المهلات وإعادة المحاولات والتراجع مع تقلقل الشبكة

مارك بروكر

حدوث حالات الفشل

كلما استدعت خدمة أو نظام ما خدمة أو نظام آخر، فقد يحدث الفشل. يمكن أن تحدث حالات الفشل هذه بسبب مجموعة متنوعة من العوامل، ويشمل ذلك الخوادم أو الشبكات أو موازنات الأحمال أو البرامج أو أنظمة التشغيل أو حتى الأخطاء التي تقع من مشغلي النظام. ولذا، فإننا نصمم أنظمتنا لتقليل احتمالية الفشل، ولكن من المستحيل إنشاء أنظمة لا تفشل أبدًا. وفي Amazon نصمم أنظمتنا للتعامل مع احتماليات الفشل وتقليلها، وتجنب تضخيم حالات الفشل الصغيرة وتطورها إلى انقطاع تام. لبناء أنظمة مرنة، نستخدم ثلاث أدوات أساسية: المهلات وإعادة المحاولة والتراجع.

تظهر العديد من أنواع حالات الفشل مثل الطلبات التي تستغرق وقتًا أطول من المعتاد، وربما لا تكتمل أبدًا. عندما ينتظر العميل فترة أطول من المعتاد لإكمال الطلب، فإنه يحتفظ أيضًا بالموارد التي كان يستخدمها لهذا الطلب لفترة أطول. عندما يحتفظ عدد من الطلبات بالموارد لفترة طويلة، يمكن أن تنفد هذه الموارد من على الخادم. يمكن أن تشكل هذه الموارد الذاكرة أو مؤشرات الترابط أو الاتصالات أو المنافذ المؤقتة أو أي شيء آخر محدود. ولتجنب هذا الموقف، يقوم العملاء بتحديد *مهلات*. والمقصود بالمهلات هو الحد الأقصى للوقت الذي ينتظره العميل لإكمال الطلب.

وفي كثير من الأحيان، تؤدي محاولة نفس الطلب مرة أخرى إلى نجاح الطلب. يحدث هذا لأن أنواع الأنظمة التي نبنيها لا تفشل في كثير من الأحيان كوحدة واحدة، ولكنها قد تعاني من فشل جزئي أو عابر. ويكون الفشل جزئيًا عندما تنجح نسبة من الطلبات. أما الفشل المؤقت فهو أن يفشل الطلب لفترة قصيرة من الزمن. *إعادة المحاولة* تسمح للعملاء بالاستمرار على الرغم من وجود حالات الفشل الجزئية العشوائية والإخفاقات المؤقتة قصيرة الأجل بإرسال الطلب نفسه مرة أخرى.

وليس من الآمن دائمًا إعادة المحاولة. يمكن أن تؤدي إعادة المحاولة إلى زيادة الحمل على النظام الذي يتم الاتصال به، إذا كان النظام قد فشل بالفعل بسبب اقترابه من الحمل الزائد. لتجنب هذه المشكلة، ننصح عملائنا باستخدام *عملية التراجع*. فهذا يزيد من الوقت بين المحاولات اللاحقة، والتي تحافظ على توازن الحمل على العمليات الخلفية. المشكلة الأخرى في إعادة المحاولة هي أن بعض الاستدعاءات البعيدة لها آثار جانبية. لا تعني المهلة أو الفشل بالضرورة أن الآثار الجانبية لم تحدث. إذا كان القيام بالآثار الجانبية عدة مرات أمرًا غير مرغوب فيه، فإن أفضل الممارسات هي تصميم واجهات برمجة التطبيقات لتكون *متساوية القوى*، مما يعني أنه يمكن إعادة المحاولة بأمان.

وأخيرًا، لا تصل حركة المرور إلى خدمات Amazon بمعدل ثابت، بل يكون معدل وصول الطلبات في كثير من الأحيان بتدفقات كبيرة. وقد يكون سبب هذه التدفقات الكبيرة سلوك العميل واستعادة الفشل وحتى شيء بسيط مثل مهمة cron دورية. إذا حدثت أخطاء بسبب الحمل، فقد تكون عمليات إعادة المحاولة غير فعالة في حالة إعادة محاولة جميع العملاء في نفس الوقت. لتجنب هذه المشكلة، نستخدم *تقلقل الشبكة*. هذا هو وقت عشوائي قبل تقديم طلب أو إعادة محاولة للمساعدة في منع التدفقات الكبيرة عن طريق نشر معدل الوصول.

