Sun Java™ System Message Queue Versionshinweise

Version 3.5 SP1

Teilenummer 817-7191-10

In diesen Versionshinweisen sind wichtige Informationen enthalten, die zum Zeitpunkt der Herausgabe der Version 3.5 SP1 von Sun Java™ System Message Queue (vormals Sun™ ONE Message Queue) zur Verfügung standen. Das Dokument umfasst außerdem den Inhalt der Versionshinweise für Message Queue 3.5 für Kunden, die von Versionen vor 3.5 aufrüsten. Hier werden neue Funktionen, Verbesserungen, bekannte Einschränkungen und Probleme, technische Hinweise und andere Informationen zu Message Queue 3.5 behandelt.

Die neueste Ausgabe dieser Versionshinweise finden Sie auf der Dokumentations-Website für Sun Java System unter `http://docs.sun.com/coll/MessageQueue_35_SP1`. Besuchen Sie diese Website vor der Installation und Konfiguration Ihrer Software und später regelmäßig, um stets die neuesten Versionshinweise und Handbücher verfügbar zu haben.

Diese Versionshinweise sind in die folgenden Abschnitte aufgegliedert:

Revisionsverlauf

Tabelle 1 Revisionsverlauf

Datum	Beschreibung der Änderungen
12. März 2004	Fehlerinformationen aktualisiert. Abschnitt „Bekannte Probleme und Einschränkungen“ aktualisiert.
	Abschnitt „Dateien für Neuverteilung“ hinzugefügt. Abschnitt „Aktualisierung der Dokumentation“ aktualisiert. Abschnitt „Kompatibilität“ aktualisiert. Abschnitt „Informationen zu Sun Java System“ aktualisiert.
9. Januar 2004	Informationen zur Versionsunterstützung für PointBase 4.8 aktualisiert, Informationen für C-API-Funktionen aktualisiert.

Info zu Message Queue 3.5 SP1

Message Queue 3.5 SP1 ist eine Aktualisierung von Message Queue 3.5 und enthält alle neuen Funktionen von Message Queue 3.5. Außerdem beinhaltet Message Queue 3.5 SP1 Fehlerbehebungen und einen neuen Markennamen. Das Produkt gehört nun zur Produktfamilie Sun Java™ System.

Message Queue 3.5 SP1 wurde für die Einhaltung der Java™ Message Service (JMS) 1.1-Spezifikation zertifiziert und hat die JMS 1.1 Compatibility Test Suite (CTS) bestanden.

In diesem Abschnitt werden die Änderungen in Message Queue 3.5 SP1 und in der Vorgängerversion, Message Queue 3.5, beschrieben.

Message Queue 3.5 SP1

In Message Queue 3.5 SP1 wurden Fehler behoben und für das Produkt und die zugehörige Dokumentation wurde ein Rebranding durchgeführt.

Message Queue 3.5

Bei Message Queue 3.5 wurden zahlreiche neue Funktionen eingeführt:

Diese werden in den folgenden Unterabschnitten beschrieben.

C-Client-Unterstützung (Enterprise Edition)

Message Queue 3.5 beinhaltet Laufzeitunterstützung für C-API und C (im Folgenden als C-Client-Funktion bezeichnet). Mit der C-Client-Funktion können Legacy-Systeme in ein Message Queue-Messaging-System integriert werden. Sie setzt die JMS-Spezifikation nahezu vollständig um. Sie unterstützt alle JMS-Funktionen mit Ausnahme bestimmter Body-Typen („map“, „stream“ und „object“), QueueBrowser-Funktionen und J2EE-Anwendungsserverfunktionen (z. B. verteilte Transaktionen und `ConnectionConsumer`-Objekte).

Die Unterstützung für die C-Client-Funktion erfolgt über einen Satz getrennt installierter Bibliotheken, die nur über eine Enterprise Edition-Lizenz aktiviert werden können. Daher müssen zum Aufrüsten von der Platform Edition auf die Enterprise Edition sowohl die Enterprise Edition-Lizenzdatei als auch die C-Bibliotheken installiert werden.

Platform Edition-Kunden, die die 90-Tage-Testlizenz für die Enterprise Edition aktivieren, können die C-Client-Funktion verwenden, wenn sie mit Sun über den Alias `imq-feedback@sun.com` Kontakt aufnehmen und das C-API SDK anfordern. Die Technikabteilung ist für die Beantwortung dieser Anfragen zuständig und stellt das C-API SDK auf der anonymen FTP-Site zur Verfügung. Nach Ablauf der 90-Tage-Lizenz für die Enterprise Edition können die Kunden auch weiterhin C-Clients aufbauen, sie jedoch nicht mit ihrem Platform Edition-Broker verbinden.

Für die C-Client-Funktion sind spezielle Compiler-Versionen auf den verschiedenen Betriebssystemplattformen erforderlich. Es werden neue Systemvoraussetzungen für die Enterprise Edition hinzugefügt (Einzelheiten dazu finden Sie im *Message Queue Installation Guide*). Die C-Client-Funktion ist außerdem von der NSPR-Bibliothek (Netscape Portable Runtime) und der NSS-Bibliothek (Network Security Service) abhängig. (In Message Queue 3.5 wurde die C-Client-Funktion erfolgreich unter Linux Red Hat Advanced Server 2.1. getestet. Die NSPR- und NSS-Bibliotheksversionen, anhand deren der Test erfolgte, wurden nicht für diese Linux-Edition zertifiziert.)

Derzeit unterstützt die C-API nicht den Authentifizierungstyp `basic`. Wenn Sie den Broker für die Verwendung dieses Authentifizierungstyps konfigurieren, schlägt ein Aufruf an die Funktion `MQCreateConnection` mit folgendem Ergebnis fehl: `MQ_UNSUPPORTED_AUTH_TYPE`.

Zur Dokumentation der C-Client-Funktion gehören Referenzdokumentation, Programmierungsdokumentation und C-API-Beispiel-Clients. Weitere Informationen finden Sie im Handbuch *Message Queue C Client Developer's Guide*.

Failover der Java-Client-Verbindung (Enterprise Edition)

Message Queue 3.5 unterstützt eine erweiterte Funktion zur automatischen Verbindungswiederherstellung, mit der eine unterbrochene Verbindung nicht nur auf dem ursprünglichen, sondern auch auf einem anderen Broker wiederhergestellt werden kann (Failover der Client-Verbindung). Die erneute Verbindung erfolgt zum Meldungsdienst und nicht zu einer speziellen Broker-Instanz. Um diese Funktionen zu implementieren, konfigurieren Sie das von der Connection Factory verwaltete Objekt (Message Queue 3.5 weist ein neues Adress-Spezifikationsschema für den Meldungsdienst auf), indem Sie eine Reihe von Broker-Adressen angeben (`imqAddressList`). Wenn die Client-Laufzeit eine Verbindung zu einem Meldungsdienst (wieder)herstellen muss, versucht sie, nacheinander in einer bestimmten Reihenfolge eine Verbindung zu den einzelnen Brokern in der Liste herzustellen, bis ein verfügbarer Broker gefunden oder die gesamte Liste erfolgreich abgearbeitet wurde. Sie können die Anzahl der Verbindungsversuche (`imqAddressListIterations`) für die einzelnen Broker und den Zeitabstand zwischen den Verbindungsversuchen (`imqAddressListInterval`) angeben.

Wenn die automatische Wiederherstellung der Verbindung mit einer anderen Brokerinstanz erfolgt als der ursprünglichen, können persistente Meldungen und andere Zustandsinformationen, die auf dem ausgefallenen (bzw. von der Verbindung getrennten) Broker gespeichert sind, verloren gehen. Dies liegt daran, dass die verschiedenen Broker-Instanzen in einem Cluster keinen gemeinsamen, hochverfügbaren Persistenzspeicher verwenden. Durch die Fähigkeit der Client-Laufzeit, automatisch eine Verbindung zu einer anderen Browser-Instanz herzustellen, erhalten Sie jedoch die Möglichkeit, Wiederherstellungsszenarios zu entwickeln, durch die ein Backup-Broker oder ein Broker-Cluster für (unvollständigen) Failover-Schutz verwendet werden kann.

Bei aktivierter automatischer Verbindungswiederherstellung macht Message Queue 3.5 nun außerdem *temporäre* Ziele persistent, wenn die zugehörige Verbindung ausfällt. Der Grund dafür ist die Möglichkeit, dass die Clients möglicherweise erneut eine Verbindung zu ihnen herstellen und auf sie zugreifen. Temporäre Ziele werden wie andere physische Ziele behandelt. Daher müssen Sie möglicherweise in regelmäßigen Abständen alle nicht verwendeten temporären Ziele vom Broker löschen.

Weitere Informationen finden Sie im Handbuch *Message Queue Java Client Developer's Guide*.

Message Queue unterstützte bisher eine Funktion zur automatischen Verbindungswiederherstellung, bei der die Client-Laufzeit bei Verbindungsausfall automatisch eine erneute Verbindung zu einem Broker herstellen konnte, außer in Fällen, in denen der clientseitige Zustand bei der Wiederherstellung der Verbindung nicht vollständig auf dem Broker wiederhergestellt werden konnte (z. B. bei der Verwendung von transaktionalen Sitzungen oder temporären Zielen, die nur für die Dauer einer Verbindung existieren).

Verbesserte Steuerung des Meldungsflusses beim Broker

Der Broker wurde dahingehend verbessert, dass der Fluss der Meldungen in die Ziele besser gesteuert und der Fall vermieden werden kann, dass Meldungen schneller erstellt werden als sie verbraucht werden können. (Außerdem können andere neue Funktionen von Message Queue 3.5 zur Vermeidung von Engpässen im Fluss der Meldungen *aus* den Zielen beitragen. Siehe „Verbesserte Steuerung des Meldungsflusses beim Java-Client“ auf Seite 7 und „Verbesserte Verfahren für die Warteschlangenzustellung (Enterprise Edition)“ auf Seite 10.)

