Membangun dasbor untuk visibilitas operasional

John O'Shea

Membangun dasbor untuk visibilitas operasional

Kita semua menjalankan aplikasi di laptop, tablet, dan ponsel cerdas. Kita dengan mudah melihat apakah perangkat sedang aktif dan apakah koneksi jaringan Wi-Fi sedang online. Kita tahu bahwa layar akan menampilkan notifikasi penting, seperti peringatan ruang disk yang hampir penuh. Faktanya, kecepatan umum dan responsifitas antarmuka pengguna (UI) bisa menjadi indikator yang baik untuk mengetahui apakah sumber daya perangkat, seperti memori atau CPU, cukup untuk menjalankan ragam aplikasi.

Siapa pun yang memberikan dukungan teknis jarak jauh untuk perangkat keluarganya bisa membuktikan bahwa mendeteksi dan mendiagnosis masalah akan sedikit lebih sulit jika tidak melihat dan berinteraksi langsung dengan perangkat. Jadi, untuk urusan menjalankan layanan berbasis cloud, tantangan kita sama: Bagaimana memantau layanan jarak jauh ini, dan bagaimana kita tahu jika pelanggan merasa senang?

Untuk mengamati layanan host tunggal, kita bisa masuk ke host itu, menjalankan berbagai alat pemantauan waktu proses, dan memeriksa log untuk menentukan akar masalah yang terjadi pada host tersebut. Namun, solusi host tunggal hanya cocok untuk layanan non-kritis yang paling sederhana. Di sisi lainnya adalah layanan mikro terdistribusi dan multi-tingkat yang berjalan di ratusan atau ribuan server, kontainer, atau lingkungan tanpa server.

Bagaimana Amazon melihat perilaku sebenarnya dari semua layanan berbasis cloud yang berjalan di beberapa Availability Zone di berbagai Wilayah di seluruh dunia? Pemantauan otomatis, alur kerja remediasi otomatis (misalnya, pengalihan lalu lintas), dan sistem penerapan otomatis sangat vital untuk mendeteksi dan menyelesaikan sebagian besar masalah pada skala ini. Namun, karena satu dan lain hal, kita masih harus bisa sewaktu-waktu melihat apa saja kegunaan layanan, alur kerja, dan penerapan ini.

Dashboarding di Amazon

Kita menggunakan dasbor sebagai salah satu mekanisme untuk mengatasi tantangan untuk tetap memantau aktivitas di layanan cloud. *Dasbor* adalah tampilan di hadapan manusia yang menampilkan sistem kita dan berisi ringkasan singkat tentang bagaimana sistem berperilaku dengan menampilkan metrik deret waktu, log, jejak, dan data alarm.

Di Amazon, pembuatan, penggunaan, dan pemeliharaan berkelanjutan atas dasbor ini kami sebut *dashboarding.* Dashboarding berkembang menjadi aktivitas utama karena sangat penting bagi keberhasilan layanan kita seperti pengiriman perangkat lunak sehari-hari dan aktivitas operasional lain, seperti merancang, membuat kode, membangun, menguji, menerapkan, dan meningkatkan layanan.

Tentu saja, sebaiknya operator kita tidak setiap saat memantau dasbor. Umumnya, tidak seorangpun yang melihat dasbor ini. Kami justru mendapati bahwa setiap proses operasional yang memerlukan tinjauan manual atas dasbor akan gagal karena kesalahan manusia, terlepas seberapa sering dasbor ditinjau. Untuk mengatasi risiko ini, kami membuat alarm otomatis yang secara konstan mengevaluasi data pemantauan terpenting yang dimunculkan oleh sistem kami. Biasanya, ini merupakan metrik yang menunjukkan bahwa sistem mendekati batas tertentu (deteksi proaktif, sebelum dampak) atau bahwa sistem sudah rusak secara tak terduga (deteksi reaktif, setelah dampak).

Alarm ini dapat menjalankan alur kerja remediasi otomatis dan dapat memberi tahu operator kami bahwa ada masalah. Pemberitahuan tersebut mengarahkan operator ke dasbor dan runbook yang tepat yang harus mereka gunakan. Saat saya sedang menelepon dan alarm memberi tahu saya saat terjadi masalah, saya dapat dengan cepat menggunakan dasbor terkait untuk mengukur dampak pelanggan, memvalidasi atau triase akar masalah, menanggulangi, dan mempersingkat waktu pemulihan. Bahkan jika alarm telah memulai alur kerja remediasi otomatis, saya harus lihat apa yang dilakukan alur kerja otomatis, apa pengaruhnya pada sistem, dan, dalam keadaan luar biasa, memajukan alur kerja dengan memberikan konfirmasi manusia untuk langkah keselamatan penting.

