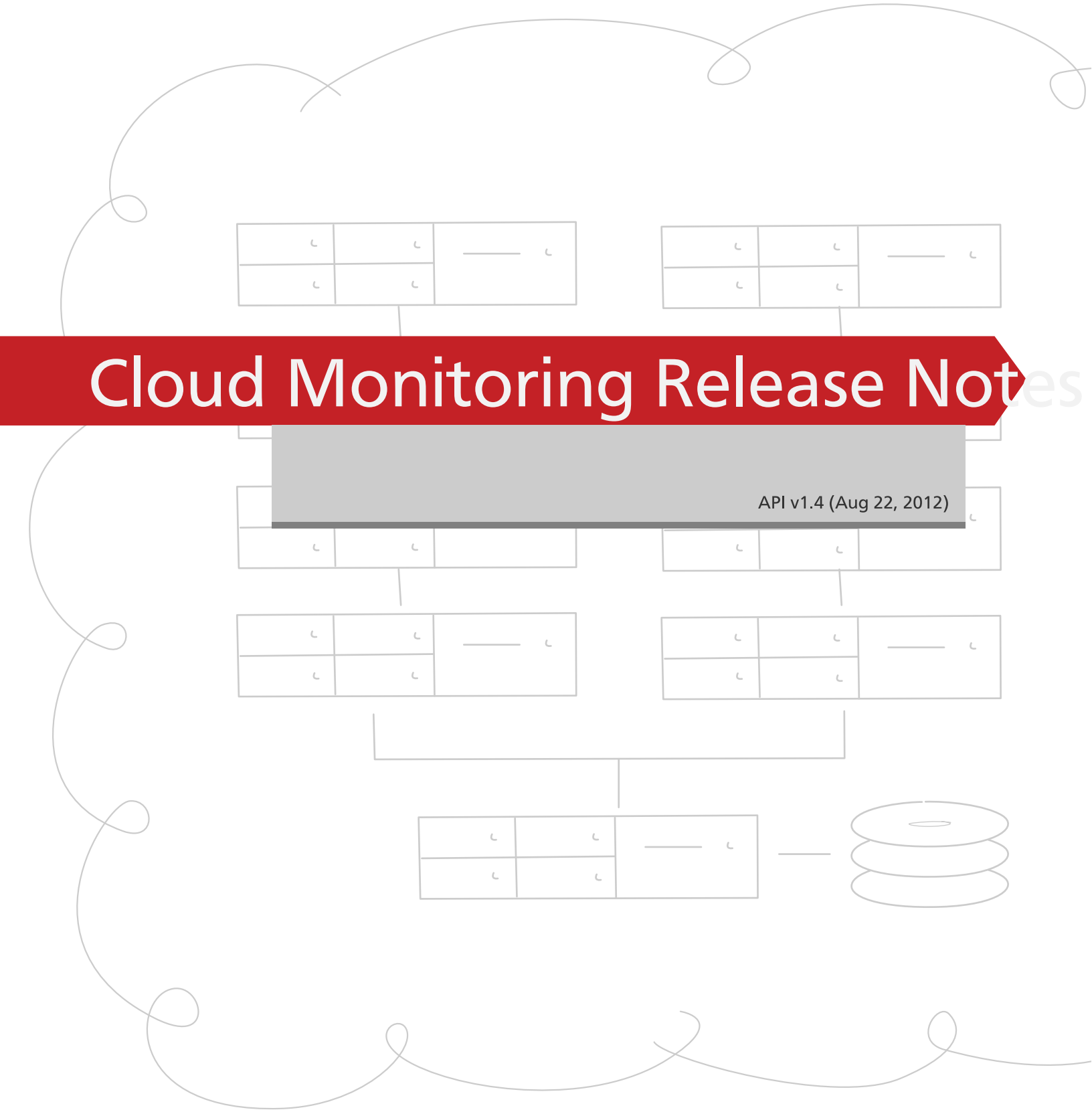

Cloud Monitoring Release Notes

API v1.4 (Aug 22, 2012)

rackspace HOSTING

docs.rackspace.com/api

Cloud Monitoring Release Notes

API v1.4 (2012-08-22)

This document is intended for software developers interested in developing applications using the Rackspace Cloud Monitoring API. The document is for informational purposes only and is provided "AS IS."

v1.4, August 22, 2012

These release notes correspond to the "Unlimited Availability" release of Cloud Monitoring.

