Heroku Error Codes

Last updated 09 March 2020

Table of Contents

- R99 - Platform error
- L10 - Drain buffer overflow
- L11 - Tail buffer overflow
- L12 - Local buffer overflow
- L13 - Local delivery error
- L14 - Certificate validation error
- L15 - Tail buffer temporarily unavailable

Whenever your app experiences an error, Heroku will return a standard error page with the HTTP status code 503. To help you debug the underlying error, however, the platform will also add custom error information to your logs. Each type of error gets its own error code, with all HTTP errors starting with the letter H and all runtime errors starting with R. Logging errors start with L.

H10 - App crashed

A crashed web dyno or a boot timeout on the web dyno will present this error.

```
2010-10-06T21:51:04-07:00 heroku[web.1]: State changed from down to starting
2010-10-06T21:51:07-07:00 app[web.1]: Starting process with command: `bundle exec rails s
erver -p 22020`
2010-10-06T21:51:09-07:00 app[web.1]: >> Using rails adapter
2010-10-06T21:51:09-07:00 app[web.1]: Missing the Rails 2.3.5 gem. Please `gem install -v
=2.3.5 rails`, update your RAILS_GEM_VERSION setting in config/environment.rb for the Rai
ls version you do have installed, or comment out RAILS_GEM_VERSION to use the latest vers
ion installed.
2010-10-06T21:51:10-07:00 heroku[web.1]: Process exited
2010-10-06T21:51:12-07:00 heroku[router]: at=error code=H10 desc="App crashed" method=GET
path="/" host=myapp.herokuapp.com fwd=17.17.17.17 dyno= connect= service= status=503 byte
s=
```

H11 - Backlog too deep

When HTTP requests arrive faster than your application can process them, they can form a large backlog on a number of routers. When the backlog on a particular router passes a threshold, the router determines that your application isn't keeping up with its incoming request volume. You'll see an H11 error for each incoming request as long as the backlog is over this size. The exact value of this threshold may change depending on various factors, such as the number of dynos in your app, response time for individual requests, and your app's normal request volume.

```
2010-10-06T21:51:07-07:00 heroku[router]: at=error code=H11 desc="Backlog too deep" metho
d=GET path="/" host=myapp.herokuapp.com fwd=17.17.17.17 dyno= connect= service= status=50
3 bytes=
```

The solution is to increase your app's throughput by adding more dynos, tuning your database (for example, adding an index), or making the code itself faster. As always, increasing performance is highly application-specific and requires profiling.

H12 - Request timeout

For more information on request timeouts (including recommendations for resolving them), take a look at **our article on the topic (https://devcenter.heroku.com/articles/request-timeout)**.

An HTTP request took longer than 30 seconds (https://devcenter.heroku.com/articles/request-timeout) to complete. In the example below, a Rails app takes 37 seconds to render the page; the HTTP router returns a 503 prior to Rails completing its request cycle, but the Rails process continues and the completion message shows after the router message.

```
2010-10-06T21:51:07-07:00 app[web.2]: Processing PostController#list (for 75.36.147.245 a
t 2010-10-06 21:51:07) [GET]
2010-10-06T21:51:08-07:00 app[web.2]: Rendering template within layouts/application
2010-10-06T21:51:19-07:00 app[web.2]: Rendering post/list
2010-10-06T21:51:37-07:00 heroku[router]: at=error code=H12 desc="Request timeout" method
=GET path="/" host=myapp.herokuapp.com fwd=17.17.17.17 dyno=web.1 connect=6ms service=300
01ms status=503 bytes=0
2010-10-06T21:51:42-07:00 app[web.2]: Completed in 37000ms (View: 27, DB: 21) | 200 OK [h
ttp://myapp.heroku.com/]
```

This 30-second limit is measured by the router, and includes all time spent in the dyno, including the kernel's incoming connection queue and the app itself.