Saat suatu peristiwa sedang berlangsung, Amazon biasanya melibatkan beberapa operator panggilan. Para operator dapat menggunakan beragam dasbor saat menjalankan serangkaian tugas. Tugas-tugas ini biasanya antara lain mengukur dampak terhadap pelanggan, triase, menelusuri berbagai layanan hingga ke akar masalah, mengamati alur kerja remediasi otomatis, dan menjalankan dan memvalidasi berbagai langkah mitigasi berbasis runbook. Sementara itu, tim sejawat dan pemangku kepentingan bisnis juga menggunakan dasbor untuk memantau dampak yang sedang berlangsung selama peristiwa berlangsung. Berbagai peserta ini berkomunikasi menggunakan alat manajemen insiden, ruang obrolan (dengan bot seperti AWS Chatbot), dan panggilan konferensi. Setiap pemangku kepentingan memberikan perspektif berbeda atas data yang mereka lihat di dasbor.

Setiap minggu, tim dan organisasi Amazon yang lebih luas juga menjalankan rapat tinjauan operasi yang dihadiri para pemimpin senior, manajer, dan banyak pakar teknis. Pada rapat itu kami menggunakan roda keberuntungan untuk memilih dasbor audit tingkat tinggi. Pemangku kepentingan meninjau pengalaman pelanggan dan tujuan utama tingkat layanan kami, seperti ketersediaan dan latensi. Dasbor audit yang digunakan pemangku kepentingan ini biasanya menampilkan data operasional dari semua Availability Zone dan Wilayah.

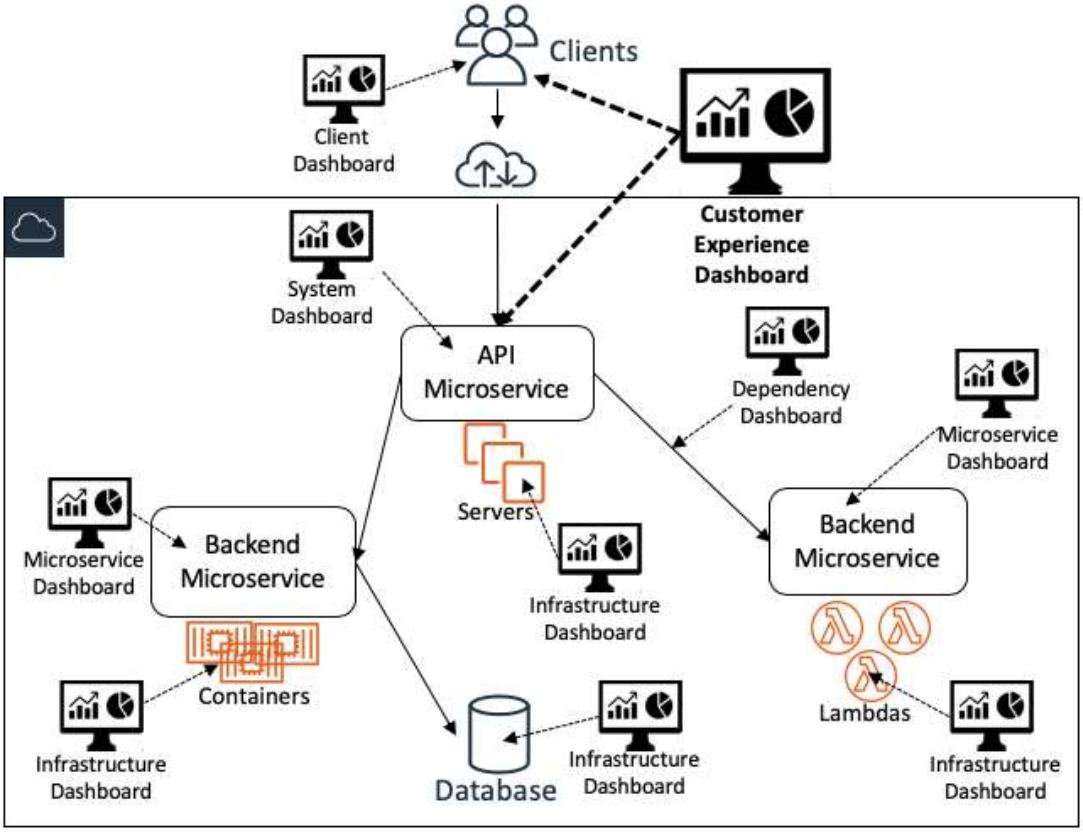
Selain itu, saat melakukan perencanaan dan prakiraan kapasitas jangka panjang, Amazon menggunakan dasbor yang memvisualisasikan bisnis tingkat tertinggi, penggunaan, dan metrik kapasitas yang dikeluarkan sistem kami dalam interval waktu lebih lama.

