechanismus
- muss als einziges eine id haben
- type='get' oder 'set' wird mit type='result' oder 'error' beantwortet

<iq/>

```
R: <iq to='juliet@capulet.lit/balcony'
     type='get' id='1'>
       <ping xmlns='urn:xmpp:ping'/>
   </iq>

J: <iq id='1' type='result'
     to='romeo@example.net/orchard'
     from='juliet@capulet.lit/balcony'/>
```

- Query/Response Mechanismus
- muss als einziges eine id haben
- type='get' oder 'set' wird mit type='result' oder 'error' beantwortet

<iq/>

```
R: <iq to='juliet@capulet.lit/balcony'
    type='get' id='1'>
      <ping xmlns='urn:xmpp:ping'/>
    </iq>

J: <iq id='1' type='result'
    to='romeo@example.net/orchard'
    from='juliet@capulet.lit/balcony'/>
```

- Query/Response Mechanismus
- muss als einziges eine id haben
- type='get' oder 'set' wird mit type='result' oder 'error' beantwortet

Gliederung

1. Ich
2. XMPP 101: Basics
   - Geschichte
   - Netzstruktur
   - JID
   - Stanzas
3. XMPP 102: Sicherheit
   - SSL/TLS
   - Dialback
   - SASL
   - End-to-End Encryption

Gliederung

1. Ich
2. XMPP 101: Basics
   - Geschichte
   - Netzstruktur
   - JID
   - Stanzas
3. XMPP 102: Sicherheit
   - SSL/TLS
   - Dialback
   - SASL
   - End-to-End Encryption

SSL/TLS

- Legacy SSL:
  - Port 5223
  - nur SSLv2
  - deprecated
- TLS:
  - STARTTLS
  - ersetzt SNI
  - kann (MAY) vom Server erzwungen werden
  - Clients müssen (MUST) warnen wenn unverschlüsselt
  - Auf c2s Verbindugen üblich s2s leider seltener
  - GoogleTalk hat keine Unterstützung für s2s

SSL/TLS

- Legacy SSL:
  - Port 5223
  - nur SSLv2
  - deprecated
- TLS:
  - STARTTLS
  - ersetzt SNI
  - kann (MAY) vom Server erzwungen werden
  - Clients müssen (MUST) warnen wenn unverschlüsselt
  - Auf c2s Verbindugen üblich s2s leider seltener
  - GoogleTalk hat keine Unterstützung für s2s

SSL/TLS

- Legacy SSL:
  - Port 5223
  - nur SSLv2
  - deprecated
- TLS:
  - STARTTLS
  - ersetzt SNI
  - kann (MAY) vom Server erzwungen werden
  - Clients müssen (MUST) warnen wenn unverschlüsselt
  - Auf c2s Verbindugen üblich s2s leider seltener
  - GoogleTalk hat keine Unterstützung für s2s

SSL/TLS

- Legacy SSL:
  - Port 5223
  - nur SSLv2
  - deprecated
- TLS:
  - STARTTLS
  - ersetzt SNI
  - kann (MAY) vom Server erzwungen werden
  - Clients müssen (MUST) warnen wenn unverschlüsselt
  - Auf c2s Verbindugen üblich s2s leider seltener
  - GoogleTalk hat keine Unterstützung für s2s

SSL/TLS

- Legacy SSL:
  - Port 5223
  - nur SSLv2
  - deprecated
- TLS:
  - STARTTLS
  - ersetzt SNI
  - kann (MAY) vom Server erzwungen werden
  - Clients müssen (MUST) warnen wenn unverschlüsselt
  - Auf c2s Verbindugen üblich s2s leider seltener
  - GoogleTalk hat keine Unterstützung für s2s

SSL/TLS

- Legacy SSL:
  - Port 5223
  - nur SSLv2
  - deprecated
- TLS:
  - STARTTLS
  - ersetzt SNI
  - kann (MAY) vom Server erzwungen werden
  - Clients müssen (MUST) warnen wenn unverschlüsselt
  - Auf c2s Verbindugen üblich s2s leider seltener
  - GoogleTalk hat keine Unterstützung für s2s

