Reliable Software Systems

Week 2: Expect Failure

January 17, 2018
Alyssa Pittman
University of Washington Allen School

Example outage: Slack

Two incidents: full outage for 14 minutes, then 13% of users for 2 hours

Routine maintenance accidentally corrupted jobs that processed queues

Fixed immediately! But 13% of users had already been disconnected

They all tried to reconnect simultaneously

That overloaded the databases

Slack added more database capacity, and over 2 hours later, it was all peachy

https://slackhq.com/this-was-not-normal-really

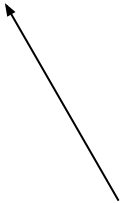
Embrace failure

“100% is the wrong reliability target for basically everything (pacemakers and anti-lock brakes being notable exceptions). In general, for any software service or system, 100% is not the right reliability target because no user can tell the difference between a system being 100% available and 99.999% available. There are many other systems in the path between user and service (their laptop, their home WiFi, their ISP, the power grid...) and those systems collectively are far less than 99.999% available. Thus, the marginal difference between 99.999% and 100% gets lost in the noise of other unavailability, and the user receives no benefit from the enormous effort required to add that last 0.001% of availability.”

https://landing.google.com/sre/sre-book/chapters/introduction/

Five 9’s

$$(1 - .99999) \cdot 365 \frac{days}{year} \cdot 24 \frac{hours}{day} \cdot 60 \frac{minutes}{hour} \cong 5.25 \frac{minutes}{year}$$



See, there’s five nines

How much failure is too much?

SLI: Service Level Indicator

Measurement of health in the system

SLO: Service Level Objective

Target levels for SLIs

SLA: Service Level Agreement

SLO promised to a customer + repercussions for failing to meet it

Example SLO: gSuite (paid gMail)

G Suite

Gmail Service Level Agreement for G Suite for ISPS (paid)

1. Gmail SLA. Google shall use all reasonable commercial efforts to ensure that the Gmail web interface is operating and available to Customers 99.9% of the time in any calendar month. In the event Customer experiences any of the service performance issues defined below due to Google's failure to provide Services, Customer will be eligible to receive the Service Credits described below (the "Gmail SLA").
2. Definitions. The following definitions shall apply to the Gmail SLA.

   "Downtime" means, for a domain, if there is more than a five percent user error rate. Downtime is measured based on server side error rate.

   "Downtime Period" means, for a domain, if a period of ten consecutive minutes of Downtime. Intermittent Downtime for a period of less than ten minutes will not be counted towards any Downtime Periods.

   "Monthly Uptime Percentage" means total number of minutes in a calendar month minus the number of minutes of Downtime suffered from all Downtime Periods in a calendar month, divided by the total number of minutes in a calendar month.

   "Scheduled Downtime" means those times where Google notifies Customers of periods of Downtime five days prior to the commencement of such Downtime. There will be no more than twelve hours of Scheduled Downtime per calendar year. Scheduled Downtime is not considered Downtime for purposes of this Gmail SLA, and will not be counted towards any Downtime Periods.

   "Service" means the G Suite for ISPs (paid) service provided by Google to You under the G Suite for ISPs (paid) Agreement.

   "Service Credit" means: (a) 10% of the total invoice charges for the affected month if the Monthly Uptime Percentage for any calendar month is between 99.0% and 99.9%; or (b) 25% of the total invoice charges for the affected month if the Monthly Uptime Percentage for any calendar month is between 99.0% and 95.0 %; or (c) 50% of the total invoice charges for the affected month if the Monthly Uptime Percentage for any calendar month is less than 95.0%.

https://gsuite.google.com/terms/partner_sla.html

Example SLO: Twilio messaging

b. *"Service Credit"* means a dollar credit, equal to 10% of your usage fees in the month in which the Unavailable Time (as defined below) occurred, that Twilio will credit back to an eligible account.

Talk to an expert | Sign up

c. *"Unavailable Time"* means the Twilio API for the applicable product is not available for use, as measured in continuous 5-minute increments. Unavailable Time does not include any unavailability resulting from any Exclusion (as defined below).