Jenis dasbor

Orang menggunakan dasbor untuk memantau layanan secara manual, tetapi satu solusi tidak bisa mengatasi semua kasus penggunaan. Untuk sebagian besar sistem, kami menggunakan banyak dasbor, yang masing-masing memberikan tampilan sistem secara berbeda-beda. Pandangan berbeda ini memungkinkan pengguna berbeda memahami bagaimana sistem kami berperilaku dari perspektif lain dan dalam interval waktu yang berbeda.

Data yang ingin dilihat setiap audiens bisa sangat bervariasi dari satu ke dasbor lainnya. Kami telah memahami untuk fokus pada audiens yang dituju saat kami mendesain dasbor. Kami memutuskan data apa yang masuk ke setiap dasbor berdasarkan siapa yang akan menggunakan dasbor dan mengapa mereka akan menggunakan dasbor. Anda mungkin pernah mendengar bahwa di Amazon kami bekerja mundur dari pelanggan. Pembuatan dasbor adalah contoh yang tepat untuk ini. Kami membangun dasbor berdasarkan kebutuhan pengguna yang diperkirakan dan persyaratan khusus mereka.

Diagram berikut menggambarkan bagaimana dasbor berbeda memberikan tampilan berbeda ke sistem secara keseluruhan:



Dasbor tingkat tinggi

Dasbor pengalaman pelanggan. Dasbor yang paling penting dan banyak digunakan di Amazon adalah dasbor pengalaman pelanggan. Dasbor ini dirancang untuk digunakan oleh sekelompok besar pengguna yang mencakup operator layanan dan banyak pemangku kepentingan lainnya. Semuanya secara efisien menyajikan metrik kesehatan layanan secara keseluruhan dan fokus pada tujuan. Semuanya menampilkan data pemantauan yang bersumber dari layanan itu sendiri dan juga dari instrumentasi klien (seperti metrik AWS SDK), penguji berkelanjutan (seperti kenari Amazon CloudWatch Synthetics), dan sistem remediasi otomatis. Dasbor ini juga berisi data yang membantu pengguna menjawab pertanyaan tentang kedalaman dan luas dampak. Sebagian pertanyaan ini kemungkinan besar adalah “Berapa banyak pelanggan yang terdampak?” dan “Pelanggan mana yang paling terdampak?”

Dasbor tingkat sistem. Titik masuk ke layanan berbasis web kita biasanya adalah titik akhir UI dan API, jadi dasbor tingkat sistem khusus harus memuat cukup data bagi operator untuk melihat bagaimana sistem dan titik akhir yang dihadapi pelanggan berperilaku. Dasbor ini terutama menampilkan data pemantauan tingkat antarmuka. Dasbor ini menampilkan tiga kategori data pemantauan untuk setiap API:

- **Data pemantauan terkait input.** Ini dapat mencakup jumlah permintaan yang diterima atau pekerjaan yang disurvei dari antrian/aliran, persentil ukuran byte permintaan, dan jumlah kegagalan otentikasi/otorisasi.
- **Data pemantauan terkait pemrosesan.** Ini dapat mencakup jalur logika bisnis multi-modal/jumlah eksekusi cabang, jumlah permintaan layanan mikro backend/persentil kegagalan/latensi, keluaran log kekeliruan dan galat, dan data pelacakan permintaan.
- **Data pemantauan terkait output.** Ini dapat mencakup jumlah tipe respons (dengan perincian untuk respons galat/kekeliruan oleh pelanggan), ukuran respons, dan persentil untuk byte respons pertama waktu-untuk-menulis dan respons selesai waktu-untuk-menulis.

Secara umum, kami bertujuan menjaga pengalaman pelanggan dan dasbor tingkat sistem ini setinggi mungkin. Kami sengaja menghindari godaan untuk menambahkan terlalu banyak metrik ke dasbor ini karena informasi yang berlebihan dapat mengalihkan perhatian dari pesan inti yang perlu disampaikan oleh dasbor tersebut.

Dasbor instans layanan. Kami membangun beberapa dasbor untuk memfasilitasi evaluasi cepat dan komprehensif atas pengalaman pelanggan dalam satu instans layanan (partisi atau sel). Tampilan sempit ini memastikan bahwa operator yang bekerja pada satu instans layanan tidak kelebihan beban dengan data yang tidak relevan dari instans layanan lain.

Dasbor audit layanan. Kami juga membangun dsasbor pengalaman pelanggan yang dimaksudkan untuk menampilkan data untuk semua instans suatu *layanan*, di semua Availability Zones dan Wilayah. Dasbor audit layanan ini digunakan oleh operator untuk mengaudit alarm otomatis di semua instans layanan. Alarm ini juga dapat ditinjau pada rapat operasi mingguan yang disebutkan di atas.