Zu den Meldungsflussverbesserungen des Brokers gehören folgende Funktionen:

- Einführung einer Obergrenze für die Zahl der einem Ziel zugeordneten Produzenten. Die Ziele weisen nun ein neues `maxNumProducers`-Attribut auf; wenn der Grenzwert für dieses Attribut erreicht ist, können keine neuen Produzenten für diese Quelle erstellt werden.
- Aktivierung neuer, konfigurierbarer Obergrenzen und spezieller Reaktionen, wenn die Ziele die Grenzen für `maxTotalMsgBytes` und `maxNumMsgs` erreichen. Insbesondere wurden folgende Änderungen implementiert:
  - Ausweitung der Zielattribute `maxTotalMsgBytes` und `maxNumMsgs` auf Themenziele (bisher galten sie nur für Warteschlangenziele)
  - Aktivierung der Einstellung von `maxTotalMsgBytes` und `maxNumMsgs` für *automatisch erstellte* Ziele
  - (Enterprise Edition) Möglichkeit, dass ein Administrator eine Reaktion auf das Erreichen eines der oben angegebenen Grenzwerte auswählt. Folgende Reaktionen können festgelegt werden: Verlangsamen der Produzenten (`FLOW_CONTROL`), Ausschließen der ältesten Meldungen (`REMOVE_OLDEST`), Ausschließen der Meldungen mit der geringsten Priorität gemäß Meldungsalter (`REMOVE_LOW_PRIORITY`) und/oder Ausschließen der neuesten Meldungen (`REJECT_NEWEST`).
  - Implementierung der Produzentenreaktion auf die Obergrenzen für die Fluss-Steuerung (`FLOW_CONTROL`) auf der Grundlage der einzelnen *Produzenten*, und nicht der einzelnen Verbindungen. (Bei der bisherigen Implementierung wurden alle Produzenten auf einer Verbindung beendet, wenn über die Verbindung zu viele Meldungen bei einem Ziel eingingen.) Durch die produzentenbasierte Fluss-Steuerung werden nur diejenigen Produzenten auf einer Verbindung beendet, die dem überlasteten Ziel zugeordnet sind. Alle anderen Produzenten auf der Verbindung können weiterhin Meldungen an andere Ziele senden.

- Möglichkeit, dass ein Administrator ein bestimmtes Ziel unterbricht (und wieder aufnimmt). Sie können die Zustellung von Meldungen von Produzenten zum Ziel und/oder vom Ziel zu den Konsumenten unterbrechen. Dies erfolgt über zwei neue `imqcmd`-Unterbefehle: `pause` und `resume`, wie im Folgenden dargestellt:
  - `imqcmd pause dst -n myQueue -t q -pst PRODUCERS`
  - `imqcmd resume dst -n myQueue -t q`

Weitere Informationen finden Sie im Handbuch *Message Queue Administration Guide*.

Verbesserte Steuerung des Meldungsflusses beim Java-Client

Die Client-Laufzeit von Message Queue 3.5 verwaltet den Meldungsfluss sowohl auf *Konsumenten*- als auch auf Verbindungsbasis. Sie können eine Obergrenze für die Anzahl der pro Konsument gepufferten Meldungen festlegen und damit verhindern, dass ein einzelner Konsument von den anderen Konsumenten zu stark ausgelastet wird. Durch diese Funktion kann auch bei einer Warteschlangenzustellung an mehrere Konsumenten die Zustellung der Meldungen besser auf die einzelnen Konsumenten verteilt werden. Außerdem erleichtert sie die Verwaltung von Speicherressourcen in der Client-Laufzeit von Message Queue.

Ein neues Connection Factory-Attribut, `imqConsumerFlowLimit`, begrenzt die Anzahl der gepufferten Meldungen *pro Konsument* für alle Konsumenten mit einer gemeinsamen Verbindung. Wenn die Anzahl der Meldungen in einem Konsumentenpuffer unter einen bestimmten Prozentsatz (`imqConsumerFlowThreshold`) von `imqConsumerFlowLimit` fällt, kann der Broker eine weitere Gruppe von Meldungen an die Client-Laufzeit zum Verbrauch durch diesen Konsumenten zustellen. Beachten Sie Folgendes: Falls die Gesamtzahl an Meldungen, die für alle Konsumenten auf einer Verbindung gepuffert werden, den Wert für `imqConnectionFlowLimit` überschreitet, wird die Zustellung von Meldungen über die Verbindung so lange unterbrochen, bis die Gesamtzahl wieder unter diese Grenze fällt.

Bei der bisherigen Implementierung der Steuerung des Meldungsflusses beim Client können Sie eine Grenze für die Anzahl der Meldungen festlegen, die in der Client-Laufzeit gepuffert werden und auf die Konsumierung warten (`imqConnectionFlowLimit`). Der Zweck dieser Funktion lag in der Begrenzung des zum Puffern von Meldungen verwendeten Client-Speichers, um zu verhindern, dass es bei langsam konsumierenden Clients aufgrund fehlenden Arbeitsspeichers zu einem Systemabsturz kommt. Diese Funktion wurde auf der Verbindungsebene implementiert, was zu folgendem Problem führte: Wenn eine Verbindung mehrere Konsumenten unterstützte und eine große Anzahl von Meldungen für einen bestimmten Konsumenten eintraf, konnten die anderen Konsumenten keine Meldungen mehr empfangen.

Weitere Informationen finden Sie im Handbuch *Message Queue Java Client Developer's Guide*.

Neue Zielmetrik

Message Queue 3.5 beinhaltet eine verbesserte Meldungs- und Konsumentenprotokollierung pro Ziel, um bessere Überwachung und Steuerung von Arbeitsspeicher und Verwendung zu ermöglichen.

Die neue Metrik erscheint als Ausgabe des neuen Unterbefehls `imqcmd metrics dst` . Mit diesem Befehl werden die kumulativen Gesamtwerte (seit dem Beginn der Probennahme), die aktuellen Werte, die Durchschnittswerte (aus den gezogenen Stichproben) und die Spitzenwerte (seit Beginn der Probennahme) sowohl für die Meldungs- als auch für die Konsumentenmetrik angezeigt.

Mit dem Befehl `imqcmd metrics dst -m ttl` beispielsweise werden folgende Informationen zurückgegeben:

- Meldungsfluss
  - Meldungsfluss eingehend: kumulativer Gesamtwert, Rate
  - Meldungsfluss ausgehend: kumulativer Gesamtwert, Rate
- Im Broker gespeicherte Meldungen (nicht bestätigte Meldungen im Arbeitsspeicher oder im Persistenzspeicher)
  - Anzahl an Meldungen: aktuell, Spitze, Durchschnitt
  - Meldungsgröße: aktuell, Spitze, Durchschnitt
- Bisher größte Meldung in Byte

Mit dem Befehl `imqcmd metrics dst -m con` werden folgende Informationen zurückgegeben:

- Zahl aktiver Konsumenten: aktuell, Spitze, Durchschnitt (siehe „Verbesserte Verfahren für die Warteschlangenzustellung (Enterprise Edition)“ auf Seite 10)
- Zahl der Backup-Konsumenten: aktuell, Spitze, Durchschnitt (siehe „Verbesserte Verfahren für die Warteschlangenzustellung (Enterprise Edition)“ auf Seite 10)

Weitere Informationen finden Sie im Handbuch *Message Queue Administration Guide*.

Entfernte API-Überwachung (Enterprise Edition)

Message Queue 3.5 bietet eine meldungsbasierte API, mit der entfernte (oder lokale) JMS-Clients die Broker-Metrik auf einfache Weise überwachen und analysieren können. Die API beruht darauf, dass der Broker in der Lage ist, Meldungen zu erstellen, die Metrikinformationen zum Broker selbst, der Java VM und zu einzelnen Zielen enthält (siehe „Neue Zielmetrik“ auf Seite 8). Diese Meldungen werden, je nach der überwachten Einheit, an spezielle Themenziele gesendet, immer wenn mindestens ein Konsument ein Abonnement für diese Ziele aufweist. Der Konsumenten-Client kann dann die Meldungen abrufen, sie mit einer Kopfzeileneigenschaft (`type`) filtern und dann die enthaltenen Metrikinformationen extrahieren.

Weitere Informationen finden Sie in folgenden Handbüchern: *Message Queue Administration Guide* und *Message Queue Java Client Developer's Guide*.

(Message Queue unterstützte bisher nur die logische Protokollierung von Broker-Metrikdaten und entfernte Abfragen von Metrikinformationen mithilfe der Administrationsdienstprogramme von Message Queue. Diese Funktionen lieferten zwar wichtige Metrikdaten, erlaubten jedoch keine einfache Analyse dieser Daten.)

Message Queue Ressourcenadapter für JMS (J2EE Application Server-Unterstützung)

Message Queue 3.5 beinhaltet einen JMS-Ressourcenadapter zur Integration des JMS-Meldungsdiensts von Message Queue in einen kompatiblen J2EE-Anwendungsserver.

Ein Ressourcenadapter ist ein standardisiertes Verfahren zur Integration zusätzlicher Funktionen in einen J2EE-Anwendungsserver (durch Anschluss an EIS, ein Nachrichtensystem usw.) gemäß der Spezifikation J2EE Connector Architecture Specification (JCA 1.5). Mit dieser Architektur kann beispielsweise jeder beliebige J2EE-Anwendungsserver JMS-Messaging unterstützen, indem eine Verbindung zu einem JMS-Provider hergestellt wird, der JCA 1.5 implementiert: J2EE-Komponenten, die in der Anwendungsserverumgebung bereitgestellt und ausgeführt werden, können über den integrierten JMS-Provider (Client-Laufzeit und Server) JMS-Meldungen austauschen.

Weitere Informationen finden Sie im Handbuch *Message Queue Administration Guide*.

Benutzerdefinierte Meldungsbestätigung

Message Queue unterstützt nun den JMS-Client-Bestätigungsmodus `CLIENT_ACKNOWLEDGE`, mit dem ein JMS-Client den Meldungskonsum explizit bestätigt. Im Modus `CLIENT_ACKNOWLEDGE` ruft der Client die Methode `acknowledge()` eines Meldungsobjekts auf, was dazu führt, dass die Sitzung alle Meldungen bestätigt, die seit dem letzten Aufruf der Methode von der Sitzung konsumiert wurden.

Message Queue 3.5 verbessert diese Funktion, indem Sie in die Lage versetzt werden, *einzelne* Meldungen zu bestätigen. Sie können also auch nur eine bestimmte Meldung bestätigen und müssen nicht mehr alle bis zu diesem Zeitpunkt konsumierten Meldungen insgesamt bestätigen. Dies erfolgt im Code, indem das Meldungsobjekt in einen besonderen Message Queue-Meldungstyp umgewandelt wird, der in einer neuen `acknowledge()`-Methode aufgerufen wird. Dadurch können Sie vom JMS-Standard abweichen, um den besonderen Anforderungen Ihrer Anwendung gerecht zu werden.

