TubeMogul

Optimizing your Monitoring and Trending tools for the Cloud

Nagios World Conference 2012

Nagios World Conference North America

Nicolas Brousse
Lead Operations Engineer
September 28th 2012

- About TubeMogul
- What are some of our challenges?
- Our environment
- Amazon Cloud Environment
- Automated Monitoring
- Efficient on-call rotation
- Efficient monitoring
- What's next?
- Q&A

About TubeMogul

- Founded in 2006
- Formerly a video distribution and analytics platform
- TubeMogul is a Brand-Focused Video Marketing Company
  - Build for Branding
  - Integrate real-time media buying, ad serving, targeting, optimization and brand measurement

TubeMogul simplifies the delivery of video ads and maximizes the impact of every dollar spent by brand marketers

http://www.tubemogul.com

What are some of our challenges?

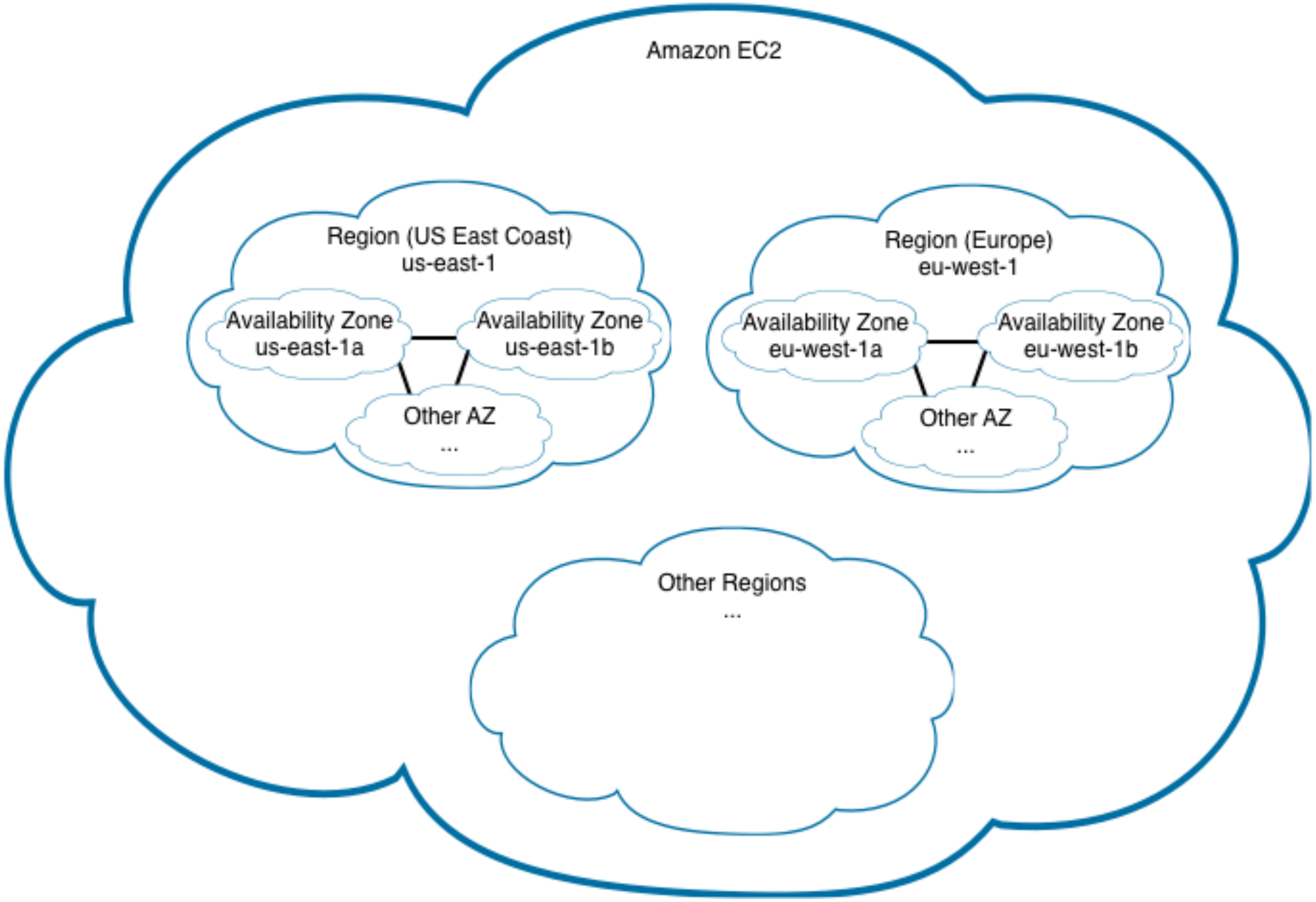
- Monitoring between 700 to 1000 servers
- Servers spread across 6 different locations
  - 4 Amazon EC2 Regions (our public cloud provider)
  - 1 Hosted (Liquidweb) & 1 VPS (Linode)
- Little monitoring resources
  - Collecting over 115,000 metrics
  - Monitoring over 20,000 services with Nagios
- Multiple billions of HTTP requests a day
  - Most of it must be served in less than 100ms
  - Lost of traffic could mean lost of business opportunity
  - Or worst, over-spending…

Our environment

Nagios® World Conference North America

- Over 80 different server profiles
- Our stack:
  - Java (Embedded Jetty, Tomcat)
  - PHP, RoR
  - Hadoop: HDFS, M/R, Hbase, Hive
  - Couchbase
  - MySQL
- Monitoring: Nagios, NSCA
- Graphing: Ganglia, sFlow, Graphite
- Configuration Management: Puppet

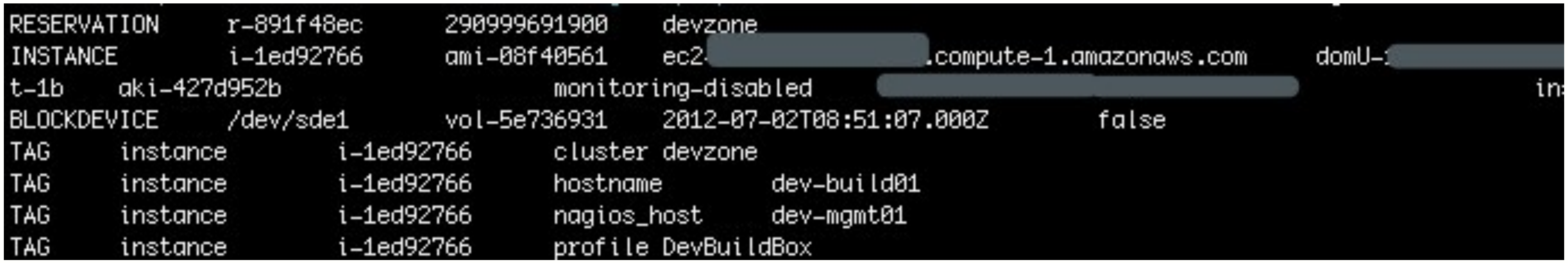
Amazon Cloud Environment


Amazon EC2
Region (US East Coast) us-east-1
Availability Zone us-east-1a
Availability Zone us-east-1b
Other AZ ...
Region (Europe) eu-west-1
Availability Zone eu-west-1a
Availability Zone eu-west-1b
Other AZ ...
Other Regions ...