Dasbor perencanaan dan prakiraan kapasitas. Untuk kasus penggunaan jangka panjang, kami juga membuat dasbor untuk perencanaan dan prakiraan kapasitas guna membantu memvisualisasikan pertumbuhan layanan kami.

Dasbor tingkat rendah

API Amazon biasanya diimplementasikan dengan mengatur permintaan di seluruh layanan mikro backend. Layanan mikro ini dapat dimiliki oleh berbagai tim berbeda, yang masing-masing bertanggung jawab atas beberapa aspek tertentu dalam memproses permintaan. Misalnya, beberapa layanan mikro didedikasikan untuk meminta otentikasi dan otorisasi, pemberlakuan katup/batas, pengukuran penggunaan, membuat/memperbarui/menghapus sumber daya, mengambil sumber daya dari penyimpanan data, dan memulai alur kerja asinkron. Tim biasanya membuat sekurangnya satu dasbor spesifik layanan mikro tersendiri yang menampilkan metrik untuk tiap API, atau unit kerja jika layanan memproses data secara asinkron.

Dasbor spesifik layanan mikro. Dasbor untuk layanan mikro biasanya menampilkan data pemantauan spesifik implementasi yang mensyaratkan pengetahuan mendalam tentang layanan. Dasbor ini terutama digunakan oleh tim yang bertanggung jawab atas layanan. Namun, karena layanan kami memiliki banyak instrumen, kami perlu menyajikan data dari instrumentasi ini dengan cara yang tentu tidak membebani operator. Jadi, dasbor ini biasanya menampilkan beberapa data dalam bentuk agregat. Bila menemukan anomali dalam data agregat, operator biasanya menggunakan berbagai alat

lain untuk mendalami, mengeksekusi kueri ad-hoc atas data pemantauan dasar yang mengurai data, melacak permintaan, dan mengungkapkan data yang berkaitan atau berkorelasi.

Dasbor infrastruktur. Layanan kami berjalan di infrastruktur AWS yang biasanya mengeluarkan metrik, jadi kami juga memiliki dasbor infrastruktur khusus. Dasbor ini fokus terutama pada metrik yang dikeluarkan oleh sumber daya komputasi yang dijalankan sistem kami, seperti instans Amazon Elastic Compute Cloud (EC2), kontainer Amazon Elastic Container Service (ECS)/Amazon Elastic Kubernetes Service (EKS), dan fungsi AWS Lambda. Metrik seperti penggunaan CPU, lalu lintas jaringan, IO disk, dan pemanfaatan ruang biasanya digunakan di dasbor ini, bersama dengan klaster terkait, Auto Scaling, dan metrik kuota yang relevan dengan sumber daya komputasi ini.

Dasbor dependensi. Selain sumber daya komputasi, dalam banyak hal layanan mikro bergantung pada layanan mikro lain. Meskipun tim yang memiliki dependensi tersebut sudah memiliki dasbornya sendiri, setiap pemilik layanan mikro biasanya membuat dasbor dependensi khusus untuk memberikan gambaran tentang bagaimana dependensi upstream (misalnya, proxy dan penyeimbang beban) dan dependensi downstream (misalnya, penyimpanan data, antrian, dan stream) berperilaku, yang diukur dengan layanan mereka. Dasbor ini juga dapat digunakan untuk melacak metrik penting lain, seperti tanggal kedaluwarsa sertifikat keamanan dan penggunaan kuota dependensi lainnya.

Desain dasbor

Bagi kami di Amazon, konsistensi dalam penyajian data sangatlah penting untuk keberhasilan pembuatan dasbor. Agar efektif, konsistensi harus dicapai di tiap dasbor dan juga di seluruh dasbor. Selama bertahun-tahun, kami telah mengidentifikasi, mengadaptasi, dan menyempurnakan serangkaian idiom dan konvensi umum desain yang kami yakini akan membuat dasbor dapat diakses oleh audiens paling luas, dan pada akhirnya meningkatkan nilainya bagi organisasi kami. Kami bahkan menemukan cara apik untuk mengukur dan meningkatkan konvensi desain ini dari waktu ke waktu. Misalnya, jika operator baru dapat dengan cepat memahami dan menggunakan data yang disajikan di dasbor untuk mempelajari cara kerja suatu layanan, ini menjadi indikasi bahwa dasbor tersebut menyajikan informasi yang benar dengan cara yang benar.

Kecenderungan yang sangat lazim saat mendesain dasbor adalah melebih-lebihkan atau meremehkan pengetahuan domain pengguna target. Sangat mudah untuk membuat dasbor yang benar-benar masuk akal bagi pembuatnya. Namun, dasbor ini mungkin tidak memberikan nilai bagi pengguna. Kami menggunakan teknik bekerja mundur dari pelanggan (dalam hal ini, pengguna dasbor) untuk meniadakan risiko ini.