Monthly Uptime Percentage	Service Credit
<99.95%[1]	10%

[1] *If you purchase the Twilio Enterprise Plan, then Twilio will make the Twilio API available 99.99% of the time each month and all references to a 99.95% Monthly Uptime Percentage in this SLA shall be replaced with a 99.99% Monthly Uptime Percentage. If you are no longer subscribed to the Twilio Enterprise Plan, then the Monthly Uptime Percentage shall revert to 99.95%.*

2. Service Commitment

Twilio will make the Twilio API available 99.95% of the time each month. If Twilio's Monthly Uptime Percentage is below 99.95% in a given calendar month, then you will be eligible to receive a Service Credit as described in Section 3 below. Availability of the Twilio API is measured by the third party performance and monitoring services contracted by Twilio (the *"Monitoring Service"*). The Monitoring Service reports of availability are currently available at http://status.twilio.com. Twilio may adjust the measure of availability by the Monitoring Service to account for any Exclusions applicable to such period.

https://www.twilio.com/legal/service-level-agreement

Example SLO: 100% availability

Cloud DNS > Documentation

Google Cloud DNS Service Level Agreement (SLA)

☆ ☆ ☆ ☆ ☆

SEND FEEDBACK

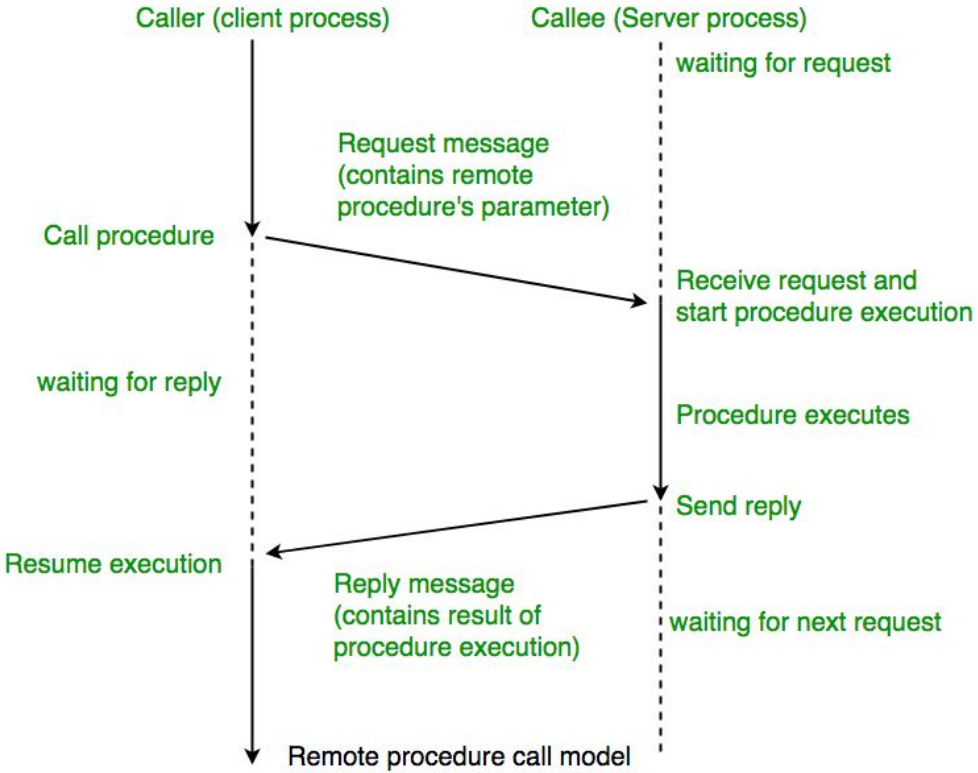
Last modified: October 4, 2016 | Previous Versions

During the Term of the Google Cloud Platform License Agreement, or Google Cloud Platform Reseller Agreement (as applicable, the "Agreement"), the Covered Service will provide a Monthly Uptime Percentage of Serving DNS queries from at least one of the Google managed Authoritative Name Servers to Customer of 100% (the "Service Level Objective" or "SLO"). If Google does not meet the SLO, and if Customer meets its obligations under this SLA, Customer will be eligible to receive the Financial Credits described below. This SLA states Customer's sole and exclusive remedy for any failure by Google to meet the SLO. Capitalized terms used in this SLA, but not defined in this SLA, have the meaning set forth in the Agreement. If the Agreement is the Google Cloud Platform Reseller Agreement, then all references to "Customer" in this SLA mean "Reseller," and any Financial Credit(s) will only apply for impacted Reseller order(s) under the Agreement.