Weitere Informationen finden Sie im Handbuch *Message Queue Java Client Developer's Guide*.

Verbesserte Verfahren für die Warteschlangenzustellung (Enterprise Edition)

Die Implementierung der Warteschlangenzustellung an mehrere Konsumenten, die bisher in drei unterschiedlichen Warteschlangenzustellungsverfahren (einzeln, Failover und Round-Robin) implementiert wurde, hat sich geändert. Message Queue 3.5 verwendet einen allgemeineren Ansatz, bei dem bei der Zustellung ein Lastenausgleich zwischen einer konfigurierbaren Anzahl von aktiven Konsumenten (und Backup-Konsumenten) besteht. Die Message Queue 3.5-Implementierung beruht auf folgenden neuen Zielattributen:

- `maxNumActiveConsumers`: Gibt die Anzahl der Konsumenten – einer oder viele – an, die in einer Warteschlangenzustellung mit Lastenausgleich aktiv sind.
- `maxNumBackupConsumers`: Gibt die Anzahl der Backup-Konsumenten – keiner oder viele – an, die aktive Konsumenten ersetzen können, wenn diese ausfallen.

(Neue Konsumenten werden zurückgewiesen, wenn die Anzahl der Konsumenten die Summe dieser beiden Attribute übersteigt.)

Message Queue Platform Edition unterstützt eine Warteschlangenzustellung mit Lastenausgleich für bis zu zwei Konsumenten und die Enterprise Edition unterstützt unbegrenzt viele Konsumenten.

Der neue Lastenausgleichsmechanismus berücksichtigt die Meldungskonsumierungsrate der verschiedenen Konsumenten. Er funktioniert folgendermaßen:

- Eine ursprüngliche Anzahl von Meldungen in der Warteschlange in einem Ziel werden in Blöcken mit einer konfigurierbaren Größe (Attribut `consumerFlowLimit` des Ziels) an verfügbare aktive Konsumenten weitergeleitet (in der Reihenfolge, in der sie beim Ziel registriert sind). Sobald diese Meldungen zugestellt wurden, werden weitere Meldungen, die an einem Ziel ankommen, einzeln an die Konsumenten weitergeleitet, immer wenn ein Konsument verfügbar wird (d. h., wenn die Konsumenten alle bisher an sie zugestellten Meldungen bestätigen). Wenn ein aktiver Konsument ausfällt, wird der erste Backup-Konsument aktiviert und übernimmt die Arbeit des ausgefallenen Konsumenten.
- In einer Broker-Cluster-Umgebung können die Zustellmechanismen so eingestellt werden, dass lokale Konsumenten priorisiert werden. Mit einem neuen Zielattribut, `localDeliveryPreferred`, können Sie festlegen, dass die Meldungen nur dann an entfernte Konsumenten zugestellt werden sollen, wenn sich auf dem lokalen Broker (dem Broker, auf dem das Ziel erstellt wurde) keine Konsumenten befinden. Dadurch können Sie die Leistung in Situationen erhöhen, wo die Weiterleitung an entfernte Clients (über die entsprechenden Home-Broker) den Durchsatz verlangsamen könnte. (Bei diesem Attribut darf der Zielbereich nicht auf ausschließlich lokale Zustellung beschränkt sein – siehe „Verbesserte Cluster-Leistung (Enterprise Edition)“ auf Seite 11.)

Weitere Informationen finden Sie im Handbuch *Message Queue Administration Guide*.

Verbesserte Cluster-Leistung (Enterprise Edition)

In einer Broker-Cluster-Umgebung werden die Ziele auf allen Brokern repliziert und alle an diese Ziele zugestellten Meldungen werden an alle Broker weitergeleitet, bei denen Konsumenten für diese Ziele registriert wurden, selbst wenn nur ein kleiner Prozentsatz dieser Meldungen an jeden einzelnen Konsumenten zugestellt wird (z. b. wenn ein dauerhafter Abonnent Auswahlkriterien verwendet oder ein Warteschlangenempfänger an einer Warteschlangenzustellung mit Lastenausgleich beteiligt ist). Dieser Datenverkehr zwischen verschiedenen Brokern kann zu einer übermäßig großen Menge von Meldungen führen, insbesondere wenn ein neuer Konsument aktiv wird. Zur Reduzierung von übermäßigem Broker-zu-Broker-Verkehr in einem Cluster bietet Message Queue 3.5 folgende Verbesserungen:

- Neue Mechanismen zur Fluss-Steuerung, die die Zustellung von Meldungen an eine Konsumentenverbindung regeln. Mit anderen Worten: Der *Konsument* steuert die Zustellung der Meldungen (vom Ziel zur Client-Laufzeit), wodurch die Weiterleitung unnötiger Meldungen von Broker zu Broker vermieden wird. (Durch diese Mechanismen sollten außerdem Protokollstaus auf Seiten der Client-Laufzeit vermieden werden – siehe „Verbesserte Steuerung des Meldungsflusses beim Java-Client“ auf Seite 7).
- Geänderte Implementierung der Warteschlangenzustellung an mehrere Konsumenten (siehe „Verbesserte Verfahren für die Warteschlangenzustellung (Enterprise Edition)“ auf Seite 10) zur Reduzierung der unnötigen Weiterleitung von Meldungen von Broker zu Broker. Zu dieser Implementierung gehört ein neues Attribut für das Warteschlangenziel, `localDeliveryPreferred`, mit dem Sie angeben können, dass lokale Konsumenten bei der Warteschlangenzustellung an mehrere Konsumenten Vorrang gegenüber den entfernten Konsumenten haben sollen (siehe „Verbesserte Verfahren für die Warteschlangenzustellung (Enterprise Edition)“ auf Seite 10).

Weitere Informationen finden Sie im Handbuch *Message Queue Administration Guide*.

Lokale Ziele (Enterprise Edition)

Mit einem neuen Zielattribut, `isLocalOnly`, können Sie festlegen, dass ein Ziel auf die Zustellung von Meldungen an lokale Konsumenten (Konsumenten mit Verbindung zu dem Broker, auf dem das Ziel erstellt wurde) beschränkt sein soll und keine Meldungen an Konsumenten zustellt, die mit anderen Brokern im Cluster verbunden sind. Ebenso kann das Ziel nur Meldungen empfangen, die von lokalen Produzenten gesendet wurden. Mit dieser Eigenschaft können Sie unabhängige, nicht miteinander interagierende Ziele mit demselben Namen auf verschiedenen Brokern innerhalb eines Clusters erstellen und Failover-Szenarien einrichten, bei denen eine Meldung an zwei Ziele geschickt wird, für den Fall, dass eines davon ausfällt.

Weitere Informationen finden Sie im Handbuch *Message Queue Administration Guide*.

Sichere Broker-Cluster (Enterprise Edition)

In Situationen, bei denen eine sichere, verschlüsselte Meldungszustellung zwischen Client und Meldungsserver erforderlich ist, unterstützt Message Queue 3.5 jetzt eine sichere Zustellung von Meldungen zwischen den Brokern in einem Cluster. Um eine sichere, verschlüsselte Zustellung der Meldungen innerhalb eines Clusters zu erreichen, müssen Sie den internen Cluster-Verbindungsdienst für die Verwendung eines SSL-basierten Transportprotokolls konfigurieren.

Weitere Informationen finden Sie im Handbuch *Message Queue Administration Guide*.

Verbesserte Leistung des Persistenzspeichers

Die Implementierung des Message Queue-Flatfile-Datenspeichers und des JDBC-kompatiblen Datenspeichers wurde in Message Queue 3.5 zur Verbesserung der Leistungsfähigkeit geändert. Diese Verbesserungen werden in den folgenden Unterabschnitten beschrieben. Weitere Informationen finden Sie im Handbuch *Message Queue Administration Guide*.

Integrierte Persistenz (Flatfile-Datenspeicher)

Die Leistungsverbesserung im Message Queue-Flatfile-Datenspeicher beinhaltet interne Änderungen am Datenformat, das nur bei Aufrüstungen von Message Queue 3.01 (oder früher) auf Message Queue 3.5 sichtbar wird.

Die Migration des Dateispeichers erfolgt automatisch, wenn eine Message Queue 3.5-Broker-Instanz ursprünglich gestartet wurde und auf eine frühere Version des Dateispeichers verweist. Eine Kopie des früheren Dateispeichers wird im Objektverzeichnis gespeichert. Diese muss nach Abschluss der Migration manuell gelöscht werden. Um den früheren Datenspeicher automatisch zu entfernen, wenn nicht genügend Speicherplatz für zwei Kopien des Speichers vorhanden ist, können Sie den Broker der Version 3.5 mit einer zusätzlichen Option starten, wie in folgendem Befehl dargestellt:

```
imqbrokerd -upgrade-store-nobackup
```

(Die Zeichenfolge `upgrade-store-nobackup` darf keine Leerzeichen enthalten.)

Der Root des neuen Flatfile-Datenspeichers hat sich geändert von:

…/instances/*Instanzenname*/filestore/

in:

…/instances/*Instanzenname*/fs350/.

Außerdem wurde das Message Queue-Befehlsprogramm (imqcmd) verbessert, sodass es nun eine Dateispeichermetrik bietet:

```
imqcmd metrics dst -n Zielname -t Typ -m dsk
```

Zu imqcmd gehört ein neuer Komprimierungsbefehl:

```
imqcmd compact dst -n Zielname -t Typ
```

Integrierte Persistenz (JDBC™-kompatibler Datenspeicher)

Der Message Queue JDBC-kompatible Datenspeicher wurde in Message Queue 3.5 geändert, sodass er Verbesserungen bei der Broker-Speicherverwaltung (siehe „Verbesserte Steuerung des Meldungsflusses beim Broker“ auf Seite 6 und „Neue Zielmetrik“ auf Seite 8) und eine größere Palette von Datenbankherstellern unterstützt. Bei den Änderungen zur Unterstützung der verbesserten Speicherverwaltung (Kategorisieren der Meldungen nach Ziel) handelt es sich um Schemaänderungen, die in transparenter Weise erfolgen, die Unterstützung für zusätzliche Datenbankhersteller beinhaltet jedoch Änderungen in der Art und Weise, wie die integrierte Persistenz konfiguriert wird.