SSL/TLS

- Legacy SSL:
  - Port 5223
  - nur SSLv2
  - deprecated
- TLS:
  - STARTTLS
  - ersetzt SNI
  - kann (MAY) vom Server erzwungen werden
  - Clients müssen (MUST) warnen wenn unverschlüsselt
  - Auf c2s Verbindugen üblich s2s leider seltener
  - GoogleTalk hat keine Unterstützung für s2s

SSL/TLS

- Legacy SSL:
  - Port 5223
  - nur SSLv2
  - deprecated
- TLS:
  - STARTTLS
  - ersetzt SNI
  - kann (MAY) vom Server erzwungen werden
  - Clients müssen (MUST) warnen wenn unverschlüsselt
  - Auf c2s Verbindugen üblich s2s leider seltener
  - GoogleTalk hat keine Unterstützung für s2s

SSL/TLS

- Legacy SSL:
  - Port 5223
  - nur SSLv2
  - deprecated
- TLS:
  - STARTTLS
  - ersetzt SNI
  - kann (MAY) vom Server erzwungen werden
  - Clients müssen (MUST) warnen wenn unverschlüsselt
  - Auf c2s Verbindugen üblich s2s leider seltener
  - GoogleTalk hat keine Unterstützung für s2s

SSL/TLS

- Legacy SSL:
  - Port 5223
  - nur SSLv2
  - deprecated
- TLS:
  - STARTTLS
  - ersetzt SNI
  - kann (MAY) vom Server erzwungen werden
  - Clients müssen (MUST) warnen wenn unverschlüsselt
  - Auf c2s Verbindugen üblich s2s leider seltener
  - GoogleTalk hat keine Unterstützung für s2s

SSL/TLS

- Legacy SSL:
  - Port 5223
  - nur SSLv2
  - deprecated
- TLS:
  - STARTTLS
  - ersetzt SNI
  - kann (MAY) vom Server erzwungen werden
  - Clients müssen (MUST) warnen wenn unverschlüsselt
  - Auf c2s Verbindugen üblich s2s leider seltener
  - GoogleTalk hat keine Unterstützung für s2s

X.509 Zertifikate nach RFC 6125

“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)”

- normative Referenz von RFC 6120
- commonName enthält einen menschen-freundlichen Namen
- Domain ist in einem SAN des Typs dNSName enthalten
- Domain und Service sind in SANs des Typs otherName, mit der Namensform SRVName gespeichert

X.509 Zertifikate nach RFC 6125

“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)”

- normative Referenz von RFC 6120
- commonName enthält einen menschen-freundlichen Namen
- Domain ist in einem SAN des Typs dNSName enthalten
- Domain und Service sind in SANs des Typs otherName, mit der Namensform SRVName gespeichert

X.509 Zertifikate nach RFC 6125

“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)”

- normative Referenz von RFC 6120
- commonName enthält einen menschen-freundlichen Namen
- Domain ist in einem SAN des Typs dNSName enthalten
- Domain und Service sind in SANs des Typs otherName, mit der Namensform SRVName gespeichert

X.509 Zertifikate nach RFC 6125

“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)”

- normative Referenz von RFC 6120
- commonName enthält einen menschen-freundlichen Namen
- Domain ist in einem SAN des Typs dNSName enthalten
- Domain und Service sind in SANs des Typs otherName, mit der Namensform SRVName gespeichert

X.509 Zertifikate nach RFC 6125

CN:	Chaos Computer Club Köln
dNSName:	koeln.ccc.de
SRVName:	_xmpp-client.koeln.ccc.de
SRVName:	_xmpp-server.koeln.ccc.de

Gliederung

1. Ich
2. XMPP 101: Basics
   - Geschichte
   - Netzstruktur
   - JID
   - Stanzas
3. XMPP 102: Sicherheit
   - SSL/TLS
   - Dialback
   - SASL
   - End-to-End Encryption