https://cloud.google.com/dns/sla

Sources of failure

Sources of failure



Sources of failure

Client can’t initially connect to server

Client can’t reach server

API mismatch

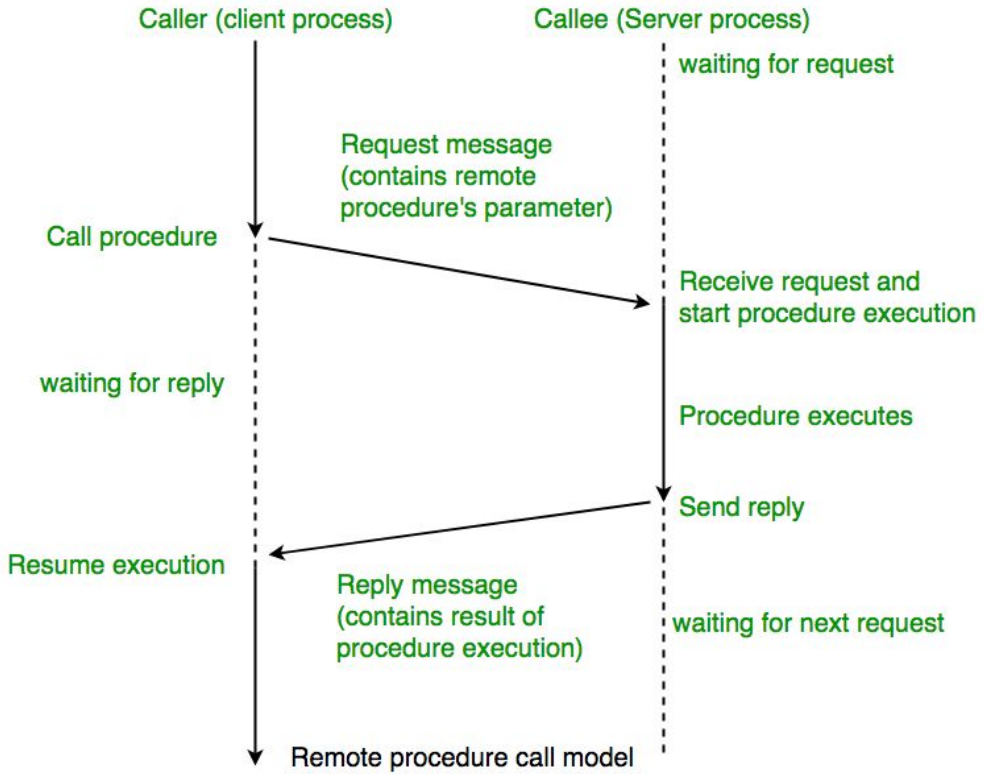
Timeout

Server overloaded

Sources of failure

Reply message may have:

- The response
- An error code
  - Server error
  - Application error
  - Timeout
  - Etc



Limit the blast radius

As a client, you should try to:

Keep failure from affecting...

- Other requests
- Other servers

Keep failure from taking down....

- The servers you contact
- The clients contacting you

Graceful degradation

Don’t crash (e.g no System.exit(1), catch exceptions)

- Otherwise “query of death” can topple the service

Consider failing open

Consider fallback paths

- Can you have default data or leave some part of the user experience out?

Consider skipping work if the system knows it’s overloaded

Pick good defaults

What do you do if cache data never loaded, or fails to reload?

- Fail
- Use an old set of data
- Use some pre-loaded default data
- Use an empty dataset

I’ve seen all of these in practice, be careful when choosing.

Protect resources from overloading

Retries

- With exponential backoff!
- And jitter!
- And stop after a maximum number of retries!

Idempotency

Idempotent APIs:

Clients can make the same call repeatedly while producing the same result.

Can you tell whether the failure actually failed or not?

A client-side timeout could have occurred after the server finished

Can you safely retry?

A read will be safe

A delete will be safe (it may fail, but the data will already be gone)

A write could overwrite data that sneaked in between

Protect resources from overloading

Set deadlines

- Propagate deadlines that you've been given

Bulkhead

- Limit the number of concurrent requests

Plan to avoid cascading failures

END