See Request Timeout (https://devcenter.heroku.com/articles/request-timeout) for more, as well as a language-specific article on this error:

- H12 - Request Timeout in Ruby (MRI) (https://devcenter.heroku.com/articles/h12-request-timeout-in-ruby-mri)

H13 - Connection closed without response

This error is thrown when a process in your web dyno accepts a connection, but then closes the socket without writing anything to it.

```
2010-10-06T21:51:37-07:00 heroku[router]: at=error code=H13 desc="Connection closed witho
ut response" method=GET path="/" host=myapp.herokuapp.com fwd=17.17.17.17 dyno=web.1 conn
ect=3030ms service=9767ms status=503 bytes=0
```

One example where this might happen is when a Unicorn web server (https://devcenter.heroku.com/articles/rails-unicorn) is configured with a timeout shorter than 30s and a request has not been processed by a worker before the timeout happens. In this case, Unicorn closes the connection before any data is written, resulting in an H13.

H14 - No web dynos running

This is most likely the result of scaling your web dynos down to 0 dynos. To fix it, scale your web dynos to 1 or more dynos:

```
$ heroku ps:scale web=1
```

Use the `heroku ps` command to determine the state of your web dynos.

```
2010-10-06T21:51:37-07:00 heroku[router]: at=error code=H14 desc="No web processes runnin
g" method=GET path="/" host=myapp.herokuapp.com fwd=17.17.17.17 dyno= connect= service= s
tatus=503 bytes=
```

H15 - Idle connection

The dyno did not send a full response and was terminated due to 55 seconds of inactivity (https://devcenter.heroku.com/articles/request-timeout). For example, the response indicated a `Content-Length` of 50 bytes which were not sent in time.

```
2010-10-06T21:51:37-07:00 heroku[router]: at=error code=H15 desc="Idle connection" method
=GET path="/" host=myapp.herokuapp.com fwd=17.17.17.17 dyno=web.1 connect=1ms service=554
49ms status=503 bytes=18
```

H16 - Redirect to herokuapp.com

Apps on Cedar's HTTP routing stack (https://devcenter.heroku.com/articles/http-routing) use the herokuapp.com domain. Requests made to a Cedar app at its deprecated heroku.com domain will be redirected to the correct herokuapp.com address and this redirect message will be inserted into the app's logs.

```
2010-10-06T21:51:37-07:00 heroku[router]: at=info code=H16 desc="herokuapp redirect" meth
od=GET path="/" host=myapp.herokuapp.com fwd=17.17.17.17 dyno= connect= service= status=3
01 bytes=
```

H17 - Poorly formatted HTTP response

Our HTTP routing stack has recently changed and no longer accepts responses that are missing a reason phrase in the **status line (http://tools.ietf.org/html/rfc7230#section-3.1.2)**. 'HTTP/1.1 200 OK' will work with the new router, but 'HTTP/1.1 200' will not.

This error message is logged when a router detects a malformed HTTP response coming from a dyno.

```
2010-10-06T21:51:37-07:00 heroku[router]: at=error code=H17 desc="Poorly formatted HTTP r
esponse" method=GET path="/" host=myapp.herokuapp.com fwd=17.17.17.17 dyno=web.1 connect=
1ms service=1ms status=503 bytes=0
```

H18 - Server Request Interrupted

The backend socket, belonging to your app's web process was closed before the backend returned an HTTP response.

```
2010-10-06T21:51:37-07:00 heroku[router]: sock=backend at=error code=H18 desc="Server Req
uest Interrupted" method=GET path="/" host=myapp.herokuapp.com fwd=17.17.17.17 dyno=web.1
connect=1ms service=1ms status=503 bytes=0
```

H19 - Backend connection timeout

A router received a connection timeout error after 5 seconds attempting to open a socket to a web dyno. This is usually a symptom of your app being overwhelmed and failing to accept new connections in a timely manner. If you have multiple dynos, the router will retry multiple dynos before logging H19 and serving a standard error page.

If your app has a single web dyno, it is possible to see H19 errors if the runtime instance running your web dyno fails and is replaced. Once the failure is detected and the instance is terminated your web dyno will be restarted somewhere else, but in the meantime, H19s may be served as the router fails to establish a connection to your dyno. This can be mitigated by running more than one web dyno.

```
2010-10-06T21:51:07-07:00 heroku[router]: at=error code=H19 desc="Backend connection time
out" method=GET path="/" host=myapp.herokuapp.com fwd=17.17.17.17 dyno=web.1 connect=5001
ms service= status=503 bytes=
```