Dialback

- Schwache Authentifizierung
- DNS basiert
- mit DNSSec sicher(er)
- recht weit verbreitet

Dialback

- Schwache Authentifizierung
- DNS basiert
- mit DNSSec sicher(er)
- recht weit verbreitet

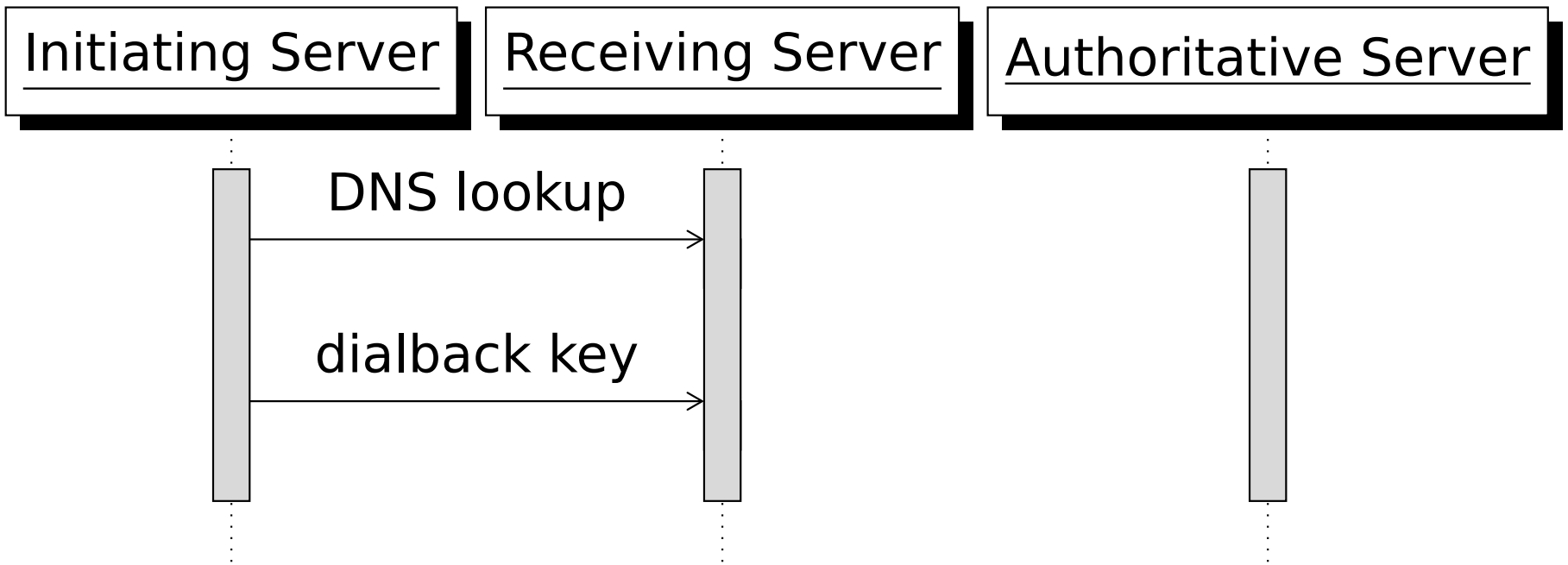
Dialback

- Schwache Authentifizierung
- DNS basiert
- mit DNSSec sicher(er)
- recht weit verbreitet