تمت مناقشة كل حل من هذه الحلول في الأقسام التالية.

المهلات

من ضمن أفضل الممارسات في Amazon تعيين مهلة على أي استدعاء عن بعد، وعمومًا على أي استدعاء عبر العمليات حتى في نفس المربع. يشمل هذا كلاً من مهلة الاتصال ومهلة الطلب. يوفر العديد من العملاء القياسيين إمكانات قوية مدمجة للمهلة.

وعادةً ما تكون المشكلة الأكثر صعوبة هي اختيار قيمة مهلة لتحديدها. وسيقلل تحديد مهلة عالية للغاية من فائدتها، لأن الموارد لا تزال تُستهلك أثناء انتظار العميل المهلة. وتحديد مهلة منخفضة للغاية ينطوي على خطرين:

- زيادة حركة المرور على الواجهة الخلفية وزيادة زمن الاستجابة بسبب إعادة محاولة العديد من الطلبات.
- زيادة زمن الاستجابة للواجهة الخلفية البسيطة مما يؤدي إلى انقطاع تام، لأن كل الطلبات تبدأ في إعادة المحاولة.

ومن الممارسات الجيدة لاختيار مهلة للاستدعاءات داخل منطقة AWS أن تبدأ بمقاييس زمن الاستجابة لخدمة نقل البيانات من الخادم. لذا في Amazon، عندما نجعل خدمة ما تستدعي خدمة أخرى، نختار معدل مقبول من المهلات الزائفة (مثل 0.1%). بعد ذلك، ننظر إلى النسبة المئوية المقابلة لزمن الاستجابة على خدمة انتقال البيانات من الخادم (p99.9 في هذا المثال). يعمل هذا النهج بشكل جيد في معظم الحالات، ولكن هناك بعض الأخطاء التي يمكن وصفها على النحو التالي:

- لا يعمل هذا النهج في الحالات التي يكون فيها للعملاء زمن استجابة كبير للشبكة عبر الإنترنت على سبيل المثال. في هذه الحالات، نضع في الاعتبار زمن استجابة الشبكة في أسوأ الحالات، مع الأخذ في الاعتبار أن العملاء قد يكونوا من أنحاء العالم.

- لا يعمل هذا النهج أيضًا مع الخدمات ذات الحدود الضيقة لزمن الاستجابة، حيث P99.9 قريبة من p50. في هذه الحالات، تساعدنا إضافة بعض الحشوات في تجنب زيادات زمن الاستجابة الصغيرة التي تسبب أعدادًا كبيرة من المهلات.

- لقد واجهنا مشكلة شائعة عند تنفيذ المهلات. يعتبر Linux's SO_RCVTIMEO قوي، لكن به بعض العيوب التي تجعله غير مناسب كمهلة مأخذ توصيل نهاية إلى نهاية. تعرض بعض اللغات، مثل Java، عنصر التحكم هذا مباشرةً. توفر لغات أخرى، مثل Go، آليات مهلة أكثر قوة.

- هناك أيضًا تطبيقات لا تغطي المهلة فيها جميع الاستدعاءات البعيدة، مثل مصافحات DNS أو TLS. نفضل بشكل عام استخدام المهلات المضمنة في العملاء الذين تم اختبارهم جيدًا. إذا قمنا بتنفيذ مهلاتنا الخاصة، فإننا نولي اهتمامًا دقيقًا بالمعنى الدقيق لخيارات مأخذ المهلة، وما هو العمل الذي يتم القيام به.

