New API Rate Limiting

Church Community Builder employs a per-day rate limiting strategy to ensure Church Partners (both customers and partner organizations) receive consistent access to the API and their Church Community Builder sites. While Church Partners have always been limited by a daily allotment of API calls, as of **Monday, August 20, 2018** (MDT), Church Community Builder will also implement a new **per-minute rate limiting strategy** that will restrict how many times per minute an API user can access an individual API Service. All API Users belonging to a Church Partner will share this per-minute allotment. These per-minute allotments are, when extrapolated out, less restrictive than the current per-day limits, though the per-day allotment will continue to be in place for now.

Why the Change?

Our primary goal in implementing rate limiting is to protect your ability to access your church's data and Church Community Builder site by ensuring that requests are made at a reasonable and sustainable rate. Our hope for per-minute rate limiting is that we can eliminate the daily API call limits and provide an easy way for you to safely meter your usage of our API via HTTP Response Header values.

HTTP Headers and Response Codes

Church Partners can use the HTTP Headers we provide in our response to API calls in order to meter their usage of our API and determine their rate limiting status per API Service. These response headers are already implemented and available for you to use so you can begin altering your clients to adhere to the new rate limits.

A successful, non-rate-limited API call generates an HTTP 200 response with the following headers:

- **x-ratelimit-limit** - The number of calls per minute that can be made to an API Service before rate limiting will take effect. If all API Users collectively make fewer than this number of calls per minute to a particular API Service, they will never be rate limited.
- **x-ratelimit-remaining** - The number of calls an API Service has left in the current rate limit window. If all API Users stop making calls to an API Service when this value approaches 0, they will never encounter rate limiting.
- **x-ratelimit-reset** - The time at which an API Service's current rate limit window resets, in UTC epoch seconds. If all API Users wait until this time to make another request to this same API Service, they will never encounter rate limiting and will ensure they are making requests at the fastest allowable rate.

When a client exceeds these per-minute rate limits, we make a record of it in our systems. As of **Monday, August 20, 2018** (MDT), requests in excess of an API Service's limit will not be processed. Instead, an HTTP 429 "Too many requests" response will be returned with the following header:

- **retry-after** - The number of seconds in which an API Service will become available after its rate limit has been exceeded

You can begin modifying your API clients now to prepare for the new rate limiting restrictions so that you never encounter our per-minute rate limits when they are turned on, because the HTTP 200 headers detailed above are already being provided by our API Services. A new API Service called `rate_limit_test` has been added to allow you to test the rate limiting service without penalty. These calls will not count toward your daily API call limit and will return an HTTP 429 response when you burst more than 5 calls or exceed 10 calls per minute. Calls to this API Service are available to all API Users by default; no additional configuration is needed.

Example

Suppose you have a single API client that wants to make calls to the API Service `individual_profiles` at the fastest allowable rate. The first call to the API Service will return the following response:

```
curl -u user:pass -i
"https://ccb.ccbchurch.com/api.php?srv=individual_profiles&modified_since=2018-01-01"

HTTP/2 200
date: Wed, 13 Jun 2018 21:20:19 GMT
content-type: text/xml;charset=utf-8
x-ratelimit-limit: 15
x-ratelimit-remaining: 14
x-ratelimit-reset: 1528924825

<?xml version="1.0" encoding="UTF-8"?>
<ccb_api>
    <request>
        <parameters>
            <argument value="individual_profiles" name="srv"/>
        </parameters>
    </request>
    <response>
        ...
    </response>
</ccb_api>
```

If you have complete control over when your client makes API requests, your client should wait until the time specified by **x-ratelimit-reset** (June 13, 2018 9:20:25 PM GMT) before making another request to this API Service. In this case, this will be six seconds after the first request, which means the client will continue to make requests to this API Service at a rate of one call every 6 seconds, or 10 calls per minute. This strategy will work to safely achieve maximum throughput for any number of clients accessing the same API Service simultaneously. Assuming the client implements this strategy, its next request, which will occur six seconds later, will receive the following response:

```
curl -u user:pass -i
"https://ccb.ccbchurch.com/api.php?srv=individual_profiles&modified_since=2018-01-01"

HTTP/2 200
date: Wed, 13 Jun 2018 21:20:25 GMT
content-type: text/xml;charset=utf-8
x-ratelimit-limit: 15
x-ratelimit-remaining: 14
x-ratelimit-reset: 1528924831

<?xml version="1.0" encoding="UTF-8"?>
<ccb_api>
    <request>
        <parameters>
            <argument value="individual_profiles" name="srv"/>
        </parameters>
    </request>
    <response>
        ...
    </response>
</ccb_api>
```

If you don't have complete control when your client makes requests (for instance, the call is triggered by a user visiting a website), you can use **x-ratelimit-limit** and **x-ratelimit-remaining** to make sure your requests will be serviced. **x-ratelimit-limit** indicates that the allowable burst rate for this API Service is 15...that is, the client can make up to 15 calls, regardless of frequency, before being limited to one call per 6 seconds. As long as **x-ratelimit-remaining** is greater than the number of clients making calls, the call will be serviced. If the client uses this strategy and makes another immediate request, it will receive the following response:

```
curl -u user:pass -i
"https://ccb.ccbchurch.com/api.php?srv=individual_profiles&modified_since=2018-01-01"

HTTP/2 200
date: Wed, 13 Jun 2018 21:20:20 GMT
content-type: text/xml;charset=utf-8
x-ratelimit-limit: 15
x-ratelimit-remaining: 13
x-ratelimit-reset: 1528924831

<?xml version="1.0" encoding="UTF-8"?>
<ccb_api>
    <request>
        <parameters>
            <argument value="individual_profiles" name="srv"/>
        </parameters>
    </request>
    <response>
        ...
    </response>
</ccb_api>
```