H20 - App boot timeout

The router will enqueue requests for 75 seconds while waiting for starting processes to reach an “up” state. If after 75 seconds, no web dynos have reached an “up” state, the router logs H20 and serves a standard error page.

```
2010-10-06T21:51:07-07:00 heroku[router]: at=error code=H20 desc="App boot timeout" metho
d=GET path="/" host=myapp.herokuapp.com fwd=17.17.17.17 dyno= connect= service= status=50
3 bytes=
```

The Ruby on Rails **asset pipeline (http://guides.rubyonrails.org/asset_pipeline.html)** can sometimes fail to run during **git push (https://devcenter.heroku.com/articles/git)**, and will instead attempt to run when your app’s **dynos (https://devcenter.heroku.com/articles/dynos)** boot. Since the Rails asset pipeline is a slow process, this can cause H20 boot timeout errors.

This error differs from R10 in that the H20 75-second timeout includes platform tasks such as internal state propagation, requests between internal components, slug download, unpacking, container preparation, etc… The R10 60-second timeout applies solely to application startup tasks.

If your application requires more time to boot, you may use the **boot timeout tool (https://tools.heroku.support/limits/boot_timeout)** to increase the limit. However, in general, slow boot times will make it harder to deploy your application and will make recovery from dyno failures slower, so this should be considered a *temporary solution*.

H21 - Backend connection refused

A router received a connection refused error when attempting to open a socket to your web process. This is usually a symptom of your app being overwhelmed and failing to accept new connections. If you have multiple dynos, the router will retry multiple dynos before logging H21 and serving a standard error page.

```
2010-10-06T21:51:07-07:00 heroku[router]: at=error code=H21 desc="Backend connection refu
sed" method=GET path="/" host=myapp.herokuapp.com fwd=17.17.17.17 dyno=web.1 connect=1ms
service= status=503 bytes=
```

H22 - Connection limit reached

A routing node has detected an elevated number of HTTP client connections attempting to reach your app. Reaching this threshold most likely means your app is under heavy load and is not responding quickly enough to keep up. The exact value of this threshold may change depending on various factors, such as the number of dynos in your app, response time for individual requests, and your app’s normal request volume.

```
2010-10-06T21:51:07-07:00 heroku[router]: at=error code=H22 desc="Connection limit reache
d" method=GET path="/" host=myapp.herokuapp.com fwd=17.17.17.17 dyno= connect= service= s
tatus=503 bytes=
```

H23 - Endpoint misconfigured

A routing node has detected a websocket handshake (https://tools.ietf.org/html/rfc6455#section-1.3), specifically the 'Sec-Websocket-Version’ header in the request, that came from an endpoint (upstream proxy) that does not support websockets.

```
2010-10-06T21:51:07-07:00 heroku[router]: at=error code=H23 desc="Endpoint misconfigured"
method=GET path="/" host=myapp.herokuapp.com fwd=17.17.17.17 dyno= connect= service= stat
us=503 bytes=
```

H24 - Forced close

The routing node serving this request was either shutdown for maintenance or terminated before the request completed.

```
2010-10-06T21:51:07-07:00 heroku[router]: at=error code=H24 desc="Forced close" method=GE
T path="/" host=myapp.herokuapp.com fwd=17.17.17.17 dyno=web.1 connect=1ms service=80000m
s status= bytes=18
```

H25 - HTTP Restriction

This error is logged when a routing node detects and blocks a valid HTTP response that is judged risky or too large to be safely parsed. The error comes in four types.

Currently, this functionality is experimental, and is only made available to a subset of applications on the platform.

Invalid content length

The response has multiple content lengths declared within the same response, with varying lengths.

```
2014-03-20T14:22:00.203382+00:00 heroku[router]: at=error code=H25 desc="HTTP restrictio
n: invalid content length" method=GET path="/" host=myapp.herokuapp.com request_id=3f336f
1a-9be3-4791-afe3-596a1f2a481f fwd="17.17.17.17" dyno=web.1 connect=0 service=1 status=50
2 bytes=537
```

Oversized cookies

The cookie in the response will be too large to be used again in a request to the Heroku router or SSL endpoints.

```
2014-03-20T14:18:57.403882+00:00 heroku[router]: at=error code=H25 desc="HTTP restrictio
n: oversized cookie" method=GET path="/" host=myapp.herokuapp.com request_id=90cfbbd2-039
7-4bab-828f-193050a076c4 fwd="17.17.17.17" dyno=web.1 connect=0 service=2 status=502 byte
s=537
```

