Some practical considerations for monitoring in OpenStack cloud

Piotr Siwczak on September 4, 2012

There is an old saying in IT: "Tough guys never do backups." While this kind of thinking doesn't prevent infrastructures from functioning, it's definitely asking for trouble. And the same goes for monitoring.

Having gone through all the pain of polishing an OpenStack installation, I can see why it might seem like a better idea to grab a couple of beers in a local bar than spend additional hours fighting Nagios or Cacti. In the world of bare metal these tools seem to do the job—they warn about abnormal activities and provide insight into trend graphs.

With the advent of the cloud, however, proper monitoring has started getting out of hand for many IT teams. On top of the relatively predictable hardware, a complicated pile of software called "cloud middleware" has been dropped in our laps. Huge numbers of virtual resources (VMs, networks, storage) are coming and going at a rapid pace, causing unpredictable workload peaks.

Proper methods of dealing with this mess are in their infancy – and could even be worth launching a startup company around. This post is an attempt to shed some light on the various aspects of cloud monitoring, from the hardware to the user's cloud ecosystem.

Problem statement

In cloud environments, we can identify three distinct areas for monitoring:

- **Cloud hardware and services:** These are different hardware and software pieces of the cluster running on bare metal, including hypervisors and storage and controller nodes. The problem is well-known and a number of tools exist to deal with it. Some of most popular are Nagios, Ganglia, Cacti, and Zabbix.
- **User's cloud ecosystem:** This is everything that makes up a user's cloud account. In case of OpenStack it is instances, persistent volumes, floating IPs, security groups, etc. For all these components, the user needs reliable and clear information on their status. This info generally should come from the internals of the cloud software.
- **Performance of cloud resources:** This is the performance of tenants' cloud infrastructures running on top of a given OpenStack installation. This specifically boils down to determining what prevents tenants' resources from functioning properly and how these problematic resources affect other cloud resources.

Next, I'll elaborate on these three aspects of monitoring.

Monitoring OpenStack services and hardware

A vast array of monitoring and notification software is on the market. To monitor OpenStack and platform services (nova-compute, nova-network, MySQL, RabbitMQ, etc.), Mirantis usually deploys Nagios. This table shows which Nagios checks do the trick:

OpenStack/platform service	Nagios check
Database	`check_mysql/check_pgsql`
RabbitMQ	We use our custom plugin, but the RabbitMQ website also recommends this set of plugins
`libvirt`	`check_libvirt`
`dnsmasq`	`check_dhcp`
`nova-api`	`check_http`
`nova-scheduler`	`check_procs`
`nova-compute`	`check_procs`
`nova-network`	`check_procs`
`keystone-api`	`check_http`
`glance-api`	`check_http`
`glance-registry`	`check_http`
`server availability`	`check_ping`

You might also want to take a look at Zenoss ZenPack for Openstack.

Monitoring status of the user's cloud ecosystem

When it comes to users, their primary expectation is usually about consistent feedback from OpenStack about their instance states. There is nothing more annoying than having an instance reported as "ACTIVE" by the dashboard, even though it's been gone for several minutes.

OpenStack tries to prevent such discrepancies by running several checks in a cron-like manner (they are called "periodic tasks" in OpenStack code) on each compute node. These checks are simply methods of the ComputeManager class located in compute/manager.py or directly in drivers for different hypervisors (some of these checks are available for certain hypervisors only).

Check behavior can usually be controlled by flags. Some flags are limited to sending a notification or setting the instance to an ERROR state, while others try to take some proactive actions (rebooting, etc.). Currently only a very limited set of checks is provided.

Method	Flag	Hypervisor	Description
`_check_instance_build_time`	`instance_build_timeout`	Any	If the instance has been in BUILD state for more time than the flag indicates, then it is set to ERROR state.
`_cleanup_running_deleted_instances`	`running_deleted_instance_action`	Any	Cleans up any instances that are erroneously still running after having been deleted. Cleaning behavior is determined by the flag. Can be: `noop` (do nothing), `log`, `reap` (shut down).