This call was serviced even though it was made before **x-ratelimit-reset** had elapsed because **x-ratelimit-remaining** still had 14 burst requests available to it. However, **x-ratelimit-reset** increased by 12 seconds (6 seconds for the first request, plus another 6 seconds for the second request), and **x-ratelimit-remaining** now only has 13 burst requests available. This is the server's way of telling the client that it needs to slow down...it won't be able to sustain the rate it's currently operating at and it's risking receiving an HTTP 429 response at its current rate.

Let's say the client continues to make requests at a rate that's faster than the intended one-call-per-6-second rate...in fact, it immediately attempts another 20 requests to this same API Service. After 13 successful requests, it will receive the following response on requests 14-20:

```
HTTP/2 429
date: Wed, 13 Jun 2018 21:20:20 GMT
content-type: text/html; charset=utf-8
x-ratelimit-limit: 15
x-ratelimit-remaining: 0
x-ratelimit-reset: 1528924909
retry-after: 6
```

Here we can see that the server is now enforcing the one-call-per-6-seconds limit by means of an HTTP 429 response with a **retry-after** value. Any calls made before the **retry-after** value will receive additional HTTP 429 responses. Additionally, **x-ratelimit-reset** has increased to a full 90 seconds after the first request (15 requests * 6 seconds), indicating to the client that it will take 90 seconds for **x-ratelimit-remaining** to fill back up to **x-ratelimit-limit**. **x-ratelimit-remaining** is now zero since all 15 burst requests were used. The safest strategy to handle this is to wait until **x-ratelimit-reset** has elapsed before accessing this API Service again. If, however, the desire is to service the next request as soon as possible (e.g, a visitor to your website it waiting on this response), the client can wait the minimum amount of time (specified by **retry-after** value) and make another request. The risk of doing so is that the client's buffer of burst allotment is depleted, so if another API User, or other clients of this same API User, is currently attempting to access ccb.ccbchurch.com's `individual_profiles` API Service as well, one or both of you will receive HTTP 429's more often. HTTP 429 responses will be monitored in our infrastructure and excessive violations may result in further limiting an API User's use of the API. Caching results and sharing these header values amongst clients can help you avoid running into this situation.

Tips to Avoid Being Rate Limited

Below are some tips for metering your API usage so that you never encounter our API Service rate limits.

For clients with predictable API usage patterns, always respect x-ratelimit-reset

For predictable workloads, like overnight synchronization jobs, always have your client wait until **x-ratelimit-reset** has elapsed to make another request to the same API Service, including when you receive an HTTP 429. This will ensure you're using the API responsibly and as fast as intended. Our per-minute limits are generous and should be adequate to allow you to get the data you need in a timely manner.

For clients with unpredictable API usage patterns, always respect x-ratelimit-limit

Whenever your API usage will be characterized by bursts of use, consult **x-ratelimit-remaining** to make sure you have enough remaining requests to make a call. Delay the call until **x-ratelimit-reset** or **retry-after** once your burst allotment is depleted. Keep in mind that multiple clients or multiple API Users accessing the same API service will deplete this value faster and that you will want to stop making calls when **x-ratelimit-remaining** is equal to or less than the number of clients accessing the API Service concurrently (not just when it reaches zero).

Cache Your Results

For data that doesn't change often, cache the results of your API calls locally. Only request data from your API when you need to refresh your local cache.

Understand that x-ratelimit-limit and x-ratelimit-reset are dynamic values

Your **x-ratelimit-limit** value and **x-ratelimit-reset** frequency may change over time. Several factors could contribute to this value changing, but the important point to note is that you should avoid hardcoding your client to specific values in this field.

Review Your Current API Usage

Now is an optimal time to take a look at your current API usage and determine if you still need all of the data you're requesting! Remember that API Rate Limits are shared by all of your API Users, but are only applied on a per API Service basis. Using one API Service will not impact your ability to use other API Services, but using an API Service at the same time as another API User may. Now is also a good time to review the API Users you've created and make sure that they all still need access to your API.

API Abuse

To keep from being flagged for API abuse, avoid triggering excessive HTTP 429 responses. Rather than waiting for an HTTP 429 response, limit your use of the API using the header values in the HTTP 200 responses, as detailed in the **Tips to Avoid Being Rate Limited** and **Examples** section of this document. Excessive HTTP 429 responses may result in your access to our API being temporarily or permanently disabled for your subdomain. Please note that you won't actually receive these HTTP 429 responses until **Monday, August 20, 2018** (MDT), but you can trigger them now using the new `rate_limit_test` API Service described above.

Avoid the temptation to create requests for larger amounts of data at once to avoid being rate limited. Requesting larger amounts of data per API call may negatively impact access to your data both through the API and your Church Community Builder site. Our new rate limiting rules won't reduce your daily allotment of calls to our APIs, they'll just require you to spread your requests at a reasonable rate.