Die Migration des JDBC-kompatiblen Dateispeichers erfolgt automatisch, wenn eine Message Queue 3.5-Broker-Instanz ursprünglich mit einer früheren Version des Persistenzspeichers gestartet wurde. Die alten Tabellen bleiben jedoch erhalten und müssen nach Abschluss der Migration mit dem neuen Befehl `imqdbmgr delete oldtbl` manuell gelöscht werden. Um die früheren Tabellen automatisch zu löschen, wenn nicht genügend Speicherplatz für zwei Kopien des Speichers vorhanden ist, können Sie den Broker ursprünglich mit einer neuen Option starten. `imqbrokerd -upgrade-store-nobackup`.

Was die erweiterte Datenbankunterstützung betrifft: Bisher konnten die SQL-Anweisungen nicht für die zu integrierende JDBC-kompatible Datenbank angepasst werden (die SQL-Anweisungen wurden intern generiert). Message Queue 3.5 bietet nun neue Eigenschaften zur Instanzenkonfiguration, mit denen Sie den SQL-Code, der das Message Queue-Datenbankschema erstellt, benutzerdefiniert anpassen können. Es gibt für jede Datenbanktabelle eine konfigurierbare Eigenschaft: Die Eigenschaft ist der SQL-Code, mit dem die Tabelle erstellt wird. Diese Eigenschaften werden zur ordnungsgemäßen Angabe der von der integrierten Datenbank verwendeten Datentypen benötigt. Beispiele werden auf der Grundlage einer in PointBase (nicht in die frühere Cloudscape-Datenbank) eingebetteten Datenbank bereitgestellt.

Instanzenbezogene Authentifizierung und Autorisierung

Standardmäßig ermöglicht Message Queue 3.5, dass jede Instanz über ein eigenes dateibasiertes Benutzer-Repository und eine eigene Zugriffssteuerungsdatei verfügt, die sich beide an folgendem Standard-Speicherort befinden: …/`instances`/*Instanzenname*/`etc`/. Diese beiden Dateien werden erstellt, wenn eine Broker-Instanz erstmals gestartet wird. Wenn der Broker diese Dateien an einem alten Speicherort findet (üblicherweise bei einer Aufrüstung von einer früheren Version), kopiert er die Dateien an den instanzenbezogenen Speicherort. Wenn der Broker diese Dateien nicht am alten Speicherort findet (üblicherweise bei Neuinstallation), legt er Standardversionen der Dateien am instanzenbezogenen Speicherort ab.

Zur Unterstützung instanzenbezogener Repositories wurde die Option -i *instanceName* zum Dienstprogramm für die Benutzerverwaltung (`imqusermgr`) hinzugefügt. Dies gibt das instanzenbezogene Benutzer-Repository an, auf das sich die einzelnen `imqusermgr`-Befehle beziehen.

Weitere Informationen finden Sie im Handbuch *Message Queue Administration Guide*.

Bisher verwendeten alle Instanzen eines Brokers auf einem Computer standardmäßig dasselbe dateibasierte Benutzer-Repository (und damit dasselbe Passwort für die Client-Anwendung) und dieselbe Zugriffssteuerungsdatei. Sie konnten jedoch die einzelnen Broker-Instanzen so konfigurieren, dass sie einen speziellen Speicherort für das LDAP-Benutzer-Repository oder eine spezielle Zugriffssteuerungsdatei verwendeten, die beide in der Konfigurationsdatei für die Instanz angegeben wurden.

RPM-basierte Linux-Installation

Die Installation von Message Queue 3.5 unter Linux erfolgt mithilfe des Red Hat Package Manager (RPM), eines über die Befehlszeile gesteuerten Paketverwaltungssystems zur Installation, Deinstallation, Überprüfung, Abfrage und Aktualisierung von Softwarepaketen (RPMs).

Außerdem hat sich die installierte Verzeichnisstruktur für Message Queue unter Linux geändert und entspricht nun den Standard-Speicherorten für nicht im Lieferumfang enthaltenen Produkte unter Linux. (Sowohl Solaris™ als auch Linux-Plattformen weisen Standards auf, die davon abhängen, ob ein Produkt im Lieferumfang des Betriebssystems integriert ist oder nicht.) Insbesondere gibt es unter Linux, ähnlich wie bei Solaris, kein Root-Installationsverzeichnis für Message Queue mehr.

Weitere Informationen finden Sie im Handbuch *Message Queue Installation Guide*.

Unterstützung für das Solaris-Betriebssystem, X86 Platform Edition

Unter Solaris 9 wird Message Queue 3.5 neben SPARC-Prozessoren auch für X86-Prozessoren unterstützt.

Hardware- und Softwareanforderungen

Die für diese Version erforderliche Hard- und Software sowie die unterstützten Produkte und Plattformen werden ausführlich im *Message Queue Installation Guide* beschrieben.

Behobene Fehler

Dieser Abschnitt enthält jeweils eine kurze Beschreibung der behobenen Fehler:

- Tabelle 2 auf Seite 16 beschreibt die in Message Queue 3.5 SP1 behobenen Fehler.
- Tabelle 3 auf Seite 17 beschreibt die in Message Queue 3.5 behobenen Fehler.

Ältere Listen mit behobenen Fehlern finden Sie in folgenden Dokumenten:

- Für Message Queue 3.0.1 Service Pack 2 finden Sie die Informationen in *Message Queue 3.0.1 Service Pack 2 Release Notes* unter:

  `http://docs.sun.com/coll/S1_MessageQueue_301_SP2`

- Für Message Queue 3.0.1 finden Sie die Informationen in *Message Queue 3.0.1 Release Notes* unter:

  `http://docs.sun.com/coll/S1_MessageQueue_301`

- Für Message Queue 3.0 finden Sie die Informationen in *Message Queue 3.0 Release Notes* unter:

  `http://docs.sun.com/coll/S1_MessageQueue_30`

Weitere Einzelheiten zu den behobenen Fehlern finden Sie im vollständigen Bericht zu diesem Thema auf der Java Developer Connection-Website unter

`http://developer.java.sun.com/developer/bugParade`

In Message Queue 3.5 SP1 behobene Fehler

Tabelle 2 nennt und beschreibt die in Message Queue 3.5 SP1 behobenen Fehler. (Tabelle 3 auf Seite 17 nennt und beschreibt die in Message Queue 3.5 behobenen Fehler.)

Tabelle 2 In Message Queue 3.5 SP1 behobene Fehler

Fehlernummer	Beschreibung
4942723	Beim Broker kann es beim Senden großer Meldungen mit der Option für gemeinsam verwendete Thread-Pools vorkommen, dass nicht genügend Arbeitsspeicher zur Verfügung steht.
4944894	Der Broker kann bei Verwendung des gemeinsamen Thread-Pools gelegentlich den Ausnahmefehler `CancelledKeyException` ausgeben.
4947239	Wiederholtes Erstellen und Schließen von Produzenten führt zu einem geringen Wachstum des Client-Arbeitsspeichers.
4947993	Ziel oder dauerhaftes Element kann nicht mit aktivem dauerhaftem Abonnenten vernichtet werden.
4948525	In der Metrikausgabe für Meldungsgröße (eingehend und ausgehend) können negative Zahlen vorkommen. Dies geschieht, wenn mehr als 2143510810 Byte gesendet wurden.
4948563	Paketkonvertierung: Für jede 2.0 SP1-Meldung, die an einen 3.5-Broker gesendet wird, wird eine INFO-Meldung angezeigt. Jedes Mal, wenn ein 2.0 SP1-Client eine Meldung an einen 3.5-Broker sendet, wird folgende INFO-Meldung angezeigt `[04/Nov/2003:10:34:16 PST] Internal Error: Unknown ProducerUID 0`
4949781	Cluster-Broadcaster kann nicht verwendet werden – Fehler beim Starten des Brokers.
4952332	Die Meldungen werden möglicherweise in der falschen Reihenfolge zugestellt, wenn der primäre Konsument ausfällt und stattdessen ein Backup-Konsument auf derselben Verbindung verwendet wird.
4956748	Mit der Oracle-Datenbank kann kein Master-Broker verwendet werden.
4964703	C-API: Die von der Funktion `MQGetMEssageHeaders()` zurückgegebene `MESSAGE_ID`-Kopfzeile weist nicht das Präfix „ID:“ auf.
4964712	C-API: Die von der Funktion `MQSetMEssageHeaders()` festgelegte `MESSAGE_ID`-Kopfzeile wird beim Senden der Meldung nicht ignoriert.
4969583	C-API: Derselbe Meldungs-Handle sollte in der Lage sein, `MQAcknowledgeMessages()` mehrmals aufzurufen.
4983150	Das Flag `JMSRedlivered` wird nicht festgelegt, wenn ein Broker neu gestartet wird und die Meldung erneut zustellt.
4983699	Der Broker verliert die Ausnahmen, die vom Speicher ausgegeben werden, wenn eine Meldung nicht persistent gemacht werden kann.

In Message Queue 3.5 behobene Fehler

Tabelle 3 nennt und beschreibt die in Message Queue 3.5 behobenen Fehler.