Oversized header

A single header line is deemed too long (over 512kb) and the response is discarded on purpose.

```
2014-03-20T14:12:28.555073+00:00 heroku[router]: at=error code=H25 desc="HTTP restrictio
n: oversized header" method=GET path="/" host=myapp.herokuapp.com request_id=ab66646e-84e
b-47b8-b3bb-2031ecc1bc2c fwd="17.17.17.17" dyno=web.1 connect=0 service=397 status=502 by
tes=542
```

Oversized status line

The status line is judged too long (8kb) and the response is discarded on purpose.

```
2014-03-20T13:54:44.423083+00:00 heroku[router]: at=error code=H25 desc="HTTP restrictio
n: oversized status line" method=GET path="/" host=myapp.herokuapp.com request_id=208588a
c-1a66-44c1-b665-fe60c596241b fwd="17.17.17.17" dyno=web.1 connect=0 service=3 status=502
bytes=537
```

H26 - Request Error

This error is logged when a request has been identified as belonging to a specific Heroku application, but cannot be delivered entirely to a dyno due to HTTP protocol errors in the request. Multiple possible causes can be identified in the log message.

Unsupported expect header value

The request has an `expect` header, and its value is not `100-Continue`, the only expect value handled by the router. A request with an unsupported `expect` value is terminated with the status code `417 Expectation Failed`.

```
2014-05-14T17:17:37.456997+00:00 heroku[router]: at=error code=H26 desc="Request Error" c
ause="unsupported expect header value" method=GET path="/" host=myapp.herokuapp.com reque
st_id=3f336f1a-9be3-4791-afe3-596a1f2a481f fwd="17.17.17.17" dyno= connect= service= stat
us=417 bytes=
```

Bad header

The request has an HTTP header with a value that is either impossible to parse, or not handled by the router, such as `connection: ,`.

```
2014-05-14T17:17:37.456997+00:00 heroku[router]: at=error code=H26 desc="Request Error" c
ause="bad header" method=GET path="/" host=myapp.herokuapp.com request_id=3f336f1a-9be3-4
791-afe3-596a1f2a481f fwd="17.17.17.17" dyno= connect= service= status=400 bytes=
```

Bad chunk

The request has a chunked transfer-encoding, but with a chunk that was invalid or couldn’t be parsed correctly. A request with this status code will be interrupted during transfer to the dyno.

```
2014-05-14T17:17:37.456997+00:00 heroku[router]: at=error code=H26 desc="Request Error" c
ause="bad chunk" method=GET path="/" host=myapp.herokuapp.com request_id=3f336f1a-9be3-47
91-afe3-596a1f2a481f fwd="17.17.17.17" dyno=web.1 connect=1 service=0 status=400 bytes=53
7
```

H27 - Client Request Interrupted

The client socket was closed either in the middle of the request or before a response could be returned. For example, the client closed their browser session before the request was able to complete.

```
2010-10-06T21:51:37-07:00 heroku[router]: sock=client at=warning code=H27 desc="Client Re
quest Interrupted" method=POST path="/submit/" host=myapp.herokuapp.com fwd=17.17.17.17 d
yno=web.1 connect=1ms service=0ms status=499 bytes=0
```

H28 - Client Connection Idle

The client did not send a full request and was terminated due to 55 seconds of inactivity (https://devcenter.heroku.com/articles/request-timeout). For example, the client indicated a `Content-Length` of 50 bytes which were not sent in time.

```
2010-10-06T21:51:37-07:00 heroku[router]: at=warning code=H28 desc="Client Connection Idl
e" method=GET path="/" host=myapp.herokuapp.com fwd=17.17.17.17 dyno=web.1 connect=1ms se
rvice=55449ms status=499 bytes=18
```

H80 - Maintenance mode

This is not an error, but we give it a code for the sake of completeness. Note the log formatting is the same but without the word “Error”.

```
2010-10-06T21:51:07-07:00 heroku[router]: at=info code=H80 desc="Maintenance mode" method
=GET path="/" host=myapp.herokuapp.com fwd=17.17.17.17 dyno= connect= service= status=503
bytes=
```