Amazon Cloud Environment

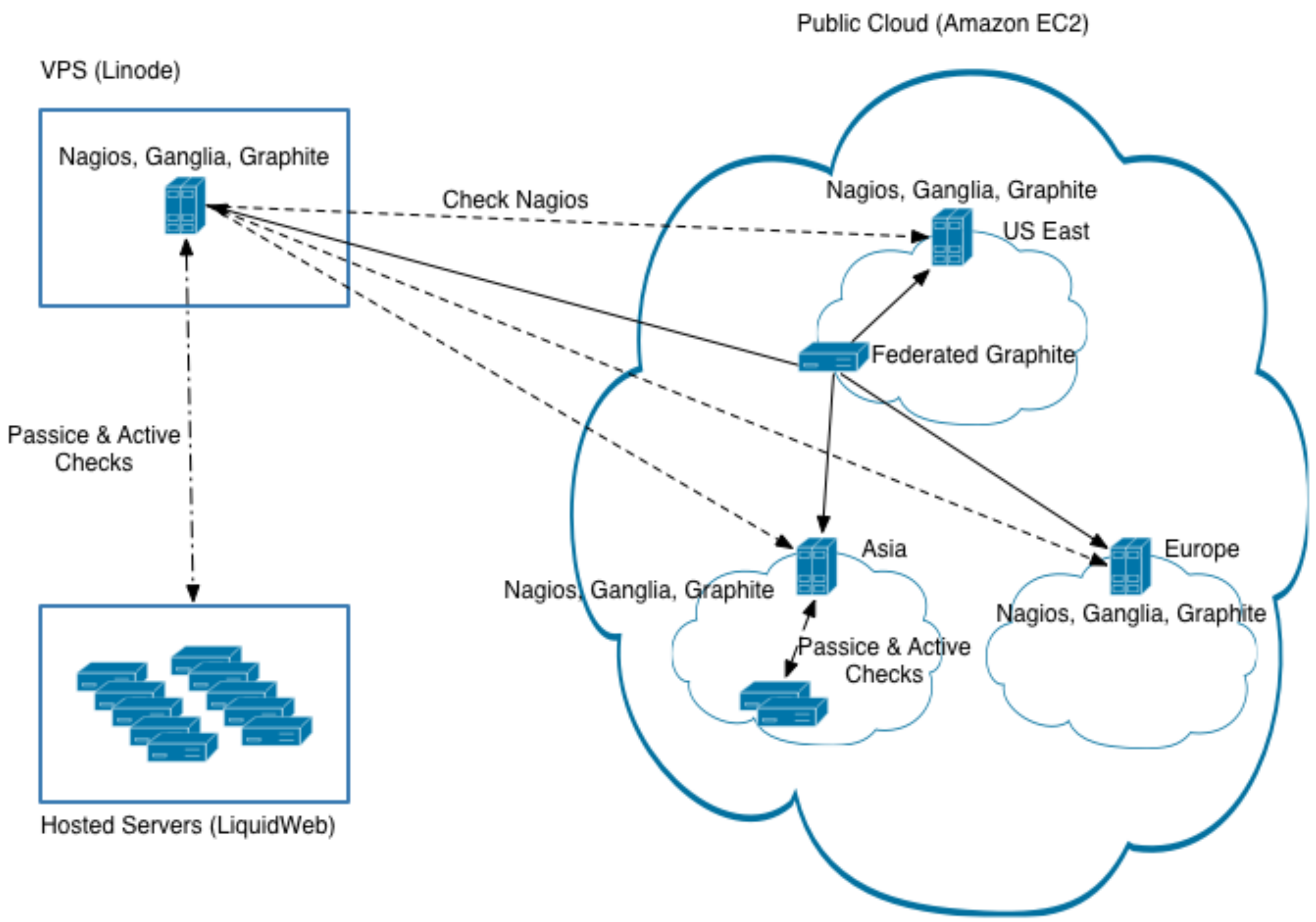
- We use EC2, SDB, SQS, EMR, S3, etc.
- We don't use ELB
- We heavily use EC2 Tags

ec2-describe-instances -F tag:hostname=dev-build01



```
RESERVATION     r-891f48ec      290999691900    devzone
INSTANCE        i-1ed92766      ami-08f40561    ec2-[redacted].compute-1.amazonaws.com      domU-[redacted]
t-1b    aki-427d952b                    monitoring-disabled     [redacted]                      in
BLOCKDEVICE     /dev/sde1       vol-5e736931    2012-07-02T08:51:07.000Z        false
TAG     instance        i-1ed92766      cluster devzone
TAG     instance        i-1ed92766      hostname        dev-build01
TAG     instance        i-1ed92766      nagios_host     dev-mgmt01
TAG     instance        i-1ed92766      profile DevBuildBox
```

Automated Monitoring



Public Cloud (Amazon EC2)
VPS (Linode)
Nagios, Ganglia, Graphite
Check Nagios
Nagios, Ganglia, Graphite
US East
Federated Graphite
Passice & Active Checks
Asia
Europe
Nagios, Ganglia, Graphite
Nagios, Ganglia, Graphite
Passice & Active Checks
Hosted Servers (LiquidWeb)

Configuring Ganglia using Puppet templates

```
globals {
 …
 override_hostname = <%= scope.lookupvar('hostname') %>
…
}
udp_send_channel {
  host = <%= scope.lookupvar('ec2_tag_nagios_host') %>
  port = 8649
  ttl = 1
}
sflow {
  udp_port = 6343
  accept_jvm_metrics = yes
  multiple_jvm_instances = yes
}
```

Automated Monitoring

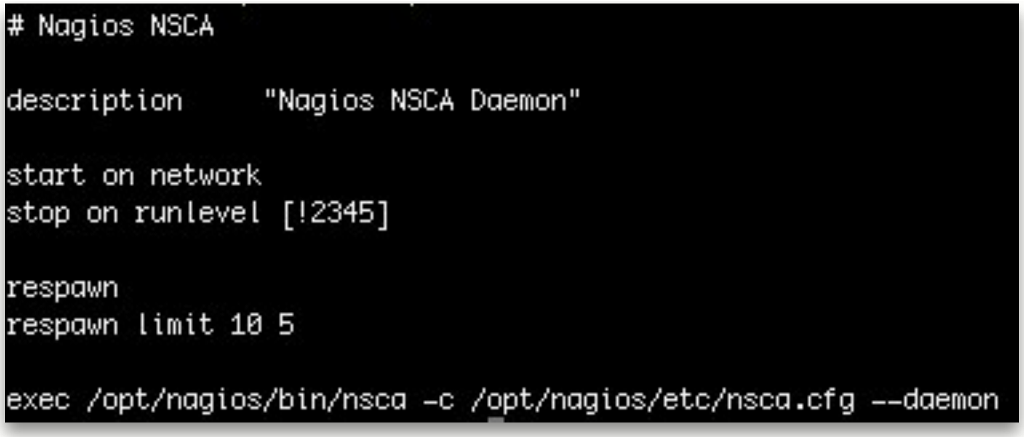
Nagios® World Conference North America

Or configuring Host sFlow using Puppet templates

```
sflow{
  DNSSD = off
  polling = 20
  sampling = 512
  collector{
   ip = <%= ec2_tag_nagios_host %>
   udpport = 6343
  }
}
```

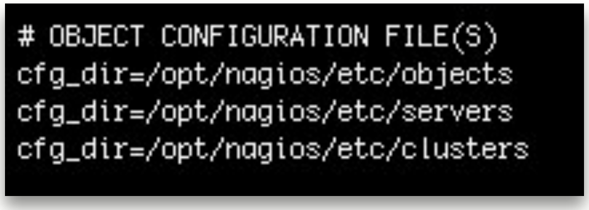
Automated Monitoring

- Puppet configure our monitoring instances
  - We use Nagios regex : **use_regexp_matching=1**
  - But we don't use true regex : **use_true_regexp_matching=0**
  - We use NSCA with Upstart



```
# Nagios NSCA

description     "Nagios NSCA Daemon"

start on network
stop on runlevel [!2345]

respawn
respawn limit 10 5

exec /opt/nagios/bin/nsca -c /opt/nagios/etc/nsca.cfg --daemon
```

  - We don't use the perfdata
  - We use pre-cached objects
  - We includes our configurations from 3 directories
    - objects => templates, contacts, commands, event_handlers
    - servers => contain a configuration file for each server
    - clusters => contain a configuration file for each cluster