Tabelle 3 In Message Queue 3.5 behobene Fehler

Fehlernummer	Beschreibung
4449354	In ganz seltenen Fällen führt der gleichzeitige Aufruf der Methoden `Connection.stop`, `Connection.start` und `Connection.close` mit den Methoden `Session.recover` und `Session.rollback` (in separaten Threads) zu einer unerwarteten Reihenfolge bei der erneuten Sendung von Meldungen.
4630183	Bei der Vernichtung eines Ziels verbleiben dauerhafte Abonnements auf dem Broker.
4753010	Uneingeschränktes Wachstum im nativen Heap-Segment (Java-Prozess) mit der Server-VM.
4761626	Umfangreiche Konsumentenerstellungs- und -löschvorgänge bei automatischer Erstellung von Warteschlangen kann zu Meldungsverlusten führen.
4855307	Der Broker kann sich nicht bei einem LDAP-Repository authentifizieren, da die Standardkonfiguration einen alten Eigenschaftsnamen (`bindDN`) verwendet.
4883126	Die Funktion zum automatischen Wiederherstellen einer Verbindung arbeitet nicht ordnungsgemäß.
4888270	Die erneute Übertragung einer Meldung, die ursprünglich in einer Transaktion gesendet wurde, verursacht einen Broker-Fehler.
4431924	`imqadmin`: Modale Dialogfelder können sich gegenseitig sperren (Deadlock). Die Administrationskonsole (`imqadmin`) verwendet anwendungsgebundene Dialogfelder. Die meisten dieser Dialogfelder werden ausdrücklich durch die Interaktion mit der grafischen Benutzeroberfläche aufgerufen, beispielsweise durch die Auswahl des Menüs „Broker hinzufügen“. Ein Dialog kann aber auch aufgrund einer getrennten Brokerverbindung angezeigt werden. Ist mehr als ein Dialogfeld geöffnet, wird die Administrationskonsole gesperrt. Sie können keines der modalen Dialogfelder mit der Option „Schließen“ beenden.
4703406	QueueBrowser sollte funktionieren, ohne dass zunächst `connection.start()` aufgerufen werden muss. `Connection.start()` muss auf einer Verbindung aufgerufen werden, bevor `QueueBrowser` eine Warteschlange durchsuchen kann. Wenn Sie `Connection.start()` nicht aufrufen, blockiert die QueueBrowser-Aufzählung `nextElement()` und meldet schließlich eine `java.util.NoSuchElementException`.
4866814	Unter Solaris kann der Broker keine Fehler und Warnungen mit „syslog“ protokollieren, wenn er mit einer 64-Bit-JVM gestartet wurde (der Broker wurde mit „-vmargs -d64“ gestartet). Dies geschieht, da die Beta-Version von Message Queue keine 64-Bit-Version der Bibliothek`libimqutil.so.1` enthält.
4872121	Der Broker startet nicht in einem nicht vernetzten System, das eine andere IP-Adresse als 127.0.0.1 aufweist.
4879902	Langsamer Arbeitsspeicheranstieg im Broker.

Tabelle 3 In Message Queue 3.5 behobene Fehler *(Fortsetzung)*

Fehlernummer	Beschreibung
4881968	Neue Überwachungs-Clients können nicht erstellt werden, wenn `imq.autocreate.topic` auf „false“ eingestellt ist.
4884827	CTS1.3 MDB/EJB CMT-Tests schlagen bei Message Queue 3.5 und AppServer 7.0 fehl.
4885654	Die Produzenten können ausfallen, wenn eine neue Meldung an ein automatisch erstelltes Ziel veröffentlicht wird, während das System gleichzeitig das Ziel beendet.
4887506	Beim Wechsel von einem einzelnen primären Konsumenten auf einen Backup-Konsumenten aufgrund eines Fehlers werden die Meldungen möglicherweise in der falschen Reihenfolge zugestellt.
4888939	C- und Java-Clients auf einem Ziel mit FLOW_CONTROL-Reaktionen nehmen möglicherweise keine Meldungen mehr entgegen, wenn die maximale Größe eines Ziels (`maxNumMsgs`) sehr klein ist (< 5 Meldungen).
4889002	Die Eigenschaft „imq.transaction.autorollback“ wird in 3.5beta nicht unterstützt.
4891874	Die konsumentenbasierte Fluss-Steuerung kann dazu führen, dass keine Meldungen mehr an die Konsumenten zugestellt werden. Dieses Problem tritt mit größerer Wahrscheinlichkeit mit 4896133: ConnectionConsumers und den Message Driven Beans des Sun Java System Application Server auf.
4895262	HTTPS-Clients konnten keine Verbindung zum Broker über `HTTPSTunnelServlet` herstellen.
4897500	Wenn ein Client in einem Cluster „unsubscribe()“ aufruft, um ein dauerhaftes Abonnement zu entfernen, wird dieses nur von dem Broker entfernt, mit dem der Client verbunden ist. Dies bedeutet, dass die an andere Broker gerichteten Meldungen weiterhin für diesen Abonnementen gespeichert werden.
4898020	Die Broker von Message Queue 3.0.* und Message Queue 3.5 können nicht zusammen in einem Cluster verwendet werden. Der Start eines gemischten Clusters führt zu einem Fehler beim 3.0.1-Broker: Nichtübereinstimmung bei Konfiguration: Verbindung mit Broker [ ] wird abgebrochen, weil folgende Konfigurationseigenschaften nicht übereinstimmen – null imq.queue.deliverypolicy
4888983	`imqcmd list dur` zeigt keine dauerhaften Abonnenten mit demselben dauerhaften Namen an.

Wichtige Informationen

In diesem Abschnitt finden Sie die aktuellsten Informationen, die nicht in der eigentlichen Produktdokumentation enthalten sind: In diesem Abschnitt werden folgende Themen behandelt:

Installationshinweise

Informationen zu den Systemvoraussetzungen, den unterstützten Softwareplattformen und -produkten, Anweisungen zu den vor der Installation zu erledigenden Aufgaben, Aufrüstungsverfahren und alle anderen Informationen, die für die Installation von Message Queue auf den Plattformen Solaris, Linux und Windows relevant sind, finden Sie im *Message Queue Installation Guide*.

Kompatibilität

In diesem Abschnitt werden Kompatibilitätsprobleme in Message Queue 3.5 SP1 und Message Queue 3.5 behandelt.

Probleme in Bezug auf die nächste Hauptversion von Message Queue

Im Folgenden werden Änderungen mit Auswirkungen auf die Kompatibilität vorgestellt, die möglicherweise in der nächsten Hauptversion von Message Queue eingeführt werden. Sie erhalten diese Informationen schon jetzt, damit Sie sich auf diese Änderungen vorbereiten können.

- Message Queue-Client-Unterstützung für alle Versionen von J2SE 1.3 wird aufgegeben. J2SE 1.4 wird weiterhin unterstützt.

- Alle Message Queue-Befehlszeilenschnittstellen werden geändert. Die Option zur Angabe eines Passworts als Befehlszeilenargument fällt weg. Beispiel:

  ```
  imqbrokerd -ldappassword <passwd> imqcmd -p <passwd>
  ```

  Es werden andere Mechanismen zur Angabe des Passworts bereitgestellt.

- Das Format der Broker-Protokolldatei ändert sich. Anwendungen, die auf das aktuelle Format angewiesen sind, funktionieren möglicherweise nicht mehr.

- Die Speicherorte der einzelnen Dateien von Message Queue kann sich ändern. Dadurch können bestehende Anwendungen beeinträchtigt werden, die vom aktuellen Speicherort bestimmter Message Queue-Dateien abhängig sind.

- Das Programm `imqkeytool` wird möglicherweise aus dem Produkt entfernt. Das J2SE-Schlüsseltool ist der unterstützte Ersatz dafür.

- Message Queue-Clients, die eine Message Queue-Version verwenden, die älter ist als die nächste Hauptversion, können möglicherweise nicht auf die neuen Funktionen zugreifen, die in der neuen Produktversion angeboten werden.

- Es wird kein Fehler ausgegeben, wenn die Funktion `MQAcknowledgeMessages()` von C-Clients für eine bereits bestätigte Meldung aufgerufen wird (über `MQ_CLIENT_ACKNOWLEDGE`). Dieses Verhalten ändert sich möglicherweise.

Probleme bei Message Queue 3.5

Message Queue 3.5 ist generell mit Message Queue 3.0 (und den Folgeversionen 3.0.1, 3.0.1 Service Pack 1 und 3.0.1 Service Pack 2) kompatibel. Es wurden jedoch Änderungen an den Broker-Eigenschaften, den verwalteten Objekten, dem Persistenzschema, den Dateispeicherorten und den Verwaltungswerkzeugen vorgenommen, die eine Aufrüstung der 3.0-Versionen von Message Queue auf Message Queue 3.5 beeinträchtigen können.

Durch die Installation von Message Queue 3.5 wird das Verzeichnis `IMQ_VARHOME` von Message Queue 3.0 nicht überschrieben. Dieses Verzeichnis enthält konfigurations- und sicherheitsbezogene Dateien. Die meisten dieser Daten sind mit Message Queue 3.5 kompatibel und können mithilfe der Anweisungen im Handbuch *Message Queue Installation Guide* beibehalten werden.

Beim Aufrüsten von Message Queue 3.0 auf Message Queue 3.5 können Sie unter anderem auf folgende Probleme stoßen:

- Broker-Kompatibilität
- Änderungen an Eigenschaften und Attributen
- Speicherort der öffentlichen Public .jar-Dateien

Informationen zur Kompatibilität der verwalteten Objekte, zur Client-Kompatibilität und zur Kompatibilität des Verwaltungswerkzeugs finden Sie im Handbuch *Message Queue Installation Guide*.

Broker-Kompatibilität

Ein Message Queue 3.5-Broker kann zwar mit einem Message Queue 3.0-Broker zusammenarbeiten. Es wurden jedoch Änderungen an den Broker-Eigenschaften und dem Persistenzspeicherschema vorgenommen. Einige Message Queue 3.0-Daten sind noch mit Message Queue 3.5 kompatibel. Weitere Informationen finden Sie im Handbuch *Message Queue Installation Guide*.

Änderungen an Eigenschaften und Attributen

In diesem Abschnitt finden Sie eine Zusammenfassung der Änderungen an den Broker-Eigenschaften, den Zielattributen und den Connection Factory-Attributen, die in Message Queue 3.5 vorgenommen wurden.

Broker-Eigenschaften. In den folgenden Tabellen werden die neuen Eigenschaften, die nicht mehr verwendeten Eigenschaften und die Änderungen an Eigenschaftsnamen in Message Queue 3.5 aufgeführt. Einzelheiten hierzu finden Sie in Kapitel 2 des *Message Queue Administration Guide*.