H81 - Blank app

No code has been pushed to this application. To get rid of this message you need to do one deploy (https://devcenter.heroku.com/articles/git#deploying-code). This is not an error, but we give it a code for the sake of completeness.

```
2010-10-06T21:51:07-07:00 heroku[router]: at=info code=H81 desc="Blank app" method=GET pa
th="/" host=myapp.herokuapp.com fwd=17.17.17.17 dyno= connect= service= status=503 bytes=
```

H82 - Free dyno quota exhausted

This indicates that an account’s free dyno hour quota (https://devcenter.heroku.com/articles/free-dyno-hours) is exhausted and that apps running free dynos are sleeping. You can view your app’s free dyno usage in the Heroku dashboard (https://dashboard.heroku.com/account/billing).

```
2015-10-06T21:51:07-07:00 heroku[router]: at=info code=H82 desc="Free dyno quota exhausted"
```

H99 - Platform error

H99 and R99 are the only error codes that represent errors in the Heroku platform.

This indicates an internal error in the Heroku platform. Unlike all of the other errors which will require action from you to correct, this one does not require action from you. Try again in a minute, or check the status site (http://status.heroku.com/).

```
2010-10-06T21:51:07-07:00 heroku[router]: at=error code=H99 desc="Platform error" method=
GET path="/" host=myapp.herokuapp.com fwd=17.17.17.17 dyno= connect= service= status=503
bytes=
```

R10 - Boot timeout

A web process took longer than 60 seconds to bind to its assigned `$PORT`. When this happens, the dyno's process is killed and the dyno is considered crashed. Crashed dynos are restarted according to the dyno manager's restart policy (https://devcenter.heroku.com/articles/dynos#automatic-dyno-restarts).

```
2011-05-03T17:31:38+00:00 heroku[web.1]: State changed from created to starting
2011-05-03T17:31:40+00:00 heroku[web.1]: Starting process with command: `bundle exec rail
s server -p 22020 -e production`
2011-05-03T17:32:40+00:00 heroku[web.1]: Error R10 (Boot timeout) -> Web process failed t
o bind to $PORT within 60 seconds of launch
2011-05-03T17:32:40+00:00 heroku[web.1]: Stopping process with SIGKILL
2011-05-03T17:32:40+00:00 heroku[web.1]: Process exited
2011-05-03T17:32:41+00:00 heroku[web.1]: State changed from starting to crashed
```

This error is often caused by a process being unable to reach an external resource, such as a database, or the application doing too much work, such as parsing and evaluating numerous, large code dependencies, during startup.

Common solutions are to access external resources asynchronously, so they don't block startup, and to reduce the amount of application code or its dependencies.

If your application requires more time to boot, you may use the **boot timeout tool (https://tools.heroku.support/limits/boot_timeout)** to increase the limit. However, in general, slow boot times will make it harder to deploy your application and will make recovery from dyno failures slower, so this should be considered a *temporary solution*.

One exception is for apps using the Java buildpack (https://devcenter.heroku.com/articles/java-support), Gradle buildpack (https://devcenter.heroku.com/articles/deploying-gradle-apps-on-heroku), heroku-deploy toolbelt plugin (https://devcenter.heroku.com/articles/war-deployment#deployment-with-the-heroku-cli), or Heroku Maven plugin (https://devcenter.heroku.com/articles/deploying-java-applications-with-the-heroku-maven-plugin), which will be allowed 90 seconds to bind to their assigned port.

R12 - Exit timeout

A process failed to exit within 30 seconds of being sent a SIGTERM indicating that it should stop. The process is sent SIGKILL to force an exit.

```
2011-05-03T17:40:10+00:00 app[worker.1]: Working
2011-05-03T17:40:11+00:00 heroku[worker.1]: Stopping process with SIGTERM
2011-05-03T17:40:11+00:00 app[worker.1]: Ignoring SIGTERM
2011-05-03T17:40:14+00:00 app[worker.1]: Working
2011-05-03T17:40:18+00:00 app[worker.1]: Working
2011-05-03T17:40:21+00:00 heroku[worker.1]: Error R12 (Exit timeout) -> Process failed to
exit within 30 seconds of SIGTERM
2011-05-03T17:40:21+00:00 heroku[worker.1]: Stopping process with SIGKILL
2011-05-03T17:40:21+00:00 heroku[worker.1]: Process exited
```