شاهدنا في نظام واحد عملت عليه في Amazon عددًا صغيرًا من المهلات التي تتحدث إلى التبعية فور نشرها. تم تعيين مهلة منخفضة للغاية، تصل إلى حوالي 20 مللي ثانية. بعيدًا عن عمليات النشر، لم نر حدوث مهلات منتظمة حتى مع انخفاض قيمة المهلة هذه. وبالبحث، وجدت أن المؤقت يتضمن إنشاء اتصال آمن جديد، تمت إعادة استخدامه بناءً على الطلبات اللاحقة. ونظرًا لأن إنشاء الاتصال استغرق وقتًا أطول من 20 مللي ثانية، فقد رأينا عددًا قليلاً من الطلبات تنقضي مهلتها عند وصول خادم جديد إلى الخدمة بعد عمليات النشر. في بعض الحالات، تمت إعادة محاولة الطلبات ونجاحها. لقد عملنا في البداية على حل هذه المشكلة عن طريق زيادة قيمة المهلة في حالة إنشاء اتصال. وقمنا بتحسين النظام في وقت لاحق من خلال إنشاء هذه الاتصالات عند بدء العملية، ولكن قبل تلقي حركة المرور. وقد مكننا ذلك من التغلب على مشكلة المهلة تماما.

إعادة المحاولة والتراجع

إعادة المحاولة «طريقة أنانية». بمعنى آخر، عندما يعيد عميل ما المحاولة مرة أخرى، فإنه يمضي وقتًا أطول في الخادم للحصول على فرصة أكبر للنجاح. وعندما تكون حالات الفشل نادرة أو عابرة، فهذه ليست مشكلة. وذلك لأن العدد الإجمالي لطلبات إعادة المحاولة صغير، وأن المقايضة بزيادة التوافر الظاهر تعمل بشكل جيد. عندما تحدث حالات الفشل بسبب الحمل الزائد، فإن إعادة المحاولات التي تزيد الحمل قد تزيد الأمور سوءًا. بل قد تتسبب في تأخير الاسترداد بالإبقاء على ارتفاع الحمل لفترة طويلة بعد حل المشكلة الأصلية. تشبه إعادة المحاولات دواءً قويًا - مفيد في الجرعة المناسبة، ولكن يمكن أن يسبب أضرارًا كبيرة عند استخدامه كثيرًا. لسوء الحظ، في الأنظمة الموزعة، لا توجد طريقة للتنسيق بين جميع العملاء لتحقيق العدد الصحيح من المحاولات.

الحل المفضل الذي نستخدمه في Amazon هو التراجع. فبدلاً من إعادة المحاولة فورًا وبقوة، ينتظر العميل بعض الوقت بين المحاولات. النمط الأكثر شيوعًا هو *التراجع المطرد*، حيث تتم زيادة وقت الانتظار بشكل كبير بعد كل محاولة. يمكن أن يؤدي التراجع المطرد إلى أوقات تراجع طويلة جدًا، لأن الدوال الأسية تنمو بسرعة. لتجنب إعادة المحاولة لفترة طويلة جدًا، عادةً تجعل عمليات التنفيذ التراجع بأعلى حد أقصى للقيمة القصوى. وهذا ما يسمى، كما هو متوقع، *التراجع المطرد المحدد*. ومع ذلك، فإن هذا يثير مشكلة أخرى. الآن جميع العملاء يحاولون باستمرار بمعدل محدد. في جميع الحالات تقريبًا، يكون حلنا هو الحد من عدد مرات إعادة محاولة العميل، والتعامل مع الفشل الناتج في وقت سابق في البنية الموجهة إلى الخدمات. في معظم الحالات، سيتخلى العميل عن استدعاء الخدمة على أي حال، لأن لديه مهلاته الخاصة.

هناك مشاكل أخرى مع إعادة المحاولة، نتناولها فيما يلي:

- غالباً ما تكون للأنظمة الموزعة طبقات متعددة. ضع في اعتبارك نظامًا حيث يتسبب استدعاء العميل للخدمة في حزمة عميقة مكونة من خمس استدعاءات للخدمة. وتنتهي باستعلام قاعدة بيانات، وثلاث محاولات لإعادة المحاولة في كل طبقة. ماذا يحدث عندما تبدأ قاعدة البيانات في فشل الاستعلامات في ظل التحميل؟ في حالة إعادة محاولة كل طبقة بشكل مستقل، فسيزيد الحمل على قاعدة البيانات بمقدار 243 ضعفًا، مما يجعل من غير المحتمل استرداده على الإطلاق. وذلك لأن المحاولات في كل طبقة تتكاثر - أولاً ثلاث محاولات، ثم تسع محاولات، وهكذا. وعلى العكس من ذلك، فإن إعادة المحاولة في أعلى طبقة من الحزمة قد تضيع العمل من الاستدعاءات السابقة، مما يقلل الكفاءة. وبشكل عام، فإن أفضل الممارسات لدينا هي إعادة المحاولة عند نقطة واحدة في الحزمة بالنسبة لعمليات control-plane وdata-plane منخفضة التكلفة.