Tabelle 4 Neue Broker-Eigenschaften in Message Queue

Eigenschaftsname	Informationen zu der Funktion
`imq.persist.file.message.max_record.size`	„Verbesserte Leistung des Persistenzspeichers“ auf Seite 12
`imq.persist.file.destination.message.filepool.limit`	„Verbesserte Leistung des Persistenzspeichers“ auf Seite 12
`imq.metrics.topic.enabled`	„Entfernte API-Überwachung (Enterprise Edition)“ auf Seite 8
`imq.metrics.topic.interval`	„Entfernte API-Überwachung (Enterprise Edition)“ auf Seite 8
`imq.metrics.topic.persist`	„Entfernte API-Überwachung (Enterprise Edition)“ auf Seite 8
`imq.metrics.topic.timetolive`	„Entfernte API-Überwachung (Enterprise Edition)“ auf Seite 8
`imq.autocreate.destination.maxNumMsgs`	„Verbesserte Steuerung des Meldungsflusses beim Broker“ auf Seite 6
`imq.autocreate.destination.maxTotalMsgBytes`	„Verbesserte Steuerung des Meldungsflusses beim Broker“ auf Seite 6
`imq.autocreate.destination.maxBytesPerMsg`	„Verbesserte Steuerung des Meldungsflusses beim Broker“ auf Seite 6

Tabelle 4 Neue Broker-Eigenschaften in Message Queue *(Fortsetzung)*

Eigenschaftsname	Informationen zu der Funktion
`imq.autocreate.destination.maxNumProducers`	„Verbesserte Steuerung des Meldungsflusses beim Broker“ auf Seite 6
`imq.autocreate.queue.maxNumActiveConsumers`	„Verbesserte Verfahren für die Warteschlangenzustellung (Enterprise Edition)“ auf Seite 10
`imq.autocreate.queue.maxNumBackupConsumers`	„Verbesserte Verfahren für die Warteschlangenzustellung (Enterprise Edition)“ auf Seite 10
`imq.autocreate.queue.consumerFlowLimit`	„Verbesserte Steuerung des Meldungsflusses beim Java-Client“ auf Seite 7 und „Verbesserte Verfahren für die Warteschlangenzustellung (Enterprise Edition)“ auf Seite 10
`imq.autocreate.topic.consumerFlowLimit`	„Verbesserte Steuerung des Meldungsflusses beim Java-Client“ auf Seite 7
`imq.autocreate.queue.localDeliveryPreferred`	„Verbesserte Cluster-Leistung (Enterprise Edition)“ auf Seite 11
`imq.autocreate.destination.isLocalOnly`	„Lokale Ziele (Enterprise Edition)“ auf Seite 11

Tabelle 5 Nicht mehr verwendete Broker-Eigenschaften in Message Queue 3.5

Eigenschaftsname
`imq.persist.file.message.fdpool.limit`
`imq.persist.file.message.filepool.limit`
`imq.redelivered.optimization`
`imq.queue.deliverypolicy`

In den folgenden Tabellen werden die neuen und die nicht mehr verwendeten Zielattribute in Message Queue 3.5 aufgeführt. Einzelheiten hierzu finden Sie in Kapitel 6 des *Message Queue Administration Guide*.

Tabelle 6 Neue Zielattribute in Message Queue

Zieltyp	Attributname	Informationen zu der Funktion
Warteschlange und Thema	`maxNumMsgs`	„Verbesserte Steuerung des Meldungsflusses beim Broker“ auf Seite 6
Warteschlange und Thema	`maxTotalMsgBytes`	„Verbesserte Steuerung des Meldungsflusses beim Broker“ auf Seite 6
Warteschlange und Thema	`limitBehavior`	„Verbesserte Steuerung des Meldungsflusses beim Broker“ auf Seite 6
Warteschlange und Thema	`maxBytesPerMsg`	„Verbesserte Steuerung des Meldungsflusses beim Broker“ auf Seite 6
Warteschlange und Thema	`maxNumProducers`	„Verbesserte Steuerung des Meldungsflusses beim Broker“ auf Seite 6
Nur Warteschlange	`maxNumActiveConsumers`	„Verbesserte Verfahren für die Warteschlangenzustellung (Enterprise Edition)“ auf Seite 10
Nur Warteschlange	`maxNumBackupConsumers`	„Verbesserte Verfahren für die Warteschlangenzustellung (Enterprise Edition)“ auf Seite 10
Warteschlange und Thema	`consumerFlowLimit`	„Verbesserte Steuerung des Meldungsflusses beim Java-Client“ auf Seite 7 und „Verbesserte Verfahren für die Warteschlangenzustellung (Enterprise Edition)“ auf Seite 10
Nur Warteschlange	`localDeliveryPreferred`	„Verbesserte Cluster-Leistung (Enterprise Edition)“ auf Seite 11
Warteschlange und Thema	`isLocalOnly`	„Lokale Ziele (Enterprise Edition)“ auf Seite 11

Tabelle 7 Nicht mehr verwendete Zielattribute in Message Queue 3.5

Zieltyp	Attributname
Warteschlange	`QueueDeliveryPolicy`

Connection Factory-Attribute. In den folgenden Tabellen werden die neuen Connection Factory-Attribute in Message Queue 3.5 aufgeführt. Einzelheiten hierzu finden Sie in Kapitel 4 des *Message Queue Java Client Developer's Guide*. Hinweis: die Connection Factory-Attribute aus Message Queue 3.0 werden in Message Queue 3.5 weiterhin unterstützt. Die Unterstützung wird bis zur nächsten Hauptversion von Message Queue beibehalten.

Tabelle 8 Neue Connection Factory-Attribute in Message Queue

Attributname	Informationen zu der Funktion
`imqAddressList`	„Failover der Java-Client-Verbindung (Enterprise Edition)“ auf Seite 5
`imqAddressListBehavior`	„Failover der Java-Client-Verbindung (Enterprise Edition)“ auf Seite 5
`imqAddressListIterations`	„Failover der Java-Client-Verbindung (Enterprise Edition)“ auf Seite 5
`imqReconnectEnabled`	„Failover der Java-Client-Verbindung (Enterprise Edition)“ auf Seite 5
`imqReconnectAttempts`	„Failover der Java-Client-Verbindung (Enterprise Edition)“ auf Seite 5
`imqReconnectInterval`	„Failover der Java-Client-Verbindung (Enterprise Edition)“ auf Seite 5
`imqConsumerFlowLimit`	„Verbesserte Steuerung des Meldungsflusses beim Java-Client“ auf Seite 7
`imqConsumerFlowThreshold`	„Verbesserte Steuerung des Meldungsflusses beim Java-Client“ auf Seite 7

Tabelle 9 Umbenannte Connection Factory-Attribute

Bisheriger Name	Name in Message Queue 3.5
`imqFlowControlCount`	`imqConnectionFlowCount`
`imqFlowControlIsLimited`	`imqConnectionFlowLimitEnabled`
`imqFlowControlLimit`	`imqConnectionFlowLimit`

Speicherort der öffentlichen Public .jar-Dateien

Auf der Solaris-Plattform wurde der Speicherort der öffentlichen `.jar`-Dateien in Message Queue 3.0.1 verschoben. In Message Queue 3.0 wurde der Speicherort `/usr/share/lib/imq/` verwendet, nun lautet der Pfad: `/usr/share/lib/`. Die symbolischen Verknüpfungen, die in Message Queue 3.0.1 im Verzeichnis `/usr/share/lib/imq/` für die verschobenen JAR-Dateien eingeführt worden waren, wurden entfernt.

Dies gilt für folgende `.jar`-Dateien:

- `jms.jar`
- `imq.jar`
- `imqxm.jar`
- `activation.jar`
- `saaj-api.jar`
- `saaj-impl.jar`
- `mail.jar`
- `commons-logging.jar`
- `jaxm-api.jar`
- `fscontext.jar`

Aktualisierung der Dokumentation

In diesem Abschnitt werden die in Version 3.5 SP1 und in Version 3.5 vorgenommenen Änderungen an der Message Queue-Dokumentation beschrieben.

Änderungen in Version 3.5 SP1

Die folgenden Message Queue 3.5 SP1-Dokumente wurden gegenüber Version 3.5 des Produkts aktualisiert:

Installation Guide

Das Handbuch *Message Queue Installation Guide* berücksichtigt die im Rahmen des Rebrandings vorgenommenen Änderungen und die Informationen zur Plattformunterstützung.

Administration Guide

Das Handbuch *Message Queue Administration Guide* wurde umbenannt (vormals *Message Queue Administrator's Guide*) und berücksichtigt die im Rahmen des Rebrandings vorgenommenen Änderungen. Außerdem enthält dieses Dokument ein erweitertes, aktualisiertes Kapitel zur Leistungsüberwachung.

Java Client Developer's Guide

Das Handbuch *Message Queue Java Client Developer's Guide* berücksichtigt die im Rahmen des Rebrandings vorgenommenen Änderungen.

C Client Developer's Guide

Das Handbuch *Message Queue C Client Developer's Guide* berücksichtigt die im Rahmen des Rebrandings vorgenommenen Änderungen.

Änderungen in Version 3.5

Die folgenden Message Queue 3.5-Dokumente wurden gegenüber Version 3.0.1 des Produkts aktualisiert: Sie finden diese aktualisierten Dokumente auf der Dokumentations-Website für Message Queue3.5 unter `http://docs.sun.com/coll/S1_MessageQueue_35`.

Installation Guide

Das Produkt Message Queue 3.5 beinhaltet einen aktualisierten *Message Queue Installation Guide*. Dazu gehören neue Softwareanforderungen, Änderungen an den Installationsanweisungen für Solaris, ein neues Installationsverfahren für Linux mithilfe von Red Hat Package Manager (RPM) und der installierten Verzeichnisstruktur sowie kleinere Änderungen bei der Installation unter Windows.

Korrektur: Hinsichtlich der Unterstützung der integrierten Persistenz wird in Tabelle 1-2 angegeben, dass PointBase Version 4.5 von Message Queue unterstützt wird. Tatsächlich wird jedoch eine andere Version, PointBase Version 4.8, unterstützt.

Administrator's Guide

Der *Message Queue Administration Guide* berücksichtigt nun die Änderungen in Message Queue 3.5 (siehe „In diesem Abschnitt werden die Änderungen in Message Queue 3.5 SP1 und in der Vorgängerversion, Message Queue 3.5, beschrieben.“ auf Seite 2).

Java Client Developer's Guide

Der *Message Queue Java Client Developer's Guide* enthält die meisten Informationen aus dem früheren Message Queue *Developer's Guide* und berücksichtigt die Änderungen an Message Queue 3.5 (siehe „In diesem Abschnitt werden die Änderungen in Message Queue 3.5 SP1 und in der Vorgängerversion, Message Queue 3.5, beschrieben.“ auf Seite 2).