R13 - Attach error

A dyno started with `heroku run` failed to attach to the invoking client.

```
2011-06-29T02:13:29+00:00 app[run.3]: Awaiting client
2011-06-29T02:13:30+00:00 heroku[run.3]: State changed from starting to up
2011-06-29T02:13:59+00:00 app[run.3]: Error R13 (Attach error) -> Failed to attach to pro
cess
2011-06-29T02:13:59+00:00 heroku[run.3]: Process exited
```

R14 - Memory quota exceeded

A dyno requires memory in excess of its quota (https://devcenter.heroku.com/articles/dynos#memory-behavior). If this error occurs, the dyno will page to swap space (http://en.wikipedia.org/wiki/Paging) to continue running, which may cause degraded process performance. The R14 error is calculated by total memory swap, rss and cache.

```
2011-05-03T17:40:10+00:00 app[worker.1]: Working
2011-05-03T17:40:10+00:00 heroku[worker.1]: Process running mem=1028MB(103.3%)
2011-05-03T17:40:11+00:00 heroku[worker.1]: Error R14 (Memory quota exceeded)
2011-05-03T17:41:52+00:00 app[worker.1]: Working
```

If you are getting a large number of R14 errors, your application performance is likely severely degraded. Resolving R14 memory errors are language specific:

- R14 - Memory Quota Exceeded in Ruby (MRI) (https://devcenter.heroku.com/articles/ruby-memory-use)
- Troubleshooting Memory Issues in Java Applications (https://devcenter.heroku.com/articles/java-memory-issues)
- Troubleshooting Node.js Memory Use (https://devcenter.heroku.com/articles/node-memory-use)

R15 - Memory quota vastly exceeded

A dyno requires vastly more memory than its quota (https://devcenter.heroku.com/articles/dynos#memory-behavior) and is consuming excessive swap space. If this error occurs, the dyno will be killed by the platform. The R15 error is calculated by total memory swap and rss; cache is not included.

```
2011-05-03T17:40:10+00:00 app[worker.1]: Working
2011-05-03T17:40:10+00:00 heroku[worker.1]: Process running mem=1029MB(201.0%)
2011-05-03T17:40:11+00:00 heroku[worker.1]: Error R15 (Memory quota vastly exceeded)
2011-05-03T17:40:11+00:00 heroku[worker.1]: Stopping process with SIGKILL
2011-05-03T17:40:12+00:00 heroku[worker.1]: Process exited
```

In Private Spaces, dynos vastly exceeding their memory quota emit R15 errors, but do not use swap space. Instead, the platform kills processes consuming large amounts of memory, but may not kill the dyno itself.

R16 - Detached

An attached dyno is continuing to run after being sent `SIGHUP` when its external connection was closed. This is usually a mistake, though some apps might want to do this intentionally.

```
2011-05-03T17:32:03+00:00 heroku[run.1]: Awaiting client
2011-05-03T17:32:03+00:00 heroku[run.1]: Starting process with command `bash`
2011-05-03T17:40:11+00:00 heroku[run.1]: Client connection closed. Sending SIGHUP to all
processes
2011-05-03T17:40:16+00:00 heroku[run.1]: Client connection closed. Sending SIGHUP to all
processes
2011-05-03T17:40:21+00:00 heroku[run.1]: Client connection closed. Sending SIGHUP to all
processes
2011-05-03T17:40:26+00:00 heroku[run.1]: Error R16 (Detached) -> An attached process is n
ot responding to SIGHUP after its external connection was closed.
```

R17 - Checksum error

This indicates an error with runtime slug checksum (https://devcenter.heroku.com/articles/slug-checksums) verification. If the checksum does not match or there is another problem with the checksum when launch a dyno, an R17 error will occur and the dyno will fail to launch. Check the log stream for details about the error.

```
2016-08-16T12:39:56.439438+00:00 heroku[web.1]: State changed from provisioning to starti
ng
2016-08-16T12:39:57.110759+00:00 heroku[web.1]: Error R17 (Checksum error) -> Checksum do
es match expected value. Expected: SHA256:ed5718e83475c780145609cbb2e4f77ec8076f6f59ebc8a
916fb790fbdb1ae64 Actual: SHA256:9ca15af16e06625dfd123ebc3472afb0c5091645512b31ac3dd355f0
d8cc42c1
2016-08-16T12:39:57.212053+00:00 heroku[web.1]: State changed from starting to crashed
```

If this error occurs, try deploying a new release with a correct checksum or rolling back to an older release. Ensure the checksum is formatted and calculated correctly (https://devcenter.heroku.com/articles/slug-checksums) with the SHA256 algorithm. The checksum must start with `SHA256:` followed by the calculated SHA256 value for the compressed slug. If you did not manually calculate the checksum and error continues to occur, please contact Heroku support.