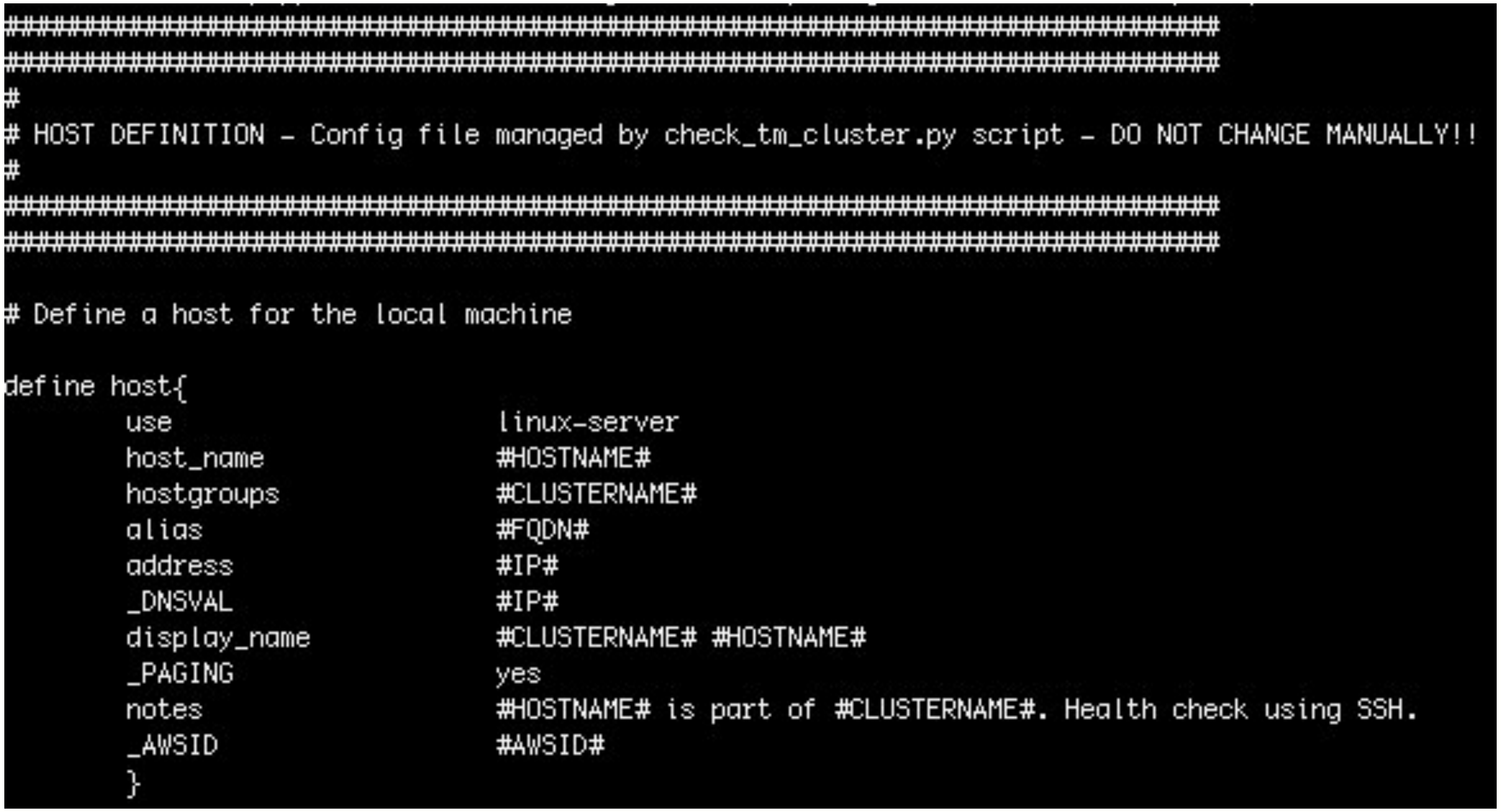
```
# OBJECT CONFIGURATION FILE(S)
cfg_dir=/opt/nagios/etc/objects
cfg_dir=/opt/nagios/etc/servers
cfg_dir=/opt/nagios/etc/clusters
```

Automated Monitoring

Process of event when starting a new host and add it to our monitoring:

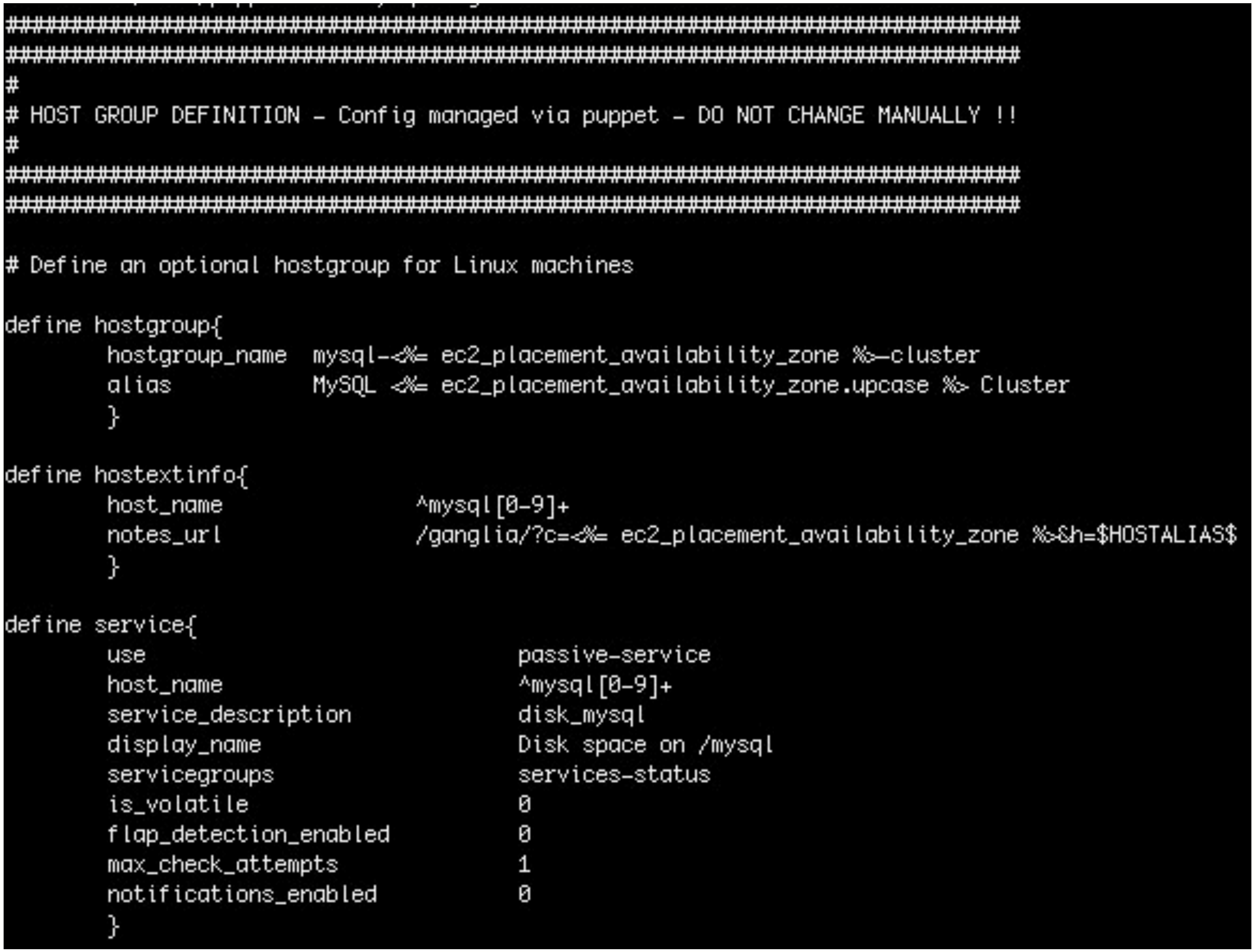
1. We start a new instance using Cerveza and Cloud-init
2. Puppet configure Gmond or Host sFlow on the instance
3. Our monitoring server running Gmond and Gmetad get data from the new instance
4. A Nagios check run every minute and check for new hosts
   - Look for new hosts using EC2 API
   - Look for EC2 tag “hostname” to confirm it’s a legit host, not a zombie / fail start
   - Look for EC2 tag “nagios_host” to see if the host belong to this monitoring instance
5. If a new host is found:
   - We build a config for the host based on a template file and doing some string replace
   - Once all config have been generated, we rebuild pre-cache objects and reload Nagios
6. If we find “Zombie” host, we generate a Warning alert
7. If the config is corrupt, we send a Critical alert

Automated Monitoring



```
################################################################################
################################################################################
#
# HOST DEFINITION - Config file managed by check_tm_cluster.py script - DO NOT CHANGE MANUALLY!!
#
################################################################################
################################################################################

# Define a host for the local machine

define host{
        use                     linux-server
        host_name               #HOSTNAME#
        hostgroups              #CLUSTERNAME#
        alias                   #FQDN#
        address                 #IP#
        _DNSVAL                 #IP#
        display_name            #CLUSTERNAME# #HOSTNAME#
        _PAGING                 yes
        notes                   #HOSTNAME# is part of #CLUSTERNAME#. Health check using SSH.
        _AWSID                  #AWSID#
        }
```