New Features

- *More HTTP Options:* Pass in custom headers for a `remote.http` check. Depending on the site you're testing, this can be crucial in determining if your site is up.

  remote.http Docs

- *Add a previous operator to the Alarm Language:* This allows you to look at a metric value in the previous time period. It is very useful for detecting changes. For example, to detect an SSH fingerprint change, you'd construct an Alarm snippet like this:

```
if (metrics['fingerprint'] != previous(metrics['fingerprint'])) {
  return CRITICAL, "SSH fingerprint changed to #{fingerprint}"
}
```

  Alarm DSL Docs

- *Improved email alerts:* Included in emails are the observations of the particular check from each monitoring zone. This allows you to get visibility on the status from any of the included monitoring zones of a check. We've also reformatted the message to look more user friendly.

  Subject:

```
** CRITICAL: host server check on web1.domain.com **
```

  Body:

```
================== Rackspace Cloud Monitoring Notification ================
==
Entity: web1.domain.com (50.56.179.42)
Check: host server check (remote.ping)
Alarm: Ping packet loss
Status: CRITICAL
Date: Mon Aug 13 2012 15:18:59 GMT+0000 (UTC)

--------------------------- Observations
 -------------------------------------

CRITICAL - Packet loss is 20%
London (LON), 25 sec ago

OK - Packet loss is normal
London (LON), 10 sec ago

OK - Packet loss is normal
London (LON), 56 sec ago

------------------------------------------------------------------------------

This is an automated Cloud Monitoring notification. You received this email
because you are listed as a notification recipient.
```

- *Retrieve overview by entityId or uri:* Include query strings *`uri`* or *`entityId`* to retrieve specific mapped Rackspace Cloud Servers (for now). You can also pull down a set of entities by referencing their id's. Keep in mind you cannot combine pulling both *`uri`*'s and *`entityId`*'s in the same request, but you can have up to 100 of them in a single request.

  Overview list by *`entityId`* and Overview list by *`uri`*.

Enhancements

- Added friendly names for the alarm_examples API.
- Add a "`metadata`" attribute to the alarm API.
- Allow including no conditionals in the alarm language, allowing you to set global filters, and rely on the check availability to determine the returned state.

Resolved Issues

- Fix un-setting optional attributes on an update (PUT).
- Fix the `check_type` -> `alarm` mapping for delivering inconsistent and non-deterministic updates.
- Fixed how we handle zlib streams when inflating a gzip http stream.
- When we combined two services we had briefly did not insert the initial `OK` state for the `alarm change log`, this is fixed now.
- Fixed updating the `target_alias` after setting the `target_hostname` on the initial creation, and vice-versa.

Internal Changes

New Features

- Deployed a staging Graylog2 cluster on the gold config.
- Consecutive count is now working in production, it is not highlighted due to it's lack of stability across restarts.

Enhancements

- Open many connections to the Cassandra server to up performance.
- Added 3 extra monitoring zones to staging.
- Added 99.9% percentile calculation to our internal metrics system.
- Deployed data warehouser analytics for the EBI team.
- Changed subject on Dead Letter emails so we can eventually add them to their ticket queue.

Resolved Issues

- Alarm/Check syncing issue discovered after better understanding the update process, fixed.
- Added code to log when the job runner fails.
- Use customer service API cache when it fails to retrieve on outbound message.
- Added the 3 extra monitoring zones to staging.

Documentation

- Try some simple monitoring exercises in the *Rackspace Cloud Monitoring Getting Started Guide* at:

  http://docs.rackspace.com/cm/api/v1.0/cm-getting-started/content/Introduction.html.

- Get reference information and examples for all endpoints in the *Rackspace Cloud Monitoring Developers Guide* at:

  http://docs.rackspace.com/cm/api/v1.0/cm-devguide/content/index.html.

Talk to Us

Do you have questions about Rackspace Cloud Monitoring? Would you like to give us feedback on how we're doing?

- Join us in our chat room at: Freenode IRC at #cloudmonitoring

  Or just click the following link:

  Webchat

- You can also provide feedback using our feedback form:

  Product Feedback Forum

- Get support here:

  File a ticket here

v1.3, June 14, 2012

These release notes correspond to the pre "Unlimited Availability" release of Cloud Monitoring.

New Features

- *Traceroute API:* This API lets you run a Traceroute from a Monitoring Zone to a Hostname or IP address. Like all Cloud Monitoring features, the Traceroute API is fully dual stack, supporting both IPv4 and IPv6. The Traceroute API can be used to debug networking issues between the Cloud Monitoring collectors and your infrastructure.

  Traceroute Docs

- *Metric interpolation on the Alarm status string:* This allows you to include metrics from a Check in the status string returned by an Alarm. For example, to include the HTTP status code from a `remote.http` Check, your Alarm could look like this:

```
if (metrics['code'] != "200") {
 return CRITICAL, "Bad HTTP Status: #{code}"
}
```

  Alarm DSL Docs

- *Add payload to the remote.http check:* This allows you send a request body during a HTTP/HTTPS request.

  remote.http attributes

Enhancements

- Allow a user to specify the number of ICMP packets to send out on the `remote.ping` check.
- Updated Certificate Authority chain used to validate SSL certificates, resulting in less false positives.
- `remote.tcp` check now features the `body_match_metric` when specifying the send_body command.
- Add the remote IP address of the collector that alerted you to the Webhook notification payload.