`_poll_rebooting_instances`	`reboot_timeout`	XEN	Forces a reboot of the instance once it hits the timeout in REBOOT state
`_sync_power_states`	N/A	Any	If the instance is not found on the hypervisor, but is in the database, then it will be set to `power_state.NOSTATE` (which is currently translated to PENDING by default).

Even though a number of checks are present for the compute node, there seems to be no periodic check for the operation of other relevant components like `nova-network` or `nova-volume`. So, for the time being, essentially **none of these features is monitored by OpenStack**:

- **Floating IP attachments:** If we flush the NAT table on the compute node, then we just lose connectivity to the instance to which it was pointing and OpenStack still reports the floating IP as "associated." (This post explains how floating IPs work.)
- **Presence of security groups:** Under the hood, these map to iptables rules on the compute node. If we flush these rules or modify them by hand, we might accidentally open access holes to instances and OpenStack will not know about them.
- **Bridge connectivity:** If we bring a bridge down accidentally, all the VMs attached to it will stop responding, but they will still be reported as ACTIVE by OpenStack.
- **Presence of iSCSI target:** If we tamper manually with the `tgtadm` command on the `nova-volume` node, or just happen to lose the underlying logical volume that holds the user's data, OpenStack will probably not notice and the volume will be reported as OK.

The above issues definitely require substantial improvement as they are critical not only from a usability standpoint, but also because they introduce a security risk. A good OpenStack design draft to watch seems to be the one on resource monitoring. It deals not only with monitoring resource statuses, but also provides a notification framework to which different tenants can subscribe. As per the blueprint page, you can see that the project has not been accepted yet by the community.

Monitoring performance of cloud resources

While monitoring farms of physical servers is a standard task even on a large scale, monitoring virtual infrastructure ("cloud scale") is much more daunting. Cloud introduces a lot of dynamic resources, which can behave unpredictably and move between different hardware components. So it is usually quite hard to tell which of the thousands of VMs has problems (without even having root access to it) and how the problem affects other resources. Standards like sFlow try to tackle this problem, by providing efficient monitoring for a high volume of events (based on probing) and determining relationships between different cloud resources.

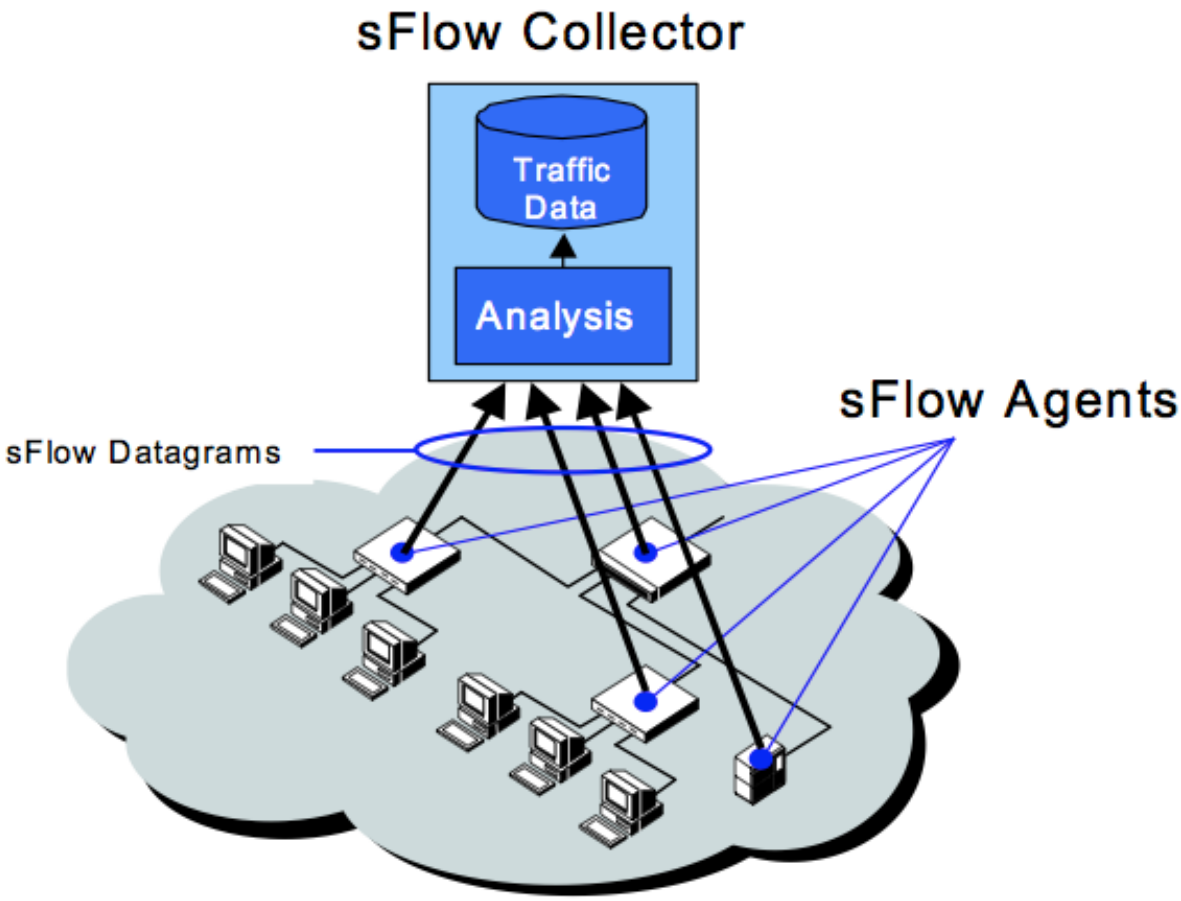
I can give no better bird's eye explanation of sFlow than its webpage provides:

> *sFlow® is an industry standard technology for monitoring high speed switched networks. It gives complete visibility into the use of networks enabling performance optimization, accounting/billing for usage, and defense against security threats.*

sFlow is worked on by a consortium of mainstream network device vendors, including ExtremeNetworks, HP, Hitachi, etc. Since it's embedded in their devices, it provides a consistent way to monitor traffic across different networks. However, from the standpoint of a number of open source projects, I must mention that it's also built into OpenvSwitch virtual switch (which is a more robust alternative to a Linux bridge).

To provide end-to-end packet flow analysis, sFlow agents need to be deployed on all devices across the network. The agents simply collect sample data on network devices (including packet headers and forwarding/routing table state), and send them to the sFlow Collector. The collector gathers data samples from all the sFlow agents and produces meaningful statistics out of them, including traffic characteristics and network topology (how packets traverse the network between two endpoints).

To achieve better performance in high-speed networks, the sFlow agent does sampling (picks one packet out of a given number). A more detailed walkthrough of sFlow is provided here.



source: http://sflow.org

Now, since it is all about networking, why should you care about it if you're in the cloud business? While sFlow itself is defined as a standard to monitor networks, it also comes with a "host sFlow agent." Per the website:

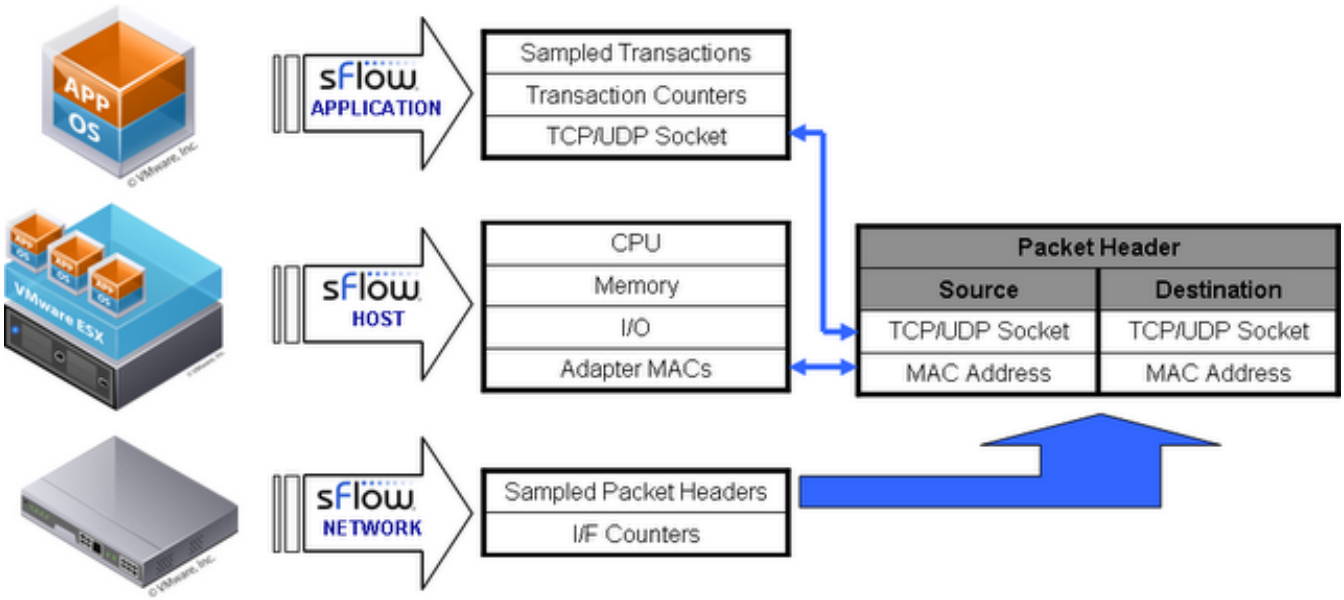
> *The **Host sFlow agent** exports physical and virtual server performance metrics using the sFlow protocol. The agent provides scalable, multi-vendor, multi-OS performance monitoring with minimal impact on the systems being monitored.*

Apart from this definition, you need to know that host sFlow agents are available for mainstream hypervisors, including Xen, KVM/libvirt, and Hyper-V (VMWare to be added soon) and can be installed on a number of operating systems (FreeBSD, Linux, Solaris, Windows) to monitor applications running on them. For the IaaS clouds based on these hypervisors, it means that it's now possible to sample different metrics of an instance (including I/O, CPU, RAM, interrupts/sec etc.) **without even logging into it**.

To make it even better, one can combine the "network" and "host" parts of sFlow data to provide a complex monitoring solution for the cloud:

> *A server exporting sFlow performance metrics includes an additional structure containing the MAC addresses associated with each of its network adapters. The inclusion of the MAC addresses provides a common key linking server performance metrics (CPU, Memory, I/O etc.) to network performance measurements (network flows, link utilizations, etc.), providing a complete picture of the server's performance.*

This diagram shows the so-called sFlow host structure—the way of pairing host metrics with corresponding network metrics. You can see that to relate the "host" and "network" data, mapping of MAC addresses is used. Also, sFlow agents can be installed on top of the instance's operating system, to measure different applications' performance based on their TCP/UDP ports. Apart from combining host and network stats into one monitored entity, sFlow host structures also allow you to track instances as they migrate across physical infrastructure (by tracking MAC addresses).



source: http://sflow.org

What does this all mean for OpenStack?

With the advent of Quantum in the Folsom release, a virtual network device will move from a Linux bridge to OpenvSwitch. If we add KVM or Xen to the mix, we will have an applicable framework to monitor instances themselves and their virtual network topologies as well.

There are a number of sFlow collectors available. The most widely used seem to be Ganglia and sFlowTrend, which are free. While Ganglia is focused mainly on monitoring the performance of clustered hosts or instance pools, sFlowTrend seems to be more robust, adding network metrics and topologies on top of that.

Since sFlow is so cool, it's hard to believe I couldn't find any signs of its presence in the OpenStack community (no blueprints, discussions, patches, etc.). On the other hand, sFlow folks have already noticed a possible use case for their product with OpenStack.

Conclusion

Cloud infrastructures have many layers and each of them requires a different approach to monitoring. For the hardware and service layer, all the tools have been there for years. And OpenStack itself follows well-established standards (web services, WSGI, REST, SQL, AMQP), which prevents sysadmins from having to write their own custom probes.

On the other hand, OpenStack's ability to track the resources it governs is rather limited now—as shown above, only the compute part is addressed and other aspects like networking and block storage seem to be left untouched. For now the user must assume that the status of the resources reported by OpenStack is unreliable. Unless this area receives more attention from the project, I would advise you to monitor cloud resources using third-party tools.