Automated Monitoring



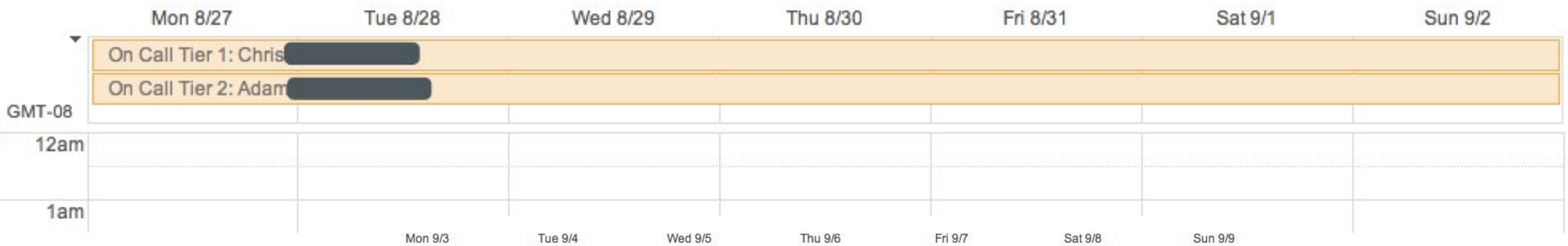
```
################################################################################
################################################################################
#
# HOST GROUP DEFINITION - Config managed via puppet - DO NOT CHANGE MANUALLY !!
#
################################################################################
################################################################################

# Define an optional hostgroup for Linux machines

define hostgroup{
        hostgroup_name  mysql-<%= ec2_placement_availability_zone %>-cluster
        alias           MySQL <%= ec2_placement_availability_zone.upcase %> Cluster
        }

define hostextinfo{
        host_name               ^mysql[0-9]+
        notes_url               /ganglia/?c=<%= ec2_placement_availability_zone %>&h=$HOSTALIAS$
        }

define service{
        use                             passive-service
        host_name                       ^mysql[0-9]+
        service_description             disk_mysql
        display_name                    Disk space on /mysql
        servicegroups                   services-status
        is_volatile                     0
        flap_detection_enabled          0
        max_check_attempts              1
        notifications_enabled           0
        }
```

Efficient on-call rotation

- Follow the sun
  - Some of our team is in Ukraine, no more Tier 1 night on-call for us
- Nagios timeperiod and escalation are a pain to maintain
  - Nagios notification plugged to Google Calendar
    - Using our own notification script for email and paging
    - Google Clendar make it easy for each team to manage their own on-call calendar
    - Support for multiple Tier and complex schedules
    - Caching Google Calendar info locally every hour
  - Simpler definitions and rules in Nagios contacts
  - Notify only people on-call, unless they asked for “off call” emails

Efficient on-call rotation

Using Google Calendar…



Mon 8/27 | Tue 8/28 | Wed 8/29 | Thu 8/30 | Fri 8/31 | Sat 9/1 | Sun 9/2

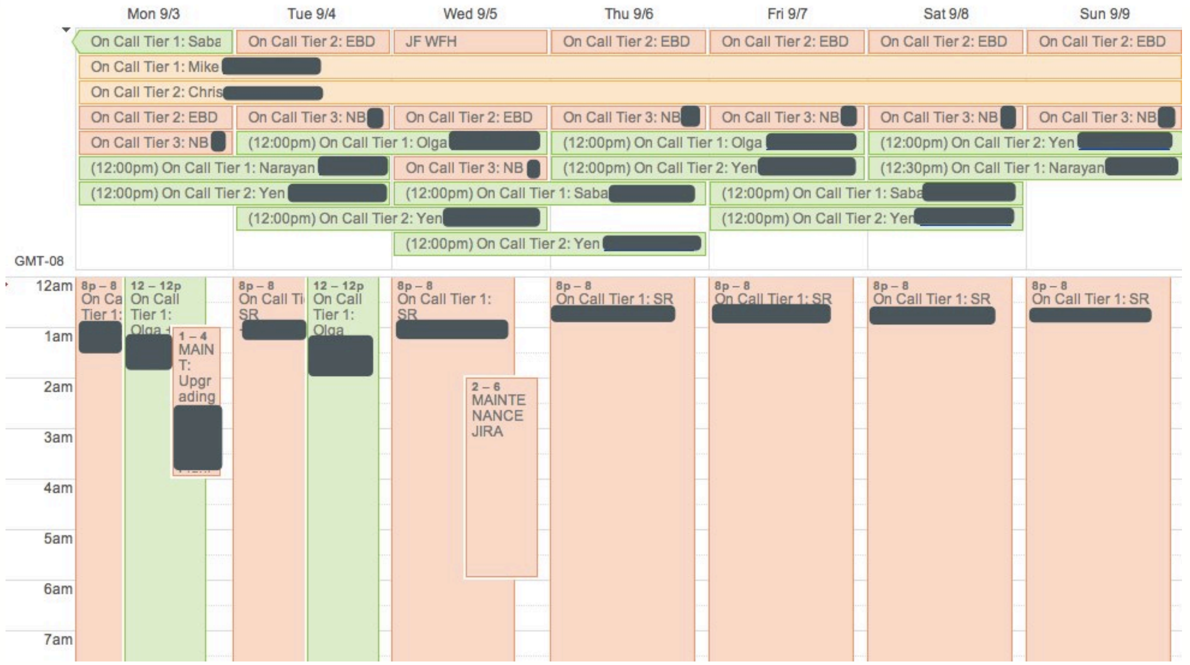
On Call Tier 1: Chris [redacted]

On Call Tier 2: Adam [redacted]

GMT-08

12am

1am



Mon 9/3 | Tue 9/4 | Wed 9/5 | Thu 9/6 | Fri 9/7 | Sat 9/8 | Sun 9/9

On Call Tier 1: Saba

On Call Tier 2: EBD

JF WFH

On Call Tier 2: EBD

On Call Tier 2: EBD

On Call Tier 2: EBD

On Call Tier 2: EBD

On Call Tier 1: Mike [redacted]

On Call Tier 2: Chris [redacted]

On Call Tier 2: EBD

On Call Tier 3: NB [redacted]

On Call Tier 2: EBD

On Call Tier 3: NB [redacted]

On Call Tier 3: NB [redacted]

On Call Tier 3: NB [redacted]

On Call Tier 3: NB [redacted]

On Call Tier 3: NB [redacted]

(12:00pm) On Call Tier 1: Olga [redacted]

(12:00pm) On Call Tier 1: Olga [redacted]

(12:00pm) On Call Tier 2: Yen [redacted]

(12:00pm) On Call Tier 1: Narayan [redacted]

On Call Tier 3: NB [redacted]

(12:00pm) On Call Tier 2: Yen [redacted]

(12:30pm) On Call Tier 1: Narayan [redacted]

(12:00pm) On Call Tier 2: Yen [redacted]

(12:00pm) On Call Tier 1: Saba [redacted]

(12:00pm) On Call Tier 1: Saba [redacted]

(12:00pm) On Call Tier 2: Yen [redacted]

(12:00pm) On Call Tier 2: Yen [redacted]

(12:00pm) On Call Tier 2: Yen [redacted]

GMT-08

12am 1am 2am 3am 4am 5am 6am 7am

8p – 8 On Ca Tier 1: [redacted]

12 – 12p On Call Tier 1: Olga [redacted]

1 – 4 MAINT: Upgrading [redacted]

8p – 8 On Call Ti SR [redacted]

12 – 12p On Call Tier 1: Olga [redacted]

8p – 8 On Call Tier 1: SR [redacted]

2 – 6 MAINTENANCE JIRA

8p – 8 On Call Tier 1: SR [redacted]

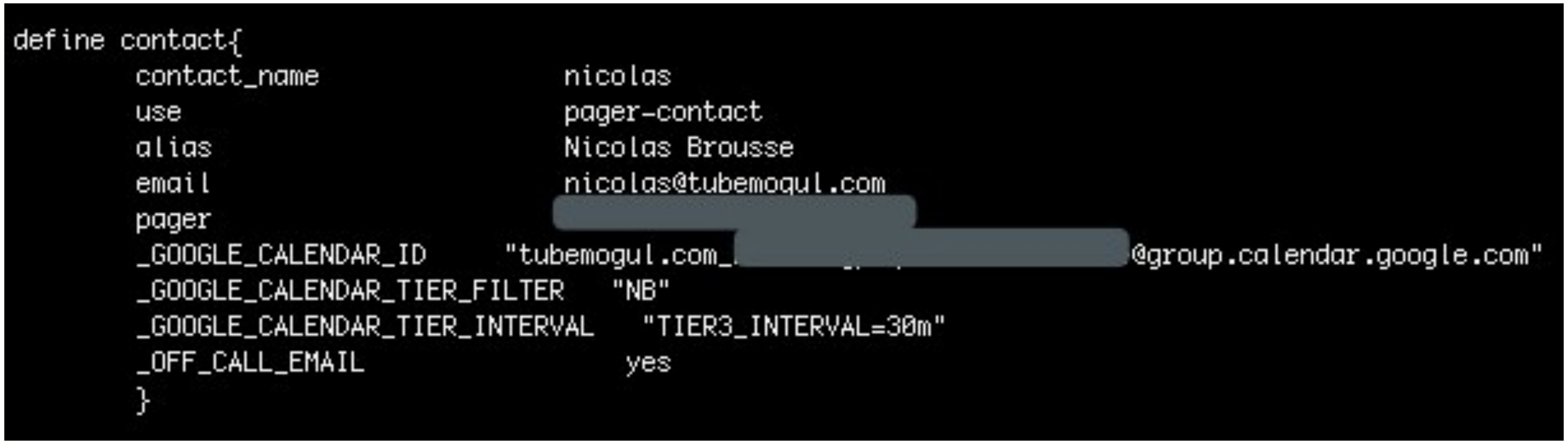
8p – 8 On Call Tier 1: SR [redacted]