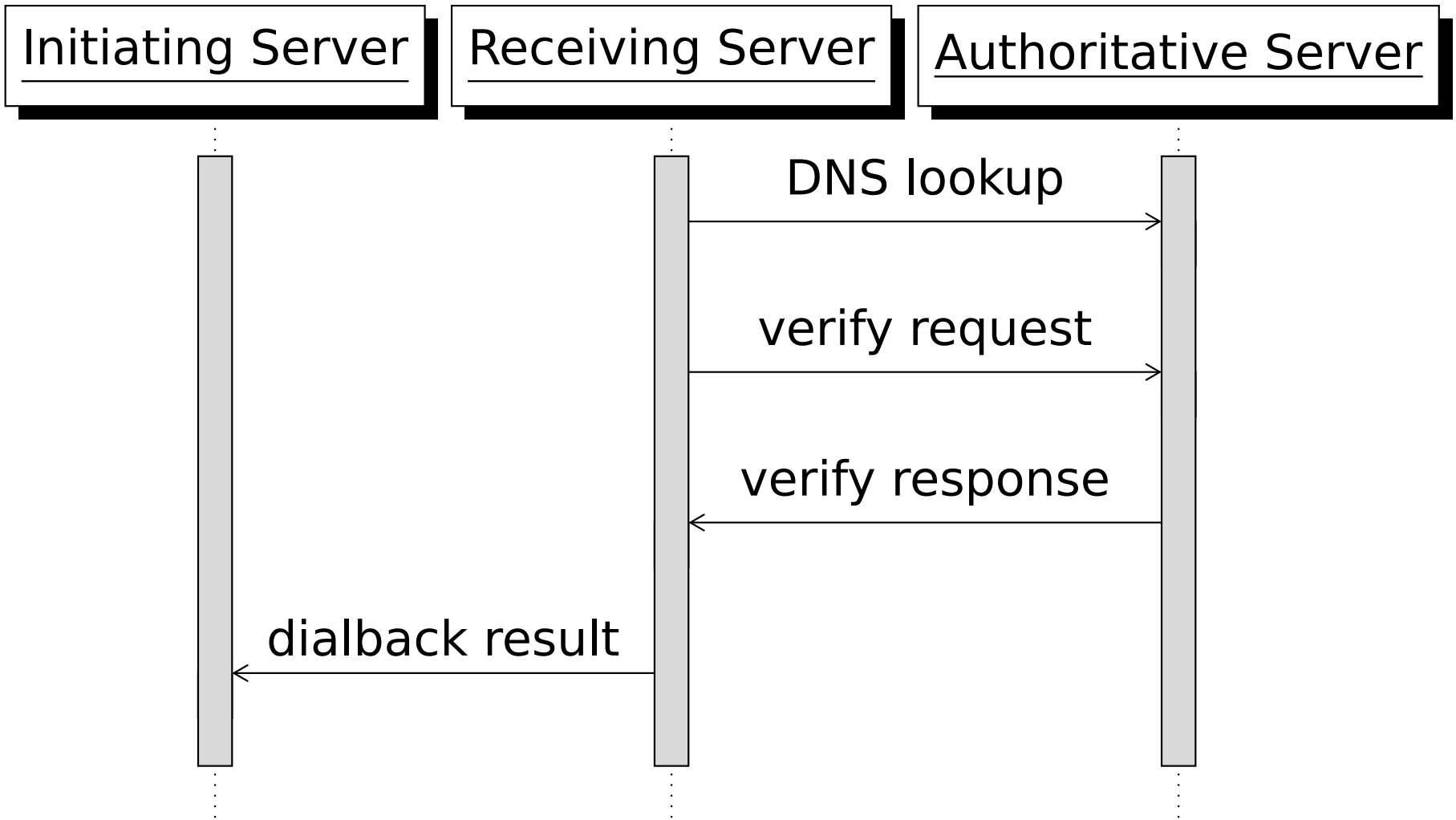
Dialback

- Schwache Authentifizierung
- DNS basiert
- mit DNSSec sicher(er)
- recht weit verbreitet

Dialback


Initiating Server
Receiving Server
Authoritative Server
DNS lookup
dialback key

Dialback


Initiating Server
Receiving Server
Authoritative Server
DNS lookup
verify request
verify response
dialback result

Gliederung

1. Ich
2. XMPP 101: Basics
   - Geschichte
   - Netzstruktur
   - JID
   - Stanzas
3. XMPP 102: Sicherheit
   - SSL/TLS
   - Dialback
   - SASL
   - End-to-End Encryption