C Client Developer's Guide

Beim *Message Queue C Client Developer's Guide* handelt es sich um ein neues Handbuch, dass zum Dokumentationssatz für Message Queue hinzugefügt wurde und die Erstellung von Message Queue-C-Client-Anwendungen beschreibt.

Bekannte Probleme und Einschränkungen

In diesem Abschnitt werden bekannte Probleme, Einschränkungen und Fehler in Message Queue 3.5 SP1 und Message Queue 3.5 behandelt. Da es sich bei Version 3.5 SP1 einfach um Message Queue 3.5 in neuem Gewand (Rebranding) handelt, gelten die Informationen in diesem Abschnitt für beide Versionen.

Für eine Liste der aktuellen Fehler, ihrem Status und Umgehungsmöglichkeiten sollten Mitglieder der Java Developer Connection™ die „Bug Parade“ der Java Developer Connection-Website besuchen. Besuchen Sie diese Seite, bevor Sie einen neuen Fehler melden. Auch wenn nicht alle Message Queue-Fehler aufgelistet sind, ist diese Seite ein guter Ausgangspunkt, wenn Sie feststellen möchten, ob ein Problem bekannt gegeben wurde.

Die Adresse der Seite lautet wie folgt:

http://developer.java.sun.com/developer/bugParade

HINWEIS Die Mitgliedschaft bei Java Developer Connection ist kostenlos. Es ist jedoch eine Registrierung erforderlich. Auf der Sun ONE-Webseite „For Developers“ wird beschrieben, wie Sie Mitglied bei der Java Developer Connection werden.

Wenn Sie einen neuen Fehler melden oder eine Funktionsanfrage einreichen möchten, senden Sie eine E-Mail an imq-feedback@sun.com.

Bekannte Probleme

In diesem Abschnitt werden bekannte Probleme in Message Queue 3.5 SP1 behandelt. Einige dieser Probleme wurden bei Message Queue 3.5 eingeführt. In diesem Abschnitt werden die Probleme in Gruppen zusammengefasst, je nachdem, ob sie sowohl auf die Enterprise als auch auf die Platform Edition von Message Queue 3.5 zutreffen oder nur auf die Enterprise Edition.

Enterprise und Platform Edition

- Aufgrund des Produkt-Rebrandings geben APIs, die bisher folgende Zeichenkette zurückgegeben haben:

  „Sun ONE Message Queue, Sun Microsystems, Inc.“,

  nun folgende Zeichenkette zurück:

  „Sun Java(tm) System Message Queue“.

- Aufgrund des Produkt-Rebrandings tritt bei C-Client-Programmen, die mit MQ 3.5 FCS kompiliert wurden und die über `MQGetMetaData()` einen genauen Vergleich des Werts durchführen, der der Eigenschaft `MQ_NAME_PROPERTY` zugeordnet ist, ein Fehler auf, wenn sie die gemeinsam genutzte Bibliothek `mqcrt` von 3.5 SP1 zur Laufzeit verwenden.

- Windows-Plattformen limitieren die Anzahl an Verbindungen mit einem Broker, die gleichzeitig über TCP/IP gestartet werden können, gemäß dem Maximalwert für den Rückstand. Der Rückstand ist ein Puffer für Verbindungen im TCP-Stapel. Die Anzahl an gleichzeitigen TCP-Verbindungsaufrufen kann die Größe dieses Puffers nicht überschreiten. Unter Windows 2000 Professional beispielsweise ist der Rückstand auf fünf beschränkt, unter Windows 2000 Server auf 200.

- Unter Windows XP ist die Anzahl der *eingehenden* Verbindungen beschränkt. Bei Windows XP Professional liegt die Höchstzahl der anderen Computer, die gleichzeitig über das Netzwerk verbunden sein dürfen, bei 10. Dieser Grenzwert umfasst alle Transporte und Ressourcenfreigabeprotokolle zusammen. Bei Windows XP Home Edition liegt die Höchstzahl der anderen Computer, die gleichzeitig über das Netzwerk verbunden sein dürfen, bei 5. Diese Einschränkung betrifft die Anzahl der Clients, die eine Verbindung zu dem unter Windows XP ausgeführten Broker herstellen können.

  Bei allen Sitzungen für Dateien, zum Drucken, für benannte Pipes oder Mail Slot-Sitzungen, die keine Aktivität aufweisen, wird die Verbindung automatisch nach Ablauf der `AutoDisconnect`-Zeit getrennt; der Standardwert für die `AutoDisconnect`-Zeit beträgt 15 Minuten. Nach Trennung der Sitzung wird eine der zehn Verbindungen verfügbar, sodass ein anderer Benutzer eine Verbindung zum Windows XP-System herstellen kann. Daher kann eine Verkürzung der `AutoDisconnect`-Zeit einige der Probleme mit der Begrenzung auf zehn bzw. fünf Verbindungen lösen, wenn das System nicht im großen Stil für Serverzwecke verwendet wird. Weitere Informationen finden Sie unter folgender Adresse:

  `http://support.microsoft.com/default.aspx?scid=kb;DE;314882`

- Die Konfigurationsdatei für eine Broker-Instanz kann nicht bearbeitet werden, ohne dass die Broker-Instanz mindestens ein Mal gestartet wurde. Der Grund hierfür besteht darin, dass die Datei `config.properties` erst erstellt wird, wenn die Broker-Instanz das erste Mal gestartet wird. Wenn Sie einen Broker für die Verwendung austauschbarer Persistenz konfigurieren oder andere Konfigurationseigenschaften festlegen möchten, führen Sie den Broker ein Mal aus (mit dem Namen der Instanz, der zum Erstellen des Brokers verwendet werden soll), um die Datei `config.properties` zu erstellen:

Plattform	Standort
Solaris	`/var/imq/instances/`*Instanzenname*`/props/config.properties`
Linux	`/var/opt/imq/instances/`*Instanzenname*`/props/config.properties`
Windows	`IMQ_VARHOME\instances\`*Instanzenname*`\props\config.properties`

  Nachdem die Datei `config.properties` erstellt wurde, bearbeiten Sie sie, um beliebige Werte für Konfigurationseigenschaften hinzuzufügen, und starten Sie anschließend den Broker neu.

Nur Enterprise Edition

- In dieser Version werden lediglich vollständig verbundene Broker-Cluster unterstützt. Dies bedeutet, dass jeder Broker in einem Cluster direkt mit allen anderen Brokern im Cluster kommunizieren muss. Wenn Sie Broker mithilfe des Befehlszeilenarguments `imqbrokerd -cluster` verbinden, stellen Sie sicher, dass alle Broker im Cluster enthalten sind.
- Wird im Broker-Cluster kein Master-Broker verwendet, werden persistente Informationen, die in einem Broker gespeichert sind, der dem Cluster neu hinzugefügt wird, nicht an die anderen Broker im Cluster weitergegeben.
- Ein Verbindungsdienst, der SSL verwendet, ist derzeit auf die Unterstützung von selbstsignierten Serverzertifikaten eingeschränkt, d. h. auf den beglaubigten Hostmodus.
- Wird ein JMS-Client bei Verwendung des http-Transports plötzlich beendet (z. B. durch den Befehl `Strg-C`), benötigt der Broker etwa eine Minute, bevor die Clientverbindung und alle damit zusammenhängenden Ressourcen freigegeben werden.

  Wird innerhalb dieses Zeitraums eine andere Instanz des Clients gestartet, die versucht, dieselbe Client-ID, Warteschlange oder dasselbe dauerhafte Abonnement zu verwenden, wird möglicherweise der Ausnahmefehler „Client-ID wird bereits verwendet“ gemeldet. Dies stellt jedoch kein Problem dar, es handelt sich lediglich um eine Nebenwirkung des vorangehend beschriebenen Beendigungsvorgangs. Wenn der Client nach etwa einer Minute gestartet wird, sollte kein Fehler gemeldet werden.

Bekannte Fehler

Tabelle 10 beschreibt die in Message Queue 3.5 SP1 noch ungelösten Fehler

Tabelle 10 Bekannte Fehler in Message Queue 3.5

Fehlernummer	Details
4683029	Die Option `-javahome` in allen solaris/win-Skripten funktioniert nicht, wenn der Wert ein Leerzeichen enthält. Die Option `-javahome` wird von den Message Queue-Befehlen und -Programmen verwendet, um eine alternative Java 2-kompatible Runtime anzugeben. Der Pfad zur alternativen Java-Runtime darf jedoch keine Leerzeichen enthalten. Beispiele für Pfade, die Leerzeichen enthalten: Windows: `C:\jdk 1.4` (Unter Windows kann ein Pfad Leerzeichen enthalten, wenn der gesamte Pfad in Anführungszeichen gesetzt wird, beispielsweise `"C:\jdk 1.4".`) Solaris: `/work/java 1.4` **Umgehung**: Installieren Sie die Java-Runtime an einem Speicherort oder unter einem Pfad, der keine Leerzeichen enthält.
4939923	Der Broker gibt möglicherweise die Ausnahme `NullPointerException` aus, wenn ein gemeinsamer Thread-Pool verwendet wird und der Arbeitsspeicher der JVM des Brokers knapp wird. **Umgehung**: Keine. Dieser Fehler wird in J2SE 1.4.2_03 behoben.
4941058	Ziele mit aktivierter Fluss-Steuerung erreichen möglicherweise nicht die Obergrenze. Es gibt Situationen, in denen Produzenten keine Meldungen mehr an ein Ziel senden, ohne dass das Ziel die konfigurierte Obergrenze erreicht hat. **Umgehung**: Keine
4941066	Bei den Zielen können die zugewiesenen Byte-Grenzen etwas überschritten werden. **Umgehung**: Keine
4941127	Das Ziel wird nicht vollständig geladen, wenn eine Meldung die individuelle Obergrenze für die Meldungsgröße überschreitet. Wenn die Obergrenze für die Größe der für ein Ziel zulässigen Meldungen geändert wird, nachdem eine größere Meldung gespeichert wurde, wird das Ziel nicht ordnungsgemäß geladen. **Umgehung**: Erhöhen Sie die Obergrenze für die Meldungsgröße, bis die große Meldung konsumiert wurde, und senken Sie anschließend den Grenzwert für die Meldungen wieder. Möglicherweise sollte die Produktion für das Ziel während dieser Zeit unterbrochen werden, um zu verhindern, dass andere große Meldungen angenommen werden.