8p – 8 On Call Tier 1: SR [redacted]

8p – 8 On Call Tier 1: SR [redacted]

Efficient on-call rotation

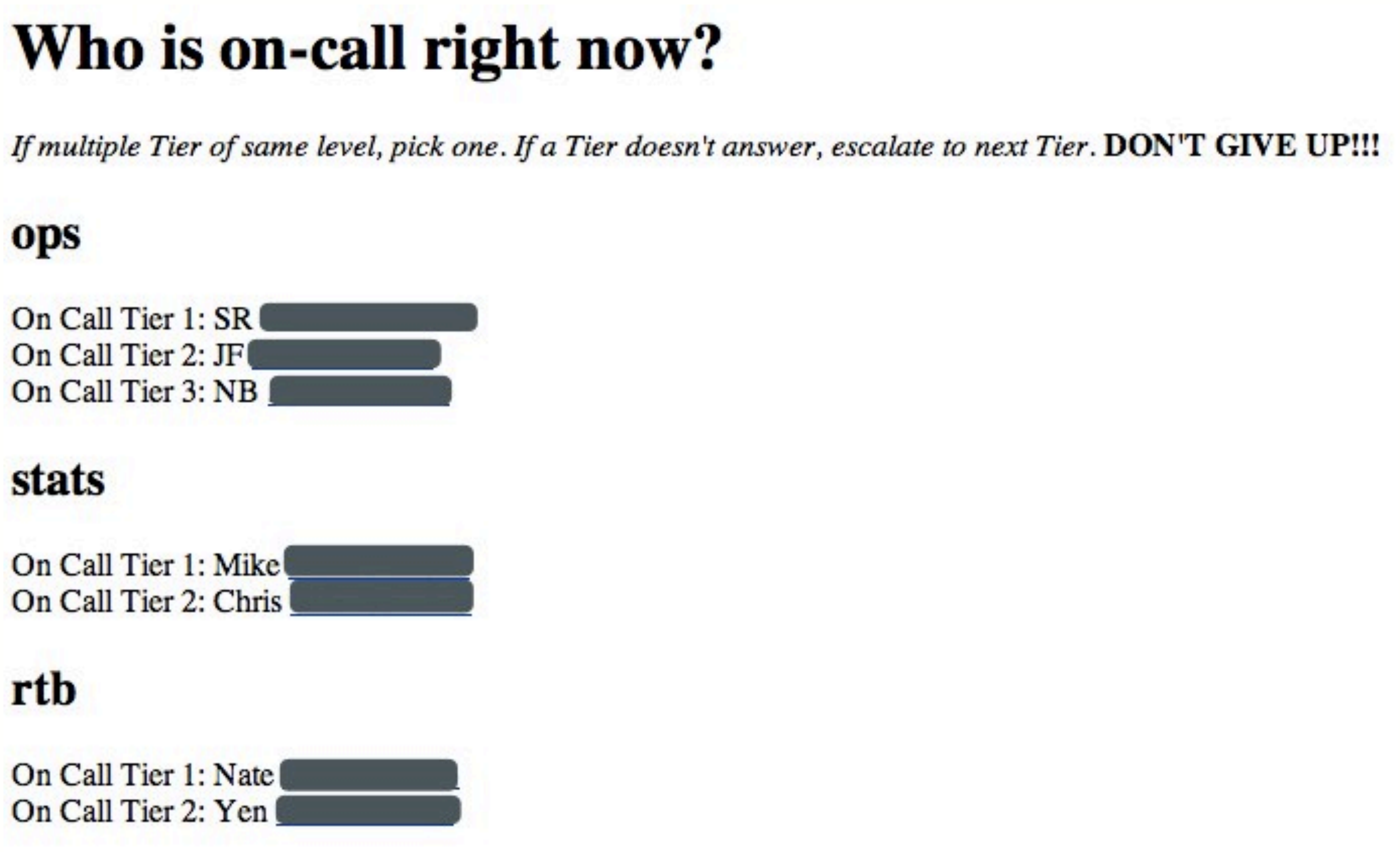
- Simple contact definitions
- Google Calendar info
- Tier Filter (Regex)
- Tier Interval (time to wait before escalating alert since last tier)
- Off call email



```
define contact{
        contact_name                    nicolas
        use                             pager-contact
        alias                           Nicolas Brousse
        email                           nicolas@tubemogul.com
        pager                           [redacted]
        _GOOGLE_CALENDAR_ID      "tubemogul.com_[redacted]@group.calendar.google.com"
        _GOOGLE_CALENDAR_TIER_FILTER    "NB"
        _GOOGLE_CALENDAR_TIER_INTERVAL    "TIER3_INTERVAL=30m"
        _OFF_CALL_EMAIL                     yes
        }
```

Efficient on-call rotation

Nagios World Conference North America



Who is on-call right now?

If multiple Tier of same level, pick one. If a Tier doesn't answer, escalate to next Tier. **DON'T GIVE UP!!!**

ops

On Call Tier 1: SR [redacted]
On Call Tier 2: JF [redacted]
On Call Tier 3: NB [redacted]

stats

On Call Tier 1: Mike [redacted]
On Call Tier 2: Chris [redacted]