Kami telah mengadopsi konvensi desain yang menstandarkan tata letak data di dasbor. Dasbor dirender dari atas ke bawah, dan pengguna cenderung menafsirkan grafik yang pertama kali dirender (terlihat saat dasbor memuat) sebagai yang paling penting. Jadi, konvensi desain kami menyarankan untuk menempatkan data paling penting di bagian atas dasbor. Kami mendapati bahwa grafik ketersediaan agregat/ikhtisar dan grafik persentil latensi ujung ke ujung biasanya menjadi dasbor paling penting untuk layanan web.

Berikut tangkapan layar bagian atas suatu dasbor untuk hipotetis Layanan Foo:

Kami menggunakan grafik lebih besar untuk metrik yang paling penting. Jika ada banyak metrik dalam satu grafik, kami memastikan bahwa legenda grafik tidak secara vertikal atau horisontal menekan data grafik yang terlihat. Jika kami menggunakan kueri penelusuran dalam grafik, kami memastikan memberi izin kumpulan hasil metrik yang lebih besar dari biasanya.

Kami menata letak grafik untuk resolusi tampilan minimum yang diharapkan. Ini menghindari memaksa pengguna menggulir secara horisontal. Operator panggilan di laptop pada pukul 3 pagi mungkin tidak memperhatikan bilah gulir horisontal tanpa petunjuk visual yang jelas bahwa ada lebih banyak grafik di sebelah kanan.

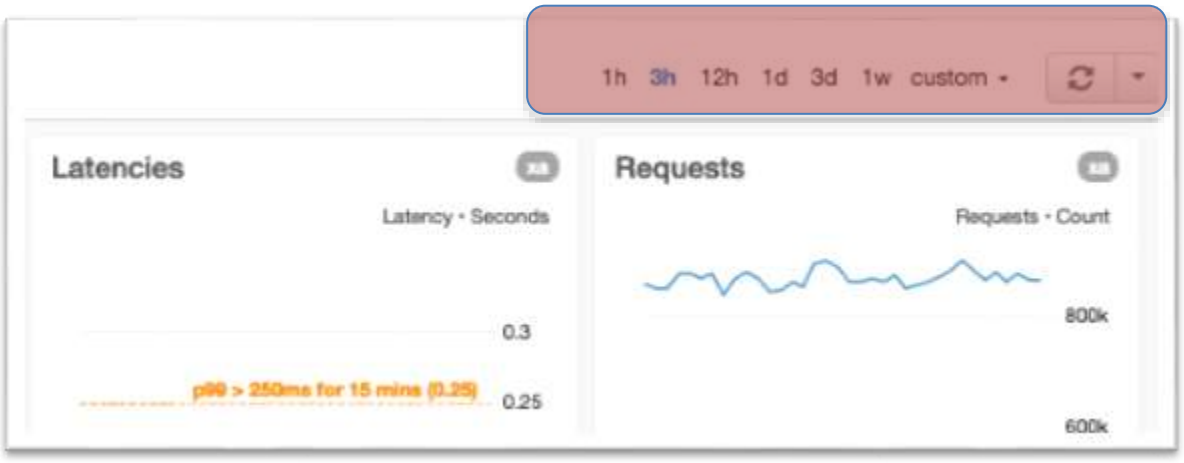
Kami menampilkan zona waktu. Untuk dasbor yang menampilkan data tanggal dan waktu, kami pastikan bahwa zona waktu yang sesuai akan terlihat di dasbor. Untuk dasbor yang digunakan bersamaan oleh operator di zona waktu berbeda, secara default kami menggunakan satu zona waktu (UTC) yang dapat mempersatukan persepsi semua pengguna. Dengan cara ini pengguna dapat berkomunikasi satu sama lain menggunakan satu zona waktu, menghemat waktu dan tenaga mengalihkan zona waktu berlebihan secara mental.

Kami menggunakan interval waktu dan periode titik data terpendek. Kami men-default ke interval waktu dan periode titik data yang relevan dengan kasus penggunaan paling umum. Kami memastikan bahwa semua grafik di dasbor sejak awal menampilkan data untuk rentang waktu dan resolusi yang sama. Kami rasa akan bermanfaat jika semua grafik dalam suatu bagian dasbor memiliki ukuran horisontal yang sama. Dengan demikian korelasi waktu akan mudah antar grafik.

Kami juga menghindari memplot terlalu banyak titik data dalam grafik karena ini memperlambat waktu muat dasbor. Selain itu, menurut amatan kami menampilkan titik data yang berlebihan kepada pengguna sebenarnya dapat menurunkan visibilitas menjadi anomali. Misalnya, grafik interval tiga jam titik data resolusi satu menit dengan hanya 180 nilai per metrik akan tampil dengan jelas bahkan pada widget dasbor kecil. Jumlah titik data ini juga memberikan cukup konteks bagi operator yang melakukan triase acara operasional yang sedang berlangsung.