Tabelle 10 Bekannte Fehler in Message Queue 3.5 *(Fortsetzung)*

Fehlernummer	Details
4946531	Beim Produzieren von Meldungen kann in seltenen Fällen eine harmlose `NullPointerException` auftreten. **Umgehung**: Keine – die Null-Zeiger-Ausnahme kann problemlos ignoriert werden.
4949398	`imqcmd query dst` meldet falsche Werte für `Anzahl Meldungen` und `Gesamtgröße Meldungen`, während das zugehörige Ziel geladen wird. Die gemeldeten Werte sind vor und nach dem Laden des Ziels korrekt. **Umgehung**: Das Problem tritt nur auf, während das Ziel geladen wird. Sobald der Ladevorgang abgeschlossen ist, sind die zurückgegebenen Werte korrekt.
4950166	Zufällige Fehler beim Broker bei der Ausführung auf den Systemen jdk1.4.2_02 und x86. Weitere Informationen finden Sie unter dem J2SE-Fehler 4947404. **Umgehung**: Starten Sie den Broker mit `-XX:UseSSE=0`, Beispiel: `imqbrokerd -tty -vmargs -XX:UseSSE=0`

4950601	`imqcmd metrics dst` löst einen zu druckenden internen Broker-Fehler aus, wenn der JDBC-Persistenzspeicher verwendet wird.
	
	Die Metrikinformationen zur Speicherplatznutzung gelten nur für den Dateispeicher. Beim Abfragen der Metrikinformationen versucht der Broker jedoch unabhängig vom Speichertyp die Informationen zur Speicherplatznutzung abzurufen. Wenn statt des Dateispeichers eine Datenbank verwendet wird, wird folgende Fehlermeldung vom Broker gedruckt:
	
	`06/Nov/2003:22:57:36 PST] ERROR [B3100]: Unexpected Broker Internal`
	
	`Error : [unable to disk usage for destinationT:topic1] :`
	
	`com.sun.messaging.jmq.jmsserver.util.BrokerException:`
	`Der Vorgang gilt nicht für integrierten Persistenzspeicher.`
	
	Umgehung: Keine

4951010	In einem Broker-Cluster reiht ein Broker Meldungen an eine entfernte Verbindung, die möglicherweise nicht gestartet wurde, in eine Warteschlange ein. **Umgehung**: Die Meldungen werden vom Konsumenten empfangen, wenn die Verbindung gestartet ist. Die Meldungen werden an einen anderen Konsumenten umgeleitet, wenn die Verbindung des Konsumenten geschlossen wird.
4953348	HTTPS `createQueueConnection` löst gelegentlich einen Ausnahmefehler unter Windows 2000 aus. **Umgehung**: Versuchen Sie noch einmal, die Verbindung herzustellen.
4953354	Zugriff auf den Broker ist nicht mehr möglich, wenn der Persistenzspeicher zu viele Ziele öffnet. **Umgehung**: Diese Bedingung wird dadurch verursacht, dass der Broker die Beschreibungsgrenze für die Dateiöffnungen im System erreicht. Verwenden Sie unter Solaris und Linux den Befehl `ulimit`, um die Dateibeschreibungsbegrenzung zu erhöhen.

Tabelle 10 Bekannte Fehler in Message Queue 3.5 *(Fortsetzung)*

Fehlernummer	Details
4954974	Bei Verwendung der CD-ROM startet die Installation unter Windows XP nicht automatisch. **Umgehung**: Doppelklicken Sie im Windows-Explorer auf den Windows-Ordner auf der CD und anschließend auf die Datei `imq3_5-ent-win.exe`, um das Installationsprogramm zu starten.
4983525	Die Erstellung eines Meldungsproduzenten für ein automatisch erstelltes Ziel schlägt unter Linux Red Hat Advanced Server 3.0 möglicherweise fehl. **Umgehung**: Versuchen Sie, den Produzenten neu zu erstellen. Beim zweiten Mal sollte es funktionieren. Alternativ können Sie ein vom Administrator erstelltes Ziel verwenden.

4986318	Der Client gibt möglicherweise unerwartet eine ACKNOWLEDGE_REPLY-Meldung aus:
	
	`******** Packet:`
	`ACKNOWLEDGE_REPLY(25):26-192.18.86.227-42976-1075458056557`
	
	`Magic/Version: 469754818/301Size: 97 Type: ACKNOWLEDGE_REPLY(25)`
	
	`Expiration: 0 Timestamp: 1075458056557`
	
	`Source IP: 192.18.86.227 Source Port: 42976Sequence: 26`
	
	Umgehung: Keine. Im Broker liegt eine seltene Zeitbedingung vor, der diese Fehlermeldung beim Client verursacht. Der Fehler kann ignoriert werden. Es gehen keine Meldungen verloren.

4991257	Wenn große persistente Meldungen an dauerhafte Abonnenten in einem Broker-Cluster gesendet werden, bei dem der Persistenzspeicher JDBC-basiert ist, kann der Broker abstürzen und/oder Fehlermeldungen ausgeben. **Umgehung**: Erhöhen Sie den Zeitüberschreitungswert für das Sperrprotokoll des Brokers. Verwenden Sie dazu folgende Broker-Eigenschaft: `imq.cluster.timeout=<timeout-in-seconds>` Der Standardwert ist 60. Wenn das Persistentmachen großer Meldungen sehr viel Zeit in Anspruch nimmt, müssen Sie möglicherweise die Persistenzspeicherdatenbank anpassen oder auf einen anderen Persistenzspeicher wechseln.

5006686	Das ARGS-Beispiel in imqbrokerd.conf ist nicht korrekt.
	
	Umgehung: Die Werte sollten nicht in Anführungsstrichen stehen, wie im Beispiel angegeben.
	
	`ARGS="-name newbroker -port 8888"`
	
	Die Werte sollten wie folgt aussehen:
	
	`ARGS=-name newbroker -port 8888`

Dateien für Neuverteilung

Sun Java System Message Queue 3.5 SP1 enthält folgende Dateien, die Sie verwenden und ohne Einschränkungen im Binärformat weiterverbreiten können:

```
jms.jar
imq.jar
imqxm.jar
fscontext.jar
providerutil.jar
jndi.jar
ldap.jar
ldapbpjar
jaas.jar
jsse.jar
jnet.jar
jcert.jar
```

Außerdem können Sie die Dateien `LICENSE` und `COPYRIGHT` weitergeben.

Problemmeldungen und Feedback

Schreiben Sie eine E-Mail an imq-feedback@sun.com.

Wenn Sie einen Unterstützungsvertrag haben und Probleme mit Message Queue auftreten, wenden Sie sich an den Kundendienst. Folgende Möglichkeiten stehen zur Auswahl:

- Sun-Softwaresupport unter:
  http://www.sun.com/service/sunone/software

  Auf dieser Website finden Sie Links zur Knowledge Base, zum Online Support Center, zum ProductTracker wie auch zu Wartungsprogrammen und Kontaktinformationen für die Kundenunterstützung.
- Die auf Ihrem Wartungsvertrag angegebene Telefonnummer.

Damit wir Sie optimal beraten können, halten Sie bitte die folgenden Informationen bereit, wenn Sie sich an die Kundenunterstützung wenden:

- Beschreibung des Problems, einschließlich der Situation, in der das Problem auftrat, sowie seine Auswirkungen auf Ihre Arbeit.
- Rechnertyp, Betriebssystem- und Produktversion, einschließlich sämtlicher Patches und anderer Software, die mit dem Problem in Zusammenhang stehen könnten.
- Zur Nachvollziehung des Problems eine ausführliche Beschreibung der einzelnen Schritte und Vorgehensweisen, die zu dem Problem geführt haben.
- Sämtliche Fehlerprotokolle oder Kernspeicherauszüge.

Kommentare sind willkommen

Sun möchte seine Dokumentation laufend verbessern. Ihre Kommentare und Vorschläge sind daher immer willkommen. Verwenden Sie das webbasierte Formular, um uns Ihr Feedback mitzuteilen:

`http://www.sun.com/hwdocs/feedback/`

Tragen Sie den vollständigen Titel der Dokumentation und die vollständige Teilenummer in die entsprechenden Felder ein. Sie finden die Teilenummer auf der Titelseite des Buchs oder oben auf dem Dokument. Dabei handelt es sich in der Regel um eine sieben- oder neunstellige Nummer. Die Teilenummer dieser *Message Queue 3.5 SP1 Versionshinweise* lautet 817-7191-10.

Weitere Informationen über Sun

Zusätzlich zur Message Queue-Dokumentation finden Sie an den unten angegebenen Orten zusätzliche Informationen.

Diskussionsforen

Sun Java System Softwareforum

Unter der nachfolgenden Adresse finden Sie ein Sun Java System Message Queue-Forum:

http://softwareforum.sun.com/NASApp/jive/forum.jsp?forum=24

Wir würden uns über Ihre Beteiligung freuen.

Java Technology Forum

Unter dem Java Technology Forum finden Sie möglicherweise ein für Sie interessantes JMS-Forum.

http://forum.java.sun.com

SunSolve Knowledge Base

Informationen zu Sun Java System Message Queue finden Sie online in der SunSolve Knowledge Base unter:

http://sunsolve.Sun.COM/pub-cgi/search.pl?mode=advanced

Wählen Sie die Option „All Free Collections“ und suchen Sie anschließend nach „Message Queue“.

Informationen zu Sun Java System

Nützliche Informationen über Sun Java System finden Sie unter den folgenden Internet-Adressen:

- Message Queue-Produktseite
  http://wwws.sun.com/software/products/message_queue/index.html
- Dokumentation zu Message Queue
  http://docs.sun.com/coll/MessageQueue_35_SP1
- Sun-Dokumentationen
  http://docs.sun.com/
- Sun Java System Softwareprodukte und Dienste
  http://www.sun.com/software
- Sun-Softwaresupport
  http://www.sun.com/service/sunone/software

- Sun-Support und Sun-Knowledge Base
  http://sunsolve.sun.com
- Sun-Support und -Schulungen
  http://www.sun.com/supportraining
- Sun-Informationen für Entwickler
  http://developers.sun.com/
- Sun-Supportdienste für Entwickler
  http://www.sun.com/developers/support
- Sun-Softwaredatenblätter
  http://www.sun.com/software