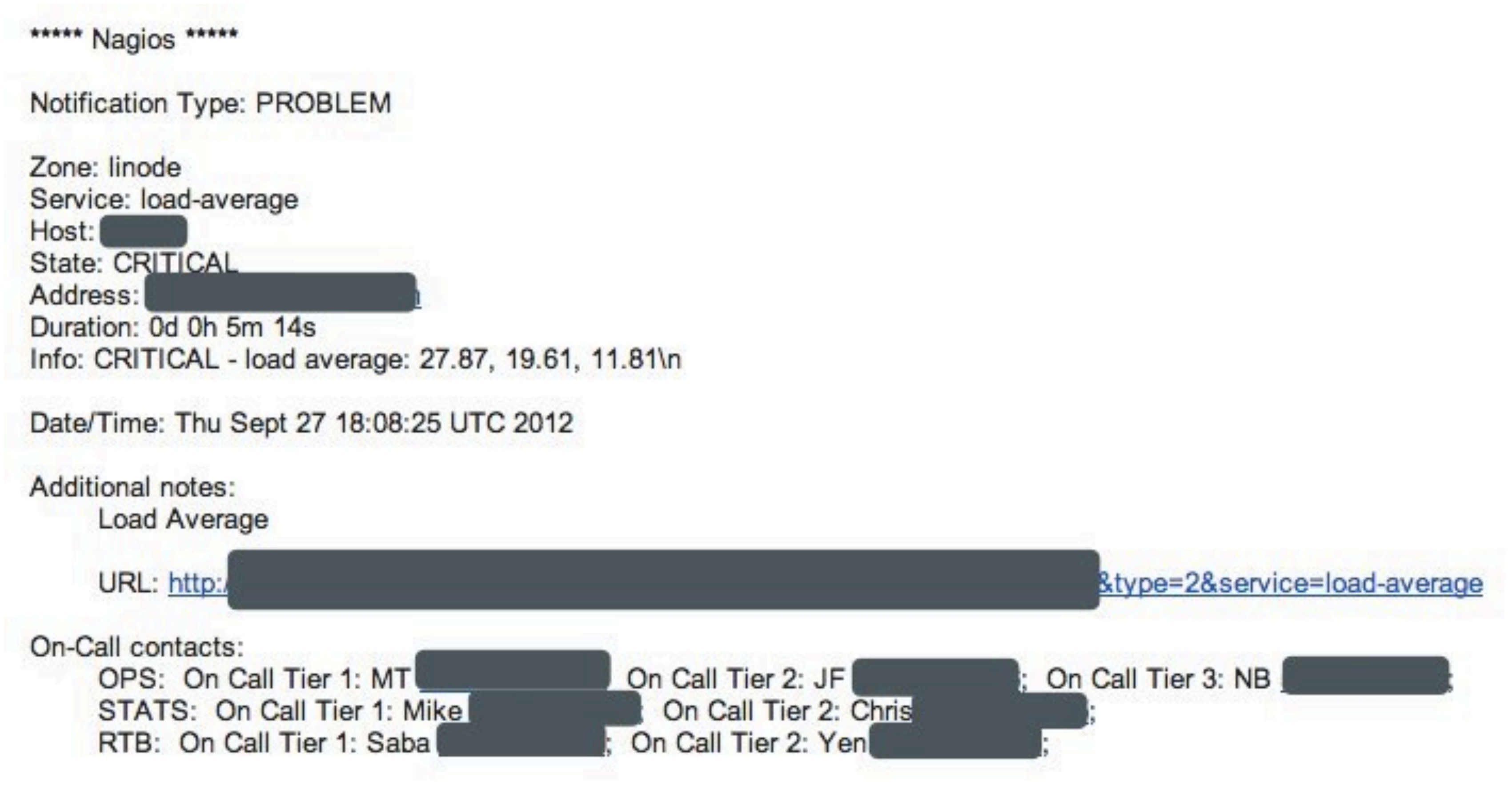
rtb

On Call Tier 1: Nate [redacted]
On Call Tier 2: Yen [redacted]

Efficient on-call rotation

Nagios World Conference North America

On-call contact fetched from Google Calendar at the bottom of the alert makes our life easier!

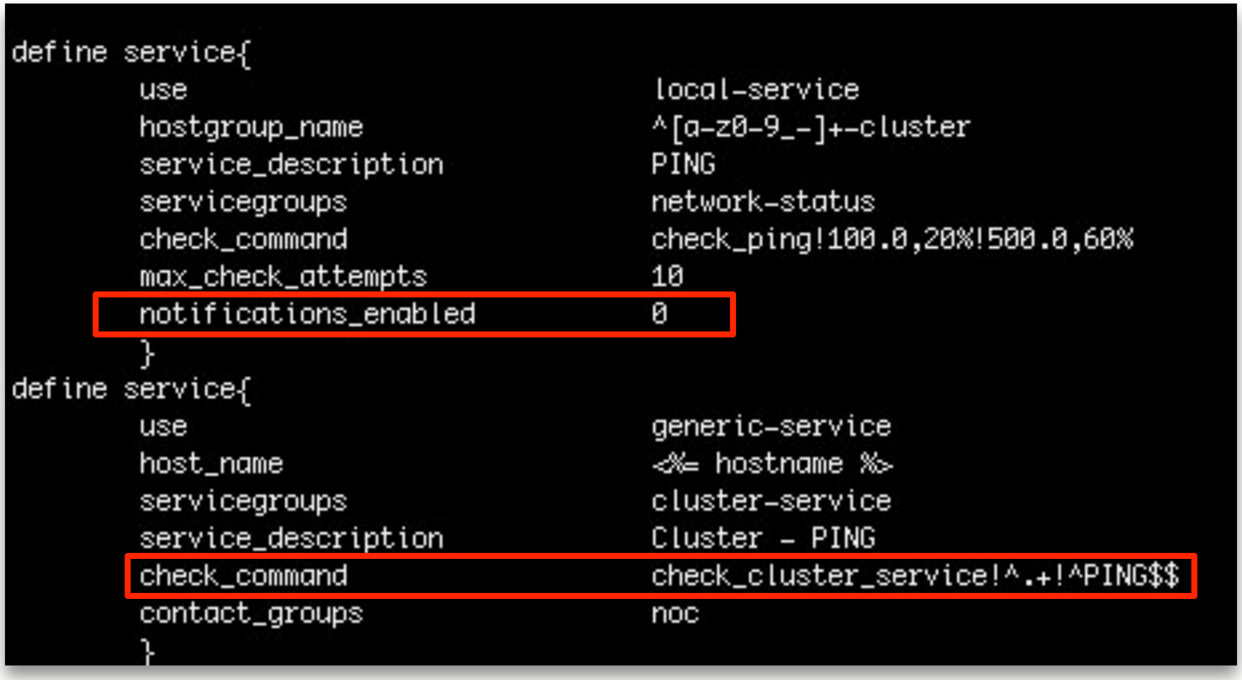


```
***** Nagios *****

Notification Type: PROBLEM

Zone: linode
Service: load-average
Host: [redacted]
State: CRITICAL
Address: [redacted]
Duration: 0d 0h 5m 14s
Info: CRITICAL - load average: 27.87, 19.61, 11.81\n

Date/Time: Thu Sept 27 18:08:25 UTC 2012

Additional notes:
      Load Average

      URL: http:/[redacted]&type=2&service=load-average

On-Call contacts:
      OPS:  On Call Tier 1: MT [redacted]  On Call Tier 2: JF [redacted];  On Call Tier 3: NB [redacted];
      STATS:  On Call Tier 1: Mike [redacted]  On Call Tier 2: Chris [redacted];
      RTB:  On Call Tier 1: Saba [redacted];  On Call Tier 2: Yen [redacted];
```

Efficient monitoring

Nagios World Conference North America

We disable most notification and only care of a cluster status

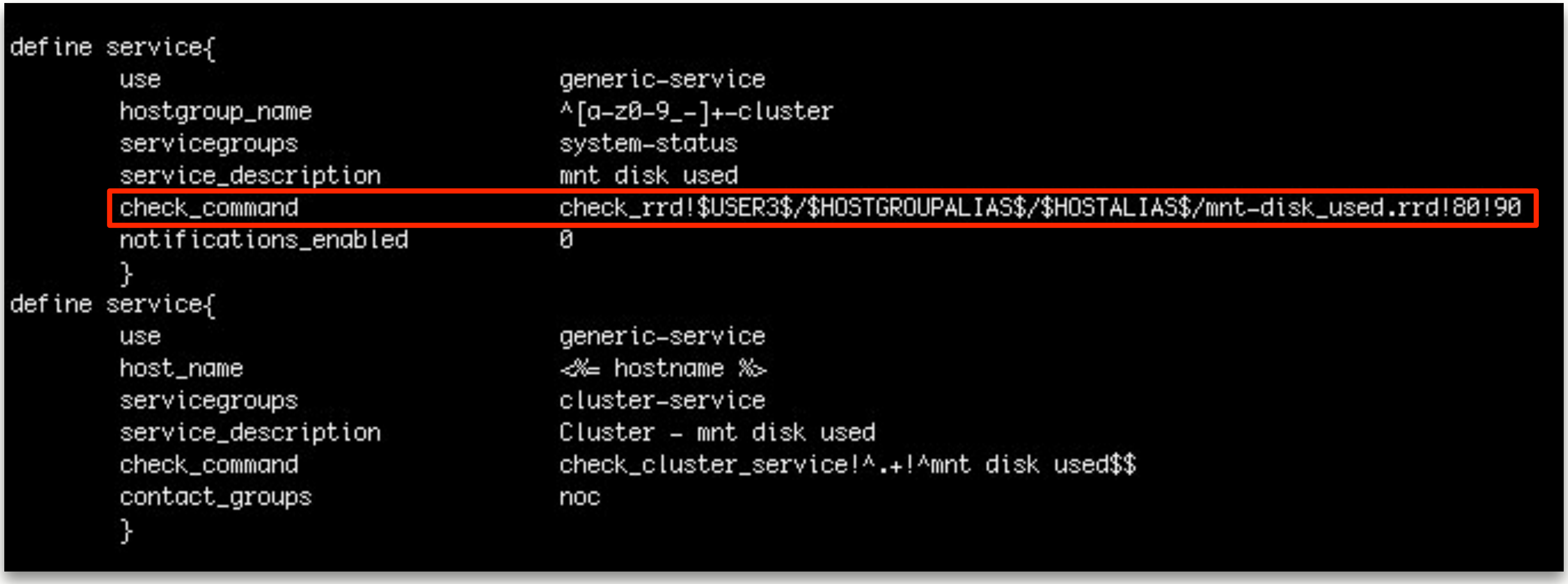


```
define service{
        use                             local-service
        hostgroup_name                  ^[a-z0-9_-]+-cluster
        service_description             PING
        servicegroups                   network-status
        check_command                   check_ping!100.0,20%!500.0,60%
        max_check_attempts              10
        notifications_enabled           0
        }
define service{
        use                             generic-service
        host_name                       <%= hostname %>
        servicegroups                   cluster-service
        service_description             Cluster - PING
        check_command                   check_cluster_service!^.+!^PING$$
        contact_groups                  noc
        }
```