R99 - Platform error

R99 and H99 are the only error codes that represent errors in the Heroku platform.

This indicates an internal error in the Heroku platform. Unlike all of the other errors which will require action from you to correct, this one does not require action from you. Try again in a minute, or check the status site (http://status.heroku.com/).

L10 - Drain buffer overflow

```
2013-04-17T19:04:46+00:00 d.1234-drain-identifier-567 heroku logplex - - Error L10 (outpu
t buffer overflow): 500 messages dropped since 2013-04-17T19:04:46+00:00.
```

The number of log messages being generated has temporarily exceeded the rate at which they can be received by a drain consumer (such as a log management add-on) and Logplex, Heroku's logging system (https://devcenter.heroku.com/articles/logging), has discarded some messages in order to handle the rate difference.

A common cause of L10 error messages is the exhaustion of capacity in a log consumer. If a log management add-on or similar system can only accept so many messages per time period, your application may experience L10s after crossing that threshold.

Another common cause of L10 error messages is a sudden burst of log messages from a dyno. As each line of dyno output (e.g. a line of a stack trace) is a single log message, and Logplex limits the total number of un-transmitted log messages it will keep in memory to 1024 messages, a burst of lines from a dyno can overflow buffers in Logplex. In order to allow the log stream to catch up, Logplex will discard messages where necessary, keeping newer messages in favor of older ones.

You may need to investigate reducing the volume of log lines output by your application (e.g. condense multiple logs lines into a smaller, single-line entry). You can also use the `heroku logs -t` command to get a live feed of logs and find out where your problem might be. A single dyno stuck in a loop that generates log messages can force an L10 error, as can a problematic code path that causes all dynos to generate a multi-line stack trace for some code paths.

L11 - Tail buffer overflow

A heroku logs –tail session cannot keep up with the volume of logs generated by the application or log channel, and Logplex has discarded some log lines necessary to catch up. To avoid this error you will need run the command on a faster internet connection (increase the rate at which you can receive logs) or you will need to modify your application to reduce the logging volume (decrease the rate at which logs are generated).

```
2011-05-03T17:40:10+00:00 heroku[logplex]: L11 (Tail buffer overflow) -> This tail sessio
n dropped 1101 messages since 2011-05-03T17:35:00+00:00
```

L12 - Local buffer overflow

The application is producing logs faster than the local delivery process (log-shuttle) can deliver them to logplex and has discarded some log lines in order to keep up. If this error is sustained you will need to reduce the logging volume of your application.

```
2013-11-04T21:31:32.125756+00:00 app[log-shuttle]: Error L12: 222 messages dropped since
2013-11-04T21:31:32.125756+00:00.
```

L13 - Local delivery error

The local log delivery process (log-shuttle) was unable to deliver some logs to Logplex and has discarded them. This can happen during transient network errors or during logplex service degradation. If this error is sustained please contact support.

```
2013-11-04T21:31:32.125756+00:00 app[log-shuttle]: Error L13: 111 messages lost since 201
3-11-04T21:31:32.125756+00:00.
```

L14 - Certificate validation error

The application is configured with a TLS syslog drain that doesn’t have a valid TLS certificate.

You should check that:

1. You’re not using a self-signed certificate.
2. The certificate is up to date.
3. The certificate is signed by a known and trusted CA.
4. The CN hostname embedded in the certificate matches the hostname being connected to.

```
2015-09-04T23:28:48+00:00 heroku[logplex]: Error L14 (certificate validation): error="bad
certificate" uri="syslog+tls://logs.example.com:6514/"
```

L15 - Tail buffer temporarily unavailable

The tail buffer that stores the last 1500 lines of your logs is temporarily unavailable. Run `heroku logs` again. If you still encounter the error, run `heroku logs -t` to stream your logs (which does not use the tail buffer).