SASL

- Simple Authentication and Security Layer
- auch von SMTP, IMAP, POP3, IRCv3,...verwendet
- bietet ein Framework für Authentifizierungsmechanismen
- MTI Mechanismen in XMPP
  - PLAIN
  - SCRAM-SHA-1(-PLUS)
  - EXTERNAL
  - DIGEST-MD5 (nicht mehr in RFC 6120)

SASL

- Simple Authentication and Security Layer
- auch von SMTP, IMAP, POP3, IRCv3,...verwendet
- bietet ein Framework für Authentifizierungsmechanismen
- MTI Mechanismen in XMPP
  - PLAIN
  - SCRAM-SHA-1(-PLUS)
  - EXTERNAL
  - DIGEST-MD5 (nicht mehr in RFC 6120)

SASL

- Simple Authentication and Security Layer
- auch von SMTP, IMAP, POP3, IRCv3,...verwendet
- bietet ein Framework für Authentifizierungsmechanismen
- MTI Mechanismen in XMPP
  - PLAIN
  - SCRAM-SHA-1(-PLUS)
  - EXTERNAL
  - DIGEST-MD5 (nicht mehr in RFC 6120)

SASL

- Simple Authentication and Security Layer
- auch von SMTP, IMAP, POP3, IRCv3,...verwendet
- bietet ein Framework für Authentifizierungsmechanismen
- MTI Mechanismen in XMPP
  - PLAIN
  - SCRAM-SHA-1(-PLUS)
  - EXTERNAL
  - DIGEST-MD5 (nicht mehr in RFC 6120)

SASL

- Simple Authentication and Security Layer
- auch von SMTP, IMAP, POP3, IRCv3,...verwendet
- bietet ein Framework für Authentifizierungsmechanismen
- MTI Mechanismen in XMPP
  - PLAIN
  - SCRAM-SHA-1(-PLUS)
  - EXTERNAL
  - DIGEST-MD5 (nicht mehr in RFC 6120)

SASL

- Simple Authentication and Security Layer
- auch von SMTP, IMAP, POP3, IRCv3,...verwendet
- bietet ein Framework für Authentifizierungsmechanismen
- MTI Mechanismen in XMPP
  - PLAIN
  - SCRAM-SHA-1(-PLUS)
  - EXTERNAL
  - DIGEST-MD5 (nicht mehr in RFC 6120)

SASL

- Simple Authentication and Security Layer
- auch von SMTP, IMAP, POP3, IRCv3,...verwendet
- bietet ein Framework für Authentifizierungsmechanismen
- MTI Mechanismen in XMPP
  - PLAIN
  - SCRAM-SHA-1(-PLUS)
  - EXTERNAL
  - DIGEST-MD5 (nicht mehr in RFC 6120)

SASL

- Simple Authentication and Security Layer
- auch von SMTP, IMAP, POP3, IRCv3,...verwendet
- bietet ein Framework für Authentifizierungsmechanismen
- MTI Mechanismen in XMPP
  - PLAIN
  - SCRAM-SHA-1(-PLUS)
  - EXTERNAL
  - DIGEST-MD5 (nicht mehr in RFC 6120)

Gliederung

1. Ich
2. XMPP 101: Basics
   - Geschichte
   - Netzstruktur
   - JID
   - Stanzas
3. XMPP 102: Sicherheit
   - SSL/TLS
   - Dialback
   - SASL
   - End-to-End Encryption

Status quo

- Noch kein “vernünftiger” Standard
- OTR funktioniert (einigermaßen)
- historical XEP für GPG
- S/MIME RFC
- ESessions
- XTLS
- Work item der WG

Status quo

- Noch kein “vernünftiger” Standard
- OTR funktioniert (einigermaßen)
- historical XEP für GPG
- S/MIME RFC
- ESessions
- XTLS
- Work item der WG

Status quo

- Noch kein “vernünftiger” Standard
- OTR funktioniert (einigermaßen)
- historical XEP für GPG
- S/MIME RFC
- ESessions
- XTLS
- Work item der WG