Kami mengaktifkan kemampuan menyesuaikan interval waktu dan periode metrik. Dasbor kami merupakan kontrol untuk menyesuaikan dengan cepat interval waktu dan periode metrik untuk semua grafik. Rasio resolusi x interval umum lainnya yang kami gunakan di dasbor kami adalah:

- 1-jam x 1 menit (60 titik data) - berguna untuk memperbesar tampilan untuk mengamati acara yang sedang berlangsung
- 12-jam x 1 menit (720 titik data)
- 1-hari x 5 menit (288 titik data) - berguna untuk melihat tren harian
- 3-hari x 5 menit (864 titik data)
- 1-minggu x 1 jam (168 titik data) - berguna untuk melihat tren mingguan
- 1-bulan x 1 jam (744 titik data)
- 3-bulan x 1 hari (90 titik data) - berguna untuk melihat tren triwulanan
- 9-bulan x 1 hari (270 titik data)
- 15-bulan x 1 hari (450 titik data) - berguna untuk tinjauan kapasitas jangka panjang



Kami menganotasi grafik dengan ambang alarm. Saat kami membuat grafik metrik yang memiliki alarm otomatis terkait, jika ambang alarm statis, kami menganotasi grafik dengan garis horisontal. Jika ambang batas alarm dinamis, yaitu, berdasarkan prakiraan atau prediksi yang dihasilkan dengan kecerdasan buatan (AI) atau pembelajaran mesin (ML), kami menampilkan metrik aktual dan ambang batas pada grafik yang sama. Jika grafik menunjukkan metrik yang mengukur aspek layanan dengan batas yang diketahui (seperti batas "maksimum yang diuji" atau batas sumber daya sulit), kami menganotasi pada grafik dengan garis horisontal yang menunjukkan di mana batas yang diketahui atau yang diuji berada. Untuk metrik yang memiliki tujuan, kami menambahkan garis horisontal agar tujuan itu segera terlihat oleh pengguna.

Kami menghindari penambahan garis horisontal pada grafik yang sudah menggunakan sumbu y kiri dan kanan. Jika Anda menambahkan garis horisontal pada grafik ini, pengguna mungkin akan kesulitan mengetahui terkait dengan sumbu y mana garis horisontal tersebut. Untuk menghindari ambiguitas ini, kami membagi grafik seperti ini menjadi dua grafik yang hanya menggunakan satu sumbu horisontal dan menambahkan garis horisontal hanya pada grafik yang sesuai.

Kami menghindari pembebanan berlebihan sumbu y dengan beberapa metrik dengan rentang nilai yang sangat berbeda. Kami menghindari situasi ini karena dapat mengakibatkan berkurangnya visibilitas ke varian satu atau lebih metrik. Contohnya adalah ketika kita memplot latensi p0 (minimum) dan p100 (maksimum) pada grafik yang sama di mana nilai titik data p100 mungkin berkali lipat lebih besar dari titik data p0.

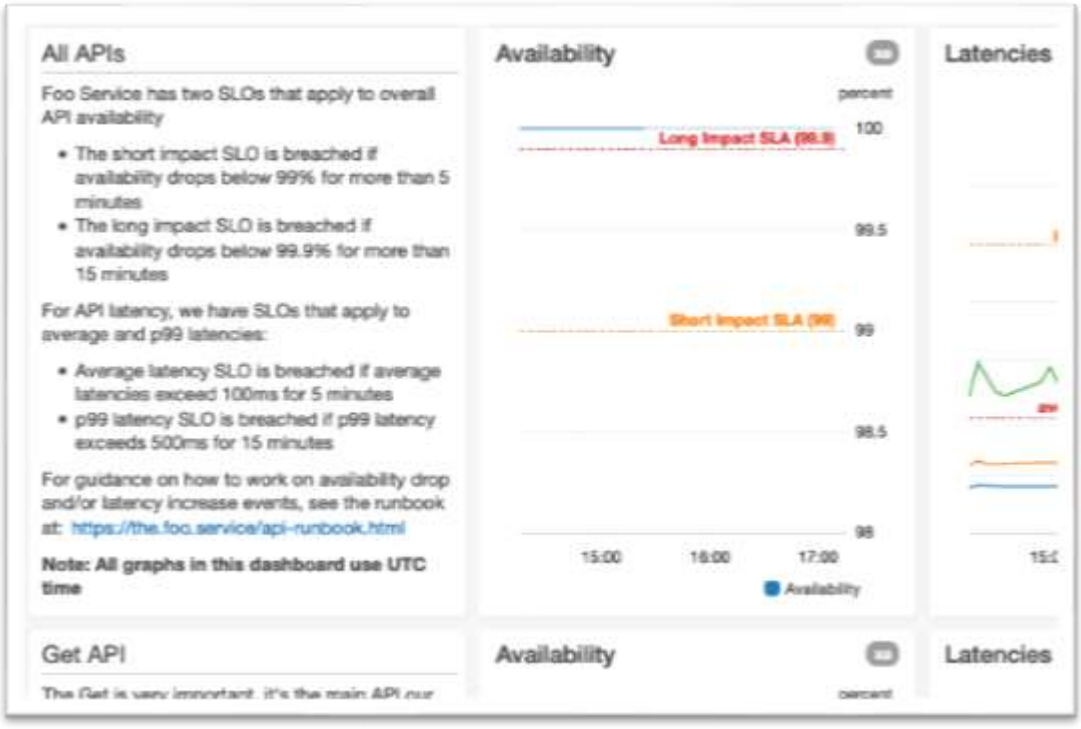
Kami berhati-hati dalam memperkecil batas sumbu y menjadi kisaran nilai titik data saat ini saja. Sekilas pandang pada grafik dengan rentang sumbu y yang terbatas pada nilai titik data dapat membuat metrik terlihat jauh lebih variabel dari yang sebenarnya.