- الحمل. حتى مع وجود طبقة واحدة من المحاولات، لا تزال حركة المرور تزيد بشكل كبير عند بدء الأخطاء. يتم تشجيع استخدام *قواطع الدائرة* على نطاق واسع لحل هذه المشكلة، حيث يتم إيقاف استدعاءات خدمة انتقال البيانات من الخادم بالكامل عند تجاوز حد الخطأ. لسوء الحظ، تعمل قواطع الدائرة على تقديم سلوك مشروط في الأنظمة التي يصعب اختبارها، ويمكنها توفير وقت إضافي كبير للاسترداد. لقد وجدنا أنه يمكننا تخفيف هذا الخطر بالحد من إعادة المحاولة محليًا باستخدام حاوية الرمز المميز. يتيح ذلك لجميع الاستدعاءات إعادة المحاولة طالما كانت هناك رموز مميزة، ثم إعادة المحاولة بمعدل ثابت عندما يتم استنفاد الرموز المميزة. أضافت AWS هذا السلوك إلى AWS SDK في عام 2016. لذا، فإن العملاء الذين يستخدمون SDK لديهم سلوك الحظر المدمج هذا.

- تحديد وقت إعادة المحاولة. وتتلخص وجهة نظرنا بشكل عام في أن واجهات برمجة التطبيقات ذات الآثار الجانبية ليست آمنة لإعادة المحاولة إلا إذا كانت تقدم خاصية تساوي القوى. ويضمن هذا أن الآثار الجانبية لن تحدث إلا مرة واحدة بغض النظر عن عدد مرات إعادة المحاولة. عادة قد تكون واجهات برمجة التطبيقات للقراءة فقط غير نشطة، بينما لا تكون واجهات برمجة التطبيقات الخاصة بإنشاء الموارد كذلك. توفر بعض واجهات برمجة التطبيقات، مثل واجهة برمجة تطبيقات تشغيل Amazon Elastic Compute Cloud (Amazon EC2) ، آليات واضحة تستند إلى الرمز المميز لتوفير تساوي القوى وجعلها آمنة لإعادة المحاولة. ينبغي تصميم واجهة برمجة تطبيقات جيدة والعناية عند تنفيذ العملاء لمنع الآثار الجانبية المكررة.

- معرفة أي محاولات فشل تستحق إعادة المحاولة. يقدم HTTP تمييزًا واضحًا بين أخطاء *العميل* و*الخادم*. يشير ذلك إلى أن أخطاء العميل يجب ألا تتم إعادة المحاولة فيها بنفس الطلب لأنها لن تنجح لاحقًا، بينما قد تنجح أخطاء الخادم في المحاولات اللاحقة. ولسوء الحظ، فإن التناسق الشامل في الأنظمة يؤدي إلى تمويه هذا الخط. قد يتغير خطأ العميل في إحدى اللحظات إلى نجاح في اللحظة التالية مع انتشار الحالة.

على الرغم من هذه المخاطر والتحديات، تُعد عمليات إعادة المحاولة آلية قوية لتوفير إمكانية توفر عالية في مواجهة الأخطاء المؤقتة والعشوائية. يلزم استخدام التقدير للعثور على المفاضلة الصحيحة لكل خدمة. في تجربتنا، للبدء من نقطة جيدة ينبغي أن نتذكر أن إعادة المحاولات طريقة أنانية. تُعد عمليات إعادة المحاولة طريقة يستخدمها العملاء لتأكيد أهمية طلباتهم والمطالبة بأن تنفق الخدمة المزيد من مواردها للتعامل مع طلباتهم. فإذا كان العميل أنانيًا جدًا، فيمكنه أن يخلق مشاكل واسعة النطاق.