Status quo

- Noch kein “vernünftiger” Standard
- OTR funktioniert (einigermaßen)
- historical XEP für GPG
- S/MIME RFC
- ESessions
- XTLS
- Work item der WG

Status quo

- Noch kein “vernünftiger” Standard
- OTR funktioniert (einigermaßen)
- historical XEP für GPG
- S/MIME RFC
- ESessions
- XTLS
- Work item der WG

Status quo

- Noch kein “vernünftiger” Standard
- OTR funktioniert (einigermaßen)
- historical XEP für GPG
- S/MIME RFC
- ESessions
- XTLS
- Work item der WG

Status quo

- Noch kein “vernünftiger” Standard
- OTR funktioniert (einigermaßen)
- historical XEP für GPG
- S/MIME RFC
- ESessions
- XTLS
- Work item der WG

draft-miller-xmpp-e2e

- Verwendet die Arbeit der JOSE WG
- “whole stanza encryption”
- Unterstützt mehrere Empfänger
- Signaturen fehlen noch

draft-miller-xmpp-e2e

- Verwendet die Arbeit der JOSE WG
- “whole stanza encryption”
- Unterstützt mehrere Empfänger
- Signaturen fehlen noch

draft-miller-xmpp-e2e

- Verwendet die Arbeit der JOSE WG
- “whole stanza encryption”
- Unterstützt mehrere Empfänger
- Signaturen fehlen noch

draft-miller-xmpp-e2e

- Verwendet die Arbeit der JOSE WG
- “whole stanza encryption”
- Unterstützt mehrere Empfänger
- Signaturen fehlen noch

draft-miller-xmpp-e2e: Vorgehen

1. Generiere einen Session Master Key (SMK)
2. Generiere einen Content Master Key (CMK) pro Stanza
3. Verschlüssele den CMK mit dem SMK
4. Verschlüssele die Stanza mit dem CMK
5. Sende (CMK, ciphertext)
6. public/private key Kryptographie zum Austausch von SMKs

draft-miller-xmpp-e2e: Vorgehen

1. Generiere einen Session Master Key (SMK)
2. Generiere einen Content Master Key (CMK) pro Stanza
3. Verschlüssele den CMK mit dem SMK
4. Verschlüssele die Stanza mit dem CMK
5. Sende (CMK, ciphertext)
6. public/private key Kryptographie zum Austausch von SMKs

draft-miller-xmpp-e2e: Vorgehen

1. Generiere einen Session Master Key (SMK)
2. Generiere einen Content Master Key (CMK) pro Stanza
3. Verschlüssele den CMK mit dem SMK
4. Verschlüssele die Stanza mit dem CMK
5. Sende (CMK, ciphertext)
6. public/private key Kryptographie zum Austausch von SMKs

draft-miller-xmpp-e2e: Vorgehen

1. Generiere einen Session Master Key (SMK)
2. Generiere einen Content Master Key (CMK) pro Stanza
3. Verschlüssele den CMK mit dem SMK
4. Verschlüssele die Stanza mit dem CMK
5. Sende (CMK, ciphertext)
6. public/private key Kryptographie zum Austausch von SMKs

draft-miller-xmpp-e2e: Vorgehen

1. Generiere einen Session Master Key (SMK)
2. Generiere einen Content Master Key (CMK) pro Stanza
3. Verschlüssele den CMK mit dem SMK
4. Verschlüssele die Stanza mit dem CMK
5. Sende (CMK, ciphertext)
6. public/private key Kryptographie zum Austausch von SMKs

draft-miller-xmpp-e2e: Vorgehen

1. Generiere einen Session Master Key (SMK)
2. Generiere einen Content Master Key (CMK) pro Stanza
3. Verschlüssele den CMK mit dem SMK
4. Verschlüssele die Stanza mit dem CMK
5. Sende (CMK, ciphertext)
6. public/private key Kryptographie zum Austausch von SMKs

Fragen?