Resolved Issues

- Fixed writable date and time fields that were supposed to be immutable.
- Fixed the DSL string escaping to allow you to specify special characters, example below:

```
if (metric['code'] nregex '2\\d\\d') {
  return CRITICAL, 'Invalid status code #{code}'
}
```

Internal Changes

New Features

- Added account level feature flags, to enable easier dark launching of new features.
- Usage is being emitted to Atom Hoppper and is in a done/done state.
- Publishing internal docs to the docs-internal site.
- Validate Alarm DSL regexes before storing them. This will allow us to make sure that we are storing completely valid regexes.
- Add an internal account dump tool that will dump everything about your account.
- Deployed the Host Agent into production.

Enhancements

- Allow an Agent to be connected and idle, this will allow you to run commands on agents not associated with an entity.
- Factored out a shared utilities library for internal Rackspace consumption.
- Deployed the "Agent Endpoint" (agent infrastructure for talking to the agent) in production and staging.
- Allow Hybrid Tenant Ids.
- Publishing agent protocol docs on the docs-internal website. This will allow people to get an early look at the flexibility and extensibility of the agent.
- Added profiling to the dashboard. Click a button on the operational dashboard and get access to hierarchical stack information and timing.
- Deployed a new Graylog2/Elasticsearch cluster with 5 TB of usable space.

  View here: https://iad2-maas-prod-glog0.cm.k1k.me

Resolved Issues

- Limit the number of body matches.
- Fixed critical ZooKeeper bug that prevented reconnects in rare network partitions.
- Switch to reading and writing with `QUORUM` majority. Most writes will get an added benefit of speed.
- Implemented Proper XML escaping from API server to the Collector (noitd).

Documentation

- Try some simple monitoring exercises in the *Rackspace Cloud Monitoring Getting Started Guide* at:

http://docs.rackspace.com/cm/api/v1.0/cm-getting-started/content/Introduction.html.

- Get reference information and examples for all endpoints in the *Rackspace Cloud Monitoring Developers Guide* at:

  http://docs.rackspace.com/cm/api/v1.0/cm-devguide/content/index.html.

Talk to Us

Do you have questions about Rackspace Cloud Monitoring? Would you like to give us feedback on how we're doing?

- Join us in our chat room at: Freenode IRC at #cloudmonitoring

  Or just click the following link:

  Webchat

- You can also provide feedback using our feedback form:

  Product Feedback Forum

- Get support here:

  File a ticket here

v1.2, May 10, 2012

These release notes correspond to the post-EAP release of Cloud Monitoring.

New Features

- **Test an Existing Check**

  Test execute an existing check. This allows you to execute a check like the traditional test check, but on a check that is already running. This will not actually execute the new check in the same check context, it executes in a new context.

- **Alarm Templates**

  Get a list of alarm recipes with templating capabilities. This helps kick-start your thinking on all the possible ways to leverage this incredibly powerful part of the monitoring system. The link to the API documentation is here:

  Alarm Examples API Documentation

- **Cascading Entity Deletion**

  Delete entities and now have it cascade to all of the associated checks and alarms in the system. This allows you to de-provision monitors quickly and easily without having to manually make the individual calls yourself. This saves you time and effort.

- **Type Checking on Alarms**

  Type check alarms so you receive alerts if you set up an incorrect alarm, or it isn't as expected. As a bonus the alarm also validates all the metrics you are checking are present, if they are not present, it automatically changes the alarm state to CRITICAL.

Enhancements

- Fixed a defect in the networking partitioning logic that leads to more consistent operation.
- Added more collector infrastructure in IAD.
- Added a meta-data field on the check type.

Resolved Issues

- Alarm states have a race condition. During replay it's possible to unintentionally store duplicates.
- Fix missing version parameter in next href.
- Fix banner match to be more consistent, be present but empty if it didn't match.
- Removed the ability to use duplicate monitoring zones.

Documentation

- Try some simple monitoring exercises in the *Rackspace Cloud Monitoring Getting Started Guide* at:

  http://docs.rackspace.com/cm/api/v1.0/cm-getting-started/content/Introduction.html.

- Get reference information and examples for all endpoints in the *Rackspace Cloud Monitoring Developers Guide* at:

  http://docs.rackspace.com/cm/api/v1.0/cm-devguide/content/index.html.

Talk to Us

Do you have questions about Rackspace Cloud Monitoring? Would you like to give us feedback on how we're doing?