Efficient monitoring

Nagios® World Conference North America

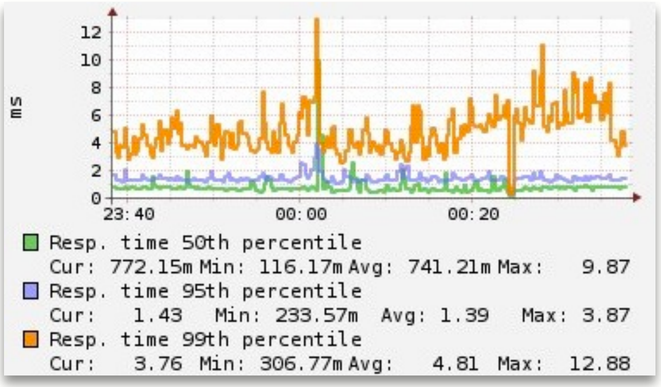
Most of our checks are based on Ganglia RRD files



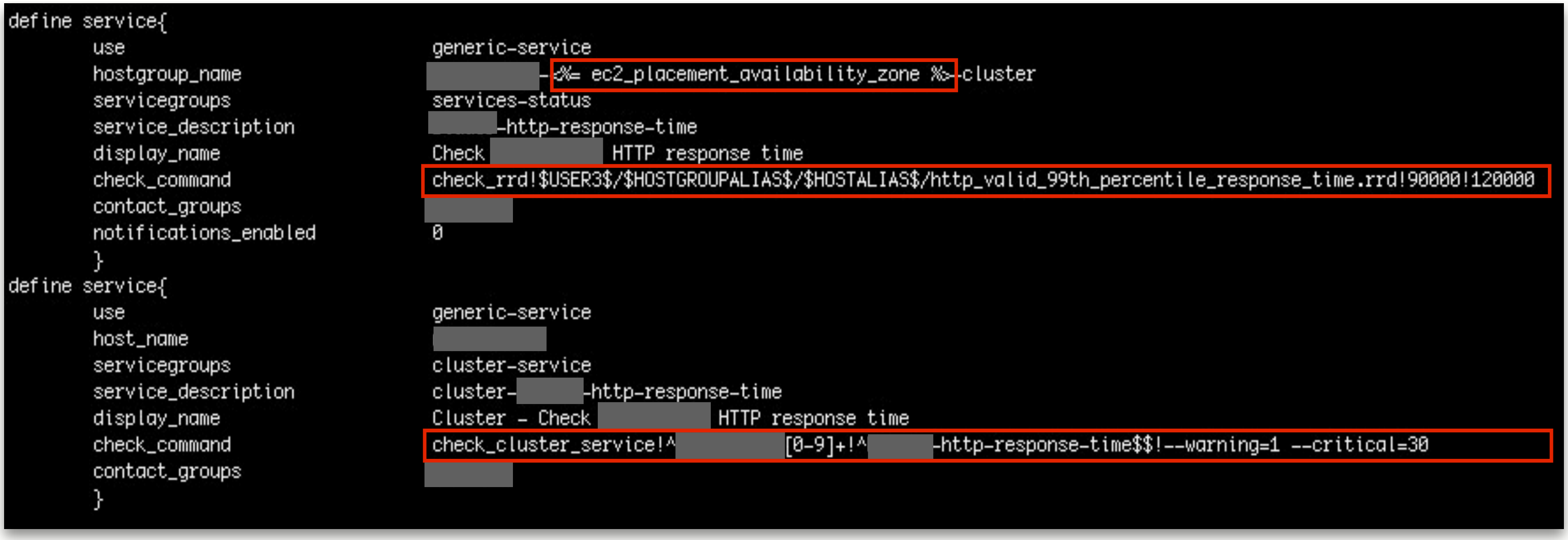
```
define service{
        use                             generic-service
        hostgroup_name                  ^[a-z0-9_-]+-cluster
        servicegroups                   system-status
        service_description             mnt disk used
        check_command                   check_rrd!$USER3$/$HOSTGROUPALIAS$/$HOSTALIAS$/mnt-disk_used.rrd!80!90
        notifications_enabled           0
        }
define service{
        use                             generic-service
        host_name                       <%= hostname %>
        servicegroups                   cluster-service
        service_description             Cluster - mnt disk used
        check_command                   check_cluster_service!^.+!^mnt disk used$$
        contact_groups                  noc
        }
```

Efficient monitoring

It become really easy to monitor any metrics returned by Ganglia



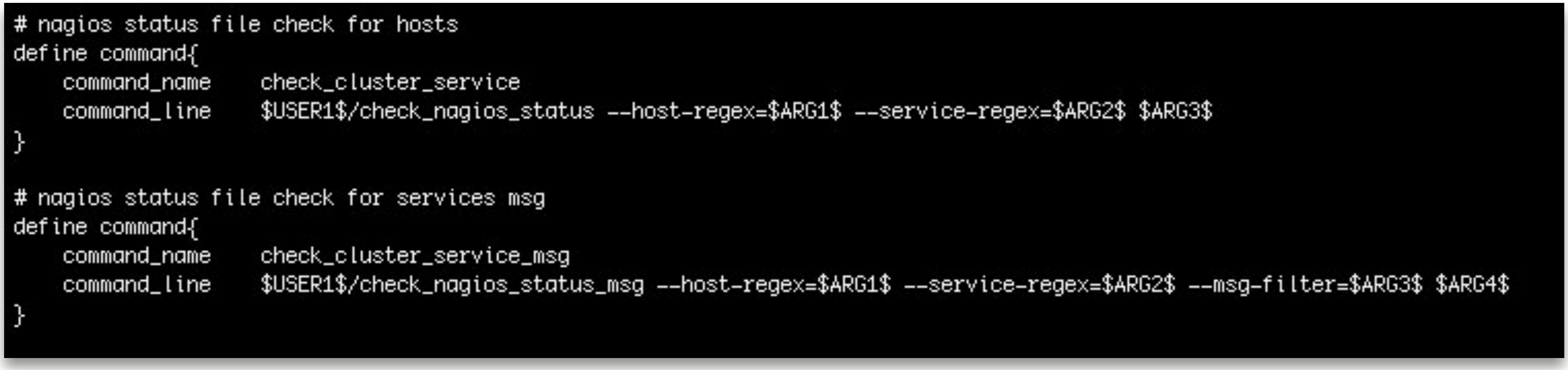
```
🟩 Resp. time 50th percentile
   Cur: 772.15m Min: 116.17m Avg: 741.21m Max:    9.87
🟦 Resp. time 95th percentile
   Cur:    1.43   Min: 233.57m   Avg: 1.39    Max: 3.87
🟧 Resp. time 99th percentile
   Cur:    3.76  Min: 306.77m Avg:    4.81  Max:   12.88
```



```
define service{
        use                     generic-service
        hostgroup_name          [redacted]-<%= ec2_placement_availability_zone %>-cluster
        servicegroups           services-status
        service_description     [redacted]-http-response-time
        display_name            Check [redacted] HTTP response time
        check_command           check_rrd!$USER3$/$HOSTGROUPALIAS$/$HOSTALIAS$/http_valid_99th_percentile_response_time.rrd!90000!120000
        contact_groups          [redacted]
        notifications_enabled   0
        }
define service{
        use                     generic-service
        host_name               [redacted]
        servicegroups           cluster-service
        service_description     cluster-[redacted]-http-response-time
        display_name            Cluster - Check [redacted] HTTP response time
        check_command           check_cluster_service!^[redacted][0-9]+!^[redacted]-http-response-time$$!--warning=1 --critical=30
        contact_groups          [redacted]
        }
```