Kami menghindari kelebihan beban grafik tunggal. Kami ingin memastikan bahwa tidak ada terlalu banyak statistik atau metrik tanpa kaitan dalam satu grafik. Misalnya, saat menambahkan grafik untuk pemrosesan permintaan, kami biasanya membuat grafik berdekatan yang terpisah di dasbor untuk yang berikut ini:

- Ketersediaan % (Kesalahan/Permintaan * 100)
- p10, Rerata, p90 latensi
- p99,9 dan Maksimum (p100) latensi

Kami berasumsi bahwa pengguna tidak tahu persis apa arti setiap metrik atau widget. Ini berlaku khususnya untuk metrik spesifik implementasi. Kami ingin memberikan konteks yang cukup pada teks dashboard, misalnya dengan teks deskripsi di samping atau di bawah setiap grafik. Operator dapat membaca teks ini untuk memahami arti metrik. Kemudian operator dapat menafsirkan seperti apa bentuk "normal" dan apa artinya jika grafik tidak "normal." Dalam teks ini, kami menyediakan tautan ke sumber daya terkait yang dapat digunakan operator untuk menentukan akar masalah. Berikut beberapa contoh jenis tautan yang kami sediakan:

- Ke runbook. Untuk ahli bidang studi, dasbor bisa jadi runbook itu sendiri.
- Ke dasbor “pendalaman” terkait.
- Ke dasbor yang setara untuk klaster atau partisi lain.
- Ke pipeline penerapan.
- Ke informasi kontak untuk dependensi.



Kami menggunakan status alarm, angka sederhana, dan/atau widget grafik deret waktu jika sesuai. Tergantung kasus penggunaan untuk dasbor, kami mendapati bahwa menampilkan widget yang berisi satu angka (misalnya, nilai terbaru suatu metrik) atau status alarm terkadang lebih sesuai dari menampilkan grafik deret waktu kompleks semua titik data terbaru.

Kami menghindari mengandalkan grafik yang menampilkan metrik rongga. *Metrik rongga* adalah metrik yang hanya ditampilkan jika ada kondisi galat tertentu. Meskipun layanan instrumen bisa efisien untuk mengeluarkan metrik ini hanya jika diperlukan, pengguna dasbor dapat dibingungkan oleh grafik kosong atau hampir kosong. Bila kami temukan metrik semacam itu saat mendesain dasbor, kami biasanya memodifikasi layanan agar terus mengeluarkan nilai aman (yaitu, nol) untuk metrik ini jika tidak ada kondisi galat. Operator kemudian dapat dengan mudah memahami bahwa tidak adanya data menyiratkan bahwa layanan tidak mengeluarkan telemetri dengan benar.

Kami menambahkan grafik tambahan yang menampilkan metrik per mode. Kami melakukan ini saat kami menampilkan grafik untuk metrik yang menggabungkan perilaku multi-model di sistem kami. Beberapa keadaan yang memicu kemungkinan kita melakukan ini termasuk:

- Jika layanan mendukung permintaan berukuran variabel, kami mungkin akan membuat grafik untuk keseluruhan latensi permintaan. Selain itu, kami juga dapat membuat grafik yang menampilkan metrik untuk permintaan kecil, sedang, dan besar.
- Jika suatu layanan menjalankan permintaan dengan cara bervariasi tergantung nilai (atau kombinasi) parameter input, maka kami dapat menambahkan grafik untuk metrik yang menangkap setiap mode eksekusi.