تقلقل الشبكة

عندما تنتج حالات الفشل عن الحمل الزائد أو التنافس، فإن التراجع في كثير من الأحيان لا يساعد بالقدر الذي ينبغي، وهذا بسبب الارتباط. فإذا تراجعت جميع الاستدعاءات الفاشلة في نفس الوقت، فإنها تتسبب في حدوث تنافس أو زيادة حمل مرة أخرى عند إعادة المحاولة. والحل الأمثل هو تقلقل الشبكة. يضيف تقلقل الشبكة بعض العشوائية إلى عملية التراجع لنشر إعادة المحاولة في جميع أنحاء الشبكة في الوقت المناسب. لمزيد من المعلومات حول مقدار تقلقل الشبكة المراد إضافته وأفضل الطرق لإضافته، راجع التراجع الأسي وتقلقل الشبكة.

تقلقل الشبكة ليس الغرض منه إعادة المحاولة فقط. لقد علمتنا الخبرة التشغيلية أن حركة المرور إلى خدماتنا، بما في ذلك both control-planes وdata-planes، تميل إلى حد الذروة كثيرًا. ويمكن أن تكون ساعات الذروة لحركة المرور قصيرة جدًا، وغالبًا ما تكون مخفية بواسطة مقاييس مجمعة. عند إنشاء أنظمة، نفكر في إضافة بعض تقلقل الشبكة إلى جميع أجهزة التوقيت والمهام الدورية وغيرها من الأعمال المتأخرة. ويساعد ذلك في انتشار ساعات ذروة العمل، ويسهل على الخدمات المتلقية للمعلومات توسيع نطاق حمل العمل.

عند إضافة تقلقل الشبكة إلى العمل المجدول، لا نختاره على كل مضيف بشكل عشوائي، بل نستخدم طريقة متسقة تنتج نفس الرقم في كل مرة على نفس المضيف. وبهذه الطريقة، إذا كانت هناك خدمة محمّلة فوق طاقتها، أو حالة سباق، فإنها تحدث بنفس الطريقة في نمط ما. ونحن جيدون في تحديد الأنماط وأكثر قدرة على تحديد السبب الأساسي، ونضمن باستخدام طريقة عشوائية أنه إذا تم تطويق أحد الموارد، فإن هذا لن يحدث إلا بشكل جيد وبصورة عشوائية، مما يجعل استكشاف الأخطاء وإصلاحها أكثر صعوبة بكثير.

على الأنظمة التي عملت عليها، مثل Amazon Elastic Block Store (Amazon EBS) و AWS Lambda، وجدنا أن العملاء يرسلون طلبات بشكل متكرر على فترات منتظمة، مثل مرة واحدة في الدقيقة. ومع ذلك، عندما يكون لدى العميل خوادم متعددة تتصرف بالطريقة نفسها، فيمكنهم اصطفاف طلباتهم وتشغيلها في نفس الوقت. قد يكون هذا في الثواني القليلة الأولى من الدقيقة، أو الثواني القليلة الأولى بعد منتصف الليل للمهام اليومية. ومن خلال الاهتمام بالحمل في الثانية الواحدة، والعمل مع العملاء للتخلص من أحمال العمل الدورية الخاصة بهم، فقد أنجزنا نفس القدر من العمل مع سعة خادم أقل.

وعلى الرغم من أننا لا نتحكم في ساعات الذروة في حركة العملاء إلا بقدر ضئيل، إلا انه من الجيد إضافة تقلقل الشبكة حيث لا يؤثر على تجربة العميل حتى بالنسبة للمهام التي يحفزها العميل.

الخاتمة

في الأنظمة الموزعة، يكون الفشل المؤقت أو زمن الاستجابة في التفاعلات البعيدة أمرًا لا مفر منه. تمنع المهلات الأنظمة من التعليق لفترة غير معقولة، ويمكن أن تؤدي إعادة المحاولات إلى إخفاء حالات الفشل هذه، كما أن التراجع وتقلقل الشبكة يمكن أن يحسن الاستخدام ويقلل من الاختناق على الأنظمة.

في Amazon، تعلمنا أنه من المهم توخي الحذر بشأن إعادة المحاولات، لأنها يمكن أن تؤدي إلى تضخيم الحمل على نظام تابع. فإذا انتهت مهلة الاستدعاءات على نظام ما، وكان ذلك النظام محملاً بشكل زائد، فقد تؤدي إعادة المحاولات إلى زيادة الحمل بشكل سيء وليس تحسينه. ونحن نتجنب هذا التضخيم من خلال إعادة المحاولة فقط عند ملاحظة أن التبعية صحية. ونتوقف عن إعادة المحاولة عندما لا تساعد في تحسين التوافر.