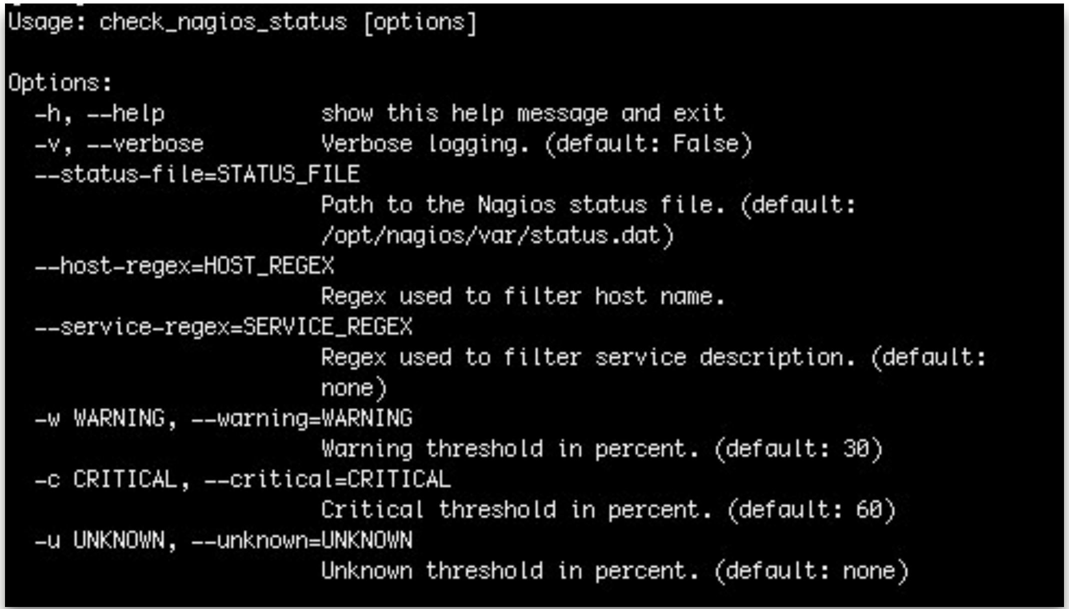
Efficient monitoring

We can check cluster status by hosts/services but also per returned messages !

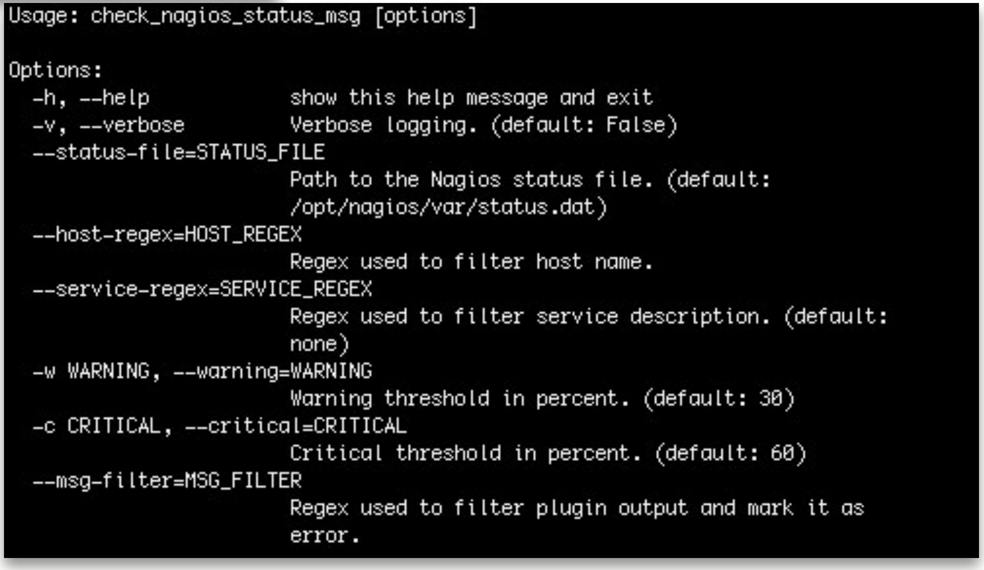


```
# nagios status file check for hosts
define command{
    command_name    check_cluster_service
    command_line    $USER1$/check_nagios_status --host-regex=$ARG1$ --service-regex=$ARG2$ $ARG3$
}

# nagios status file check for services msg
define command{
    command_name    check_cluster_service_msg
    command_line    $USER1$/check_nagios_status_msg --host-regex=$ARG1$ --service-regex=$ARG2$ --msg-filter=$ARG3$ $ARG4$
}
```

Efficient monitoring



```
Usage: check_nagios_status [options]

Options:
  -h, --help            show this help message and exit
  -v, --verbose         Verbose logging. (default: False)
  --status-file=STATUS_FILE
                        Path to the Nagios status file. (default:
                        /opt/nagios/var/status.dat)
  --host-regex=HOST_REGEX
                        Regex used to filter host name.
  --service-regex=SERVICE_REGEX
                        Regex used to filter service description. (default:
                        none)
  -w WARNING, --warning=WARNING
                        Warning threshold in percent. (default: 30)
  -c CRITICAL, --critical=CRITICAL
                        Critical threshold in percent. (default: 60)
  -u UNKNOWN, --unknown=UNKNOWN
                        Unknown threshold in percent. (default: none)
```

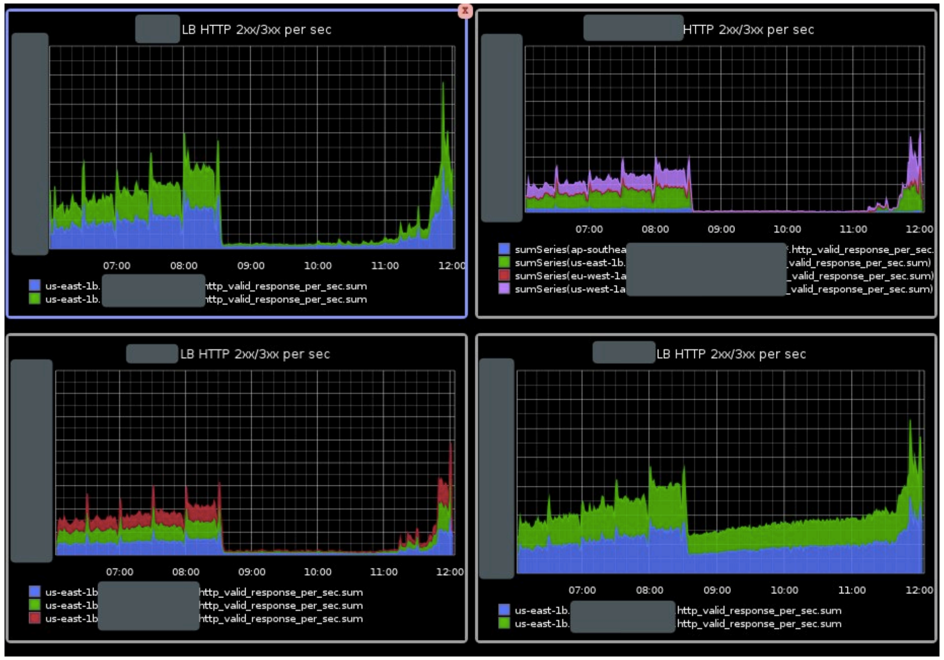


```
Usage: check_nagios_status_msg [options]

Options:
  -h, --help            show this help message and exit
  -v, --verbose         Verbose logging. (default: False)
  --status-file=STATUS_FILE
                        Path to the Nagios status file. (default:
                        /opt/nagios/var/status.dat)
  --host-regex=HOST_REGEX
                        Regex used to filter host name.
  --service-regex=SERVICE_REGEX
                        Regex used to filter service description. (default:
                        none)
  -w WARNING, --warning=WARNING
                        Warning threshold in percent. (default: 30)
  -c CRITICAL, --critical=CRITICAL
                        Critical threshold in percent. (default: 60)
  --msg-filter=MSG_FILTER
                        Regex used to filter plugin output and mark it as
                        error.
```

Efficient monitoring

Using Graphite Federated Storage

- One place to see all our metrics from all the world
- No delay due to rsync of RRD files
- Graph close to real time, delay only due to rrdcached flushing interval



What’s next?

Nagios® World Conference North America

Some hot topics…

- Do trending alert with Nagios based on Graphite/Ganglia data
- Better automation for non-cloud servers
- Ensure we can scale our monitoring when using hybrid cloud (Eucalyptus) or multiple public cloud provider
- Get a better centralized view of our different Nagios

Ops @ TubeMogul

All this wouldn't be possible without a strong System Operation team

Andrey Shestakov

Eamon Bisson Donahue

Justin Francisconi

Marylene Tanfin

Nicolas Brousse

Stan Rudenko

Thank you…

Nagios® World Conference North America

TubeMogul is Hiring !

http://www.tubemogul.com/jobs

jobs@tubemogul.com

Follow us on Twitter

@TubeMogul

@orieg