- Join us in our chat room at: Freenode IRC at #cloudmonitoring

  Or just click the following link:

  Webchat

- You can also provide feedback using our feedback form:

  Product Feedback Forum

- Get support here:

  File a ticket here

v1.1, March 28, 2012

These release notes correspond to the EAP for Rackspace Cloud Monitoring.

New Features

- **Test Notifications**

  Test execute a notification. This makes it easy to execute test checks and verify that you'll receive the payload. We also added the ability to test existing notifications as well.

- **JSONP Support**

  Support for JSONP for retrieval in the API. This gives you the tools to access the API from the browser directly.

- **Rate Limiting Exposed**

  We now expose rate limiting and the time window of the rate limiting to you through headers. This helps us with transparency and lets you know when you're getting close to reaching your account rate limits.

- **Email Notifications**

  We added email as a notification type so you can receive email on your alerts.

Enhancements

- Renamed "Alarm History" to "Alarm Notification History."
- Now support a key-value hash to facilitate multiple body matches.
- Added previous state to webhook payload.
- Audits now expose the tenantId.
- Added HTTP CORS headers so you can now do cross domain requests.
- Unified the notification history log and the alarm change log.
- Improved support dealing with time ordered mixed results.
- Improved type checking for the alarm subsystem.
- Added detection for missing metrics and automatically output a failed state when detected.

Resolved Issues

- Fixed hashing method to prevent a collector affinity.
- Improved the location header returned so it works in non-standard port setups.
- Added previous state to the webhook field.

- Added a label to the alarm object.
- Unified the language around the different states in the system.

Documentation

- Try some simple monitoring exercises in the *Rackspace Cloud Monitoring Getting Started Guide* at:

  http://docs.rackspace.com/cm/api/v1.0/cm-getting-started/content/Introduction.html.

- Get reference information and examples for all endpoints in the *Rackspace Cloud Monitoring Developers Guide* at:

  http://docs.rackspace.com/cm/api/v1.0/cm-devguide/content/index.html.

Talk to Us

Do you have questions about Rackspace Cloud Monitoring? Would you like to give us feedback on how we're doing?

- Join us in our chat room at: Freenode IRC at #cloudmonitoring

  Or just click the following link:

  Webchat

- You can also provide feedback using our feedback form:

  Product Feedback Forum

- Get support here:

  File a ticket here

v1.0, December 15, 2011

These release notes correspond to the Private Beta for Rackspace Cloud Monitoring.

Product Features

- **Multiple Monitoring Zones Perspective**

  Monitor a single resource from many different monitoring zones.

- **Alarm History**

  Use an API to retrieve time series data from the system. Use this to go back in time and verify which notifications went out and why.

- **Remote IPv4/v6 Checks**

  Monitor IPv4/IPv6 services via IP addresses or using DNS resolution. Using DNS resolution, specify which type of address to resolve, A or AAAA.

- **Data Model**

  We've disconnected the concept of alerting and data collection so you can create robust and rich monitors. Check many aspects of a single resource; for instance, make sure a site is responding with a 201-status code, and features the copyright as a body match on the bottom of the page. Ensure all these conditions are met before the status of an alarm is "OK."

- **Alarm Language**

  Create a unique query with our language purposefully built to express thresholds and alert conditions. String compare metrics, use a regular expression operator, check the rate of change, and much more. With each of these primitives you can chain together multiple conditionals to set the state of an alarm. Our documentation highlights best practices for building robust monitors and alarms.

- **On-demand Simulation**

  Run a check on-demand, to verify its status before adding it. This allows you to get a feel for a monitor before being alerted by it. On top of that, you can take the results and pass it another endpoint to simulate your alarm language criteria match! This is extremely powerful as it takes some of the complexity out of guessing and allows you to simulate responses as they happen in normal operation.

- **Audit Log**

  Track all changes to an entity, check and alarm. Get the before and after of every changed object in the system. You can even add a "who" and "why" parameter to each POST/PUT request to keep a log of why the changes were made.

- **Change Alert Destination**

Depending upon which event is generated, you can execute different notifications; for instance on a warning alert, fire a webhook, but on an error alert, send sysadmin@yourcompany.com an urgent email.

Documentation

- Try some simple monitoring exercises in the *Rackspace Cloud Monitoring Getting Started Guide* at:

  http://docs.rackspace.com/cm/api/v1.0/cm-getting-started/content/Introduction.html.

- Get reference information and examples for all endpoints in the *Rackspace Cloud Monitoring Developers Guide* at:

  http://docs.rackspace.com/cm/api/v1.0/cm-devguide/content/index.html.

Talk to Us

Do you have questions about Rackspace Cloud Monitoring? Would you like to give us feedback on how we're doing?

- Join us in our chat room at: Freenode IRC at #cloudmonitoring

  Or just click the following link:

  Webchat

- You can also provide feedback using our feedback form:

  Product Feedback Forum

- Get email support here:

  Email Support List