Pemeliharaan dasbor

Membangun dasbor yang menampilkan banyak tampilan sistem kami adalah langkah pertama. Namun, sistem kami terus berkembang dan meningkat, dan dasbor perlu berkembang bersama dengan itu, seiring fitur baru ditambahkan dan arsitektur ditingkatkan. Memelihara dan memperbarui dasbor tertanam dalam proses pengembangan kami. Sebelum menyelesaikan perubahan, dan selama peninjauan kode, pengembang kami bertanya, "Apakah dasbor perlu diperbarui?" Mereka diberdayakan untuk membuat perubahan pada dasbor sebelum perubahan yang mendasarinya diterapkan. Ini menghindari situasi di mana operator harus memperbarui dasbor selama atau setelah penerapan sistem guna memvalidasi perubahan yang sedang diterapkan.

Jika dasbor berisi informasi yang jauh lebih rinci dari biasanya, ini mungkin menunjukkan bahwa operator mengandalkan dasbor tersebut untuk deteksi anomali manual sebagai ganti peringatan dan remediasi otomatis. Kami terus mengaudit dasbor kami untuk menentukan apakah upaya manual ini dapat diredam dengan meningkatkan instrumentasi pada layanan dan meningkatkan alarm otomatis kami. Kami juga secara agresif memangkas atau memperbarui grafik yang tidak lagi menambah nilai pada dasbor.

Dengan memungkinkan pengembang kami memperbarui dasbor, kami memastikan terciptanya seperangkat dasbor lengkap dan identik untuk lingkungan pra-produksi (alfa, beta, atau gamma) kami. Pipeline penerapan otomatis kami menerapkan perubahan ke lingkungan pra-produksi terlebih

dahulu. Jadi, tim kami harus bisa dengan mudah memvalidasi perubahan di lingkungan pengujian ini menggunakan dasbor terkait (dan peringatan otomatis) dengan cara yang benar-benar konsisten dengan cara validasi perubahan tersebut saat perubahan didorong ke lingkungan produksi kami.

Sebagian besar sistem terus berkembang karena persyaratan diperbarui, fitur baru ditambahkan, dan arsitektur perangkat lunak berubah untuk mengakomodasi peningkatan dari waktu ke waktu. Dasbor kami adalah komponen penting sistem kami, jadi kami mengikuti proses Infrastructure-as-Code (IaC) untuk pemeliharaannya. Proses ini memastikan bahwa dasbor kami terpelihara dalam sistem kontrol versi dan bahwa perubahan diterapkan pada dasbor kami menggunakan alat yang sama yang digunakan oleh pengembang dan operator untuk layanan kami.

Saat kami melakukan post-mortem untuk kejadian operasional yang tidak terduga, tim kami meninjau apakah penyempurnaan dasbor (dan peringatan otomatis) dapat mendahului kejadian tersebut, mengidentifikasi akar masalah lebih cepat, atau mengurangi waktu rerata pemulihan. Kami biasanya bertanya pada diri sendiri, “Secara retrospeksi, apakah dasbor dengan jelas menunjukkan dampak pelanggan, membantu operator melakukan triangulasi untuk menentukan akar masalah utama, dan membantu mengukur waktu pemulihan?” Jika jawaban untuk salah satu pertanyaan ini adalah tidak, maka post-mortem kami menyertakan tindakan untuk menyempurnakan dasbor tersebut.

Penutup

Di Amazon, kami mengoperasikan layanan skala besar di seluruh dunia. Sistem otomatis kami terus memantau, mendeteksi, memperingatkan, dan memulihkan masalah apa pun yang terjadi. Kami butuh kemampuan untuk memantau, mendalami, mengaudit, dan meninjau semua layanan dan sistem otomatis ini. Untuk mencapai hal ini, kami membangun dan memelihara dasbor yang memberikan banyak tampilan berbeda sistem kami. Kami merancang dasbor ini untuk audiens luas dan khusus dengan bekerja mundur dari pengguna dasbor. Agar dasbor lebih mudah dipahami oleh operator dan pemilik layanan, kami menggunakan kumpulan idiom dan konvensi desain yang konsisten untuk memastikan kegunaan dan utilitas dasbor.

Dasbor kami memberikan banyak perspektif dan pandangan berbeda tentang bagaimana layanan AWS beroperasi. Semua itu memainkan peran penting dalam memberikan pengalaman pelanggan yang luar biasa dengan membantu tim Amazon memahami, mengoperasikan, dan meningkatkan layanan kami. Kami harap artikel ini membantu Anda saat Anda merancang, membangun, dan memelihara dasbor Anda sendiri. Jika Anda ingin melihat contoh cara membuat dasbor menggunakan layanan AWS, berikut video singkat dan panduan layan mandiri.