As for monitoring the cloud resources' performance, sFlow seems to be the best path to follow. The standard is designed to scale to monitor thousands of resources without serious performance impact and all the metrics are gathered without direct interaction with tenants' data, which ensures privacy. The relationship between the host metrics and network metrics, plus the determination of network flows allows us to see how problems on a given host affect the whole network (and possibly other hosts).

To properly monitor all the layers of the architecture, it is currently necessary to employ a number of different tools. There is a great market niche now for startup companies to integrate tools such as sFlow into existing cloud platforms. Or maybe even better—provide one, complex solution, which could monitor all the cloud aspects on one console.

15 comments

15 Responses

1. Miguel Lavalle

   Piotr,

   Great posting. I think it is very complete and thorough. One aspect that you didn't cover, though, is the role that "old guard" enterprise monitoring suites might play in this picture. I am talking about suites like the ones sold by HP Software, BMC, CA, etc. These software suites have huge installed bases in the enterprise market. I know them well, having spent the last 15 years with

HP.

I think that by enabling “Monitoring performance of cloud resources” (as you call it above) with “old guard” tools, we can help drive the adoption of OpenStack by enterprise customers. Most of these customers have been using “old guard” tools for many years, feel comfortable with them and the last thing they want is another set of monitoring tools. As a consequence, if we enable them to monitor virtual infrastructure with the tools they already have, we might be removing one barrier for OpenStack adoption.

As a matter of fact, over the past 4 weeks I have been writing a prototype application that uses the Nova API and a couple of the HP tools (SiteScope and Operations Orchestration) to launch and monitor the performance of instances in OpenStack. I would love to show it to you and get your feedback.

Best regards

Miguel

September 30, 2012 16:13
Reply

- Piotr Siwczak

  Miguel,

  Agreed about your point on “old guard” tools. I omitted this area as I believe I don’t have relevant experience on commercial enterprise monitoring tools (always relied on free stuff).

  If it is possible, I would love to hear from you about the work you are doing in the field 😉

  -Piotr

  September 30, 2012 23:41
  Reply

2. Ruslan Kiianchuk

   Just to keep the post updated:
   Monitoring status of the user’s cloud ecosystem is now monitored by OpenStack Ceilometer project.

   September 19, 2013 15:25
   Reply

3. 

Kyle Sawyer

   Read it again, greate post! But I am a little confused about the difference between “User’s cloud ecosystem” and “Performance of cloud resources”. As sFlow is so cool, can the latter replace s the

former ? AFAIK ceilometer seems to be doing something like what sFlow does, collecting samples of different metrics, and producing meaningful statistics, but through reading other services' api.
Thanks!

November 30, 2013 08:29
Reply

4. Janusz

Great post,
It would be also great to consider options for HA deplyment monitoring. I'm actually starting to implement monitoring on my newly deployed Multinode HA environment based on Mirantis Fuel. I wonder which componnents should be monitored to ensure HA is working correctly.

March 6, 2014 05:03
Reply

Continuing the Discussion

1. 学习sFlow » 陈沙克日志

   [...] http://www.mirantis.com/blog/openstack-monitoring/ [...]

   September 12, 201206:38

2. The Data Center Journal Monitis Disrupts IT Monitoring, Again.

   [...] Monitis Inc., the enterprise-grade web and cloud systems monitoring company, today announced it will...e most feature-rich and comprehensive free hosted web and cloud monitoring solutions available, meaning that even the smallest company can gain insight into their web systems performance that was previously out of their reach. [...]

   November 11, 201222:20

3. OpenStack Monitoring – Part 1 Installing Sensu | Professional OpenStack

   [...] complex and have any number of service interdependencies which makes monitoring it difficult. The Mirantis blog does a much better job of describing this than I could. This series will cover installing and [...]

   May 26, 201321:21

4. 如何成为OpenStack工程师 | Way Forever

   [...] OpenStack 监控: http://www.mirantis.com/blog/openstack-monitoring/ [...]

   August 12, 201309:40

5. OpenStack Hacker养成指南 | M2M

   [...] OpenStack 监控: http://www.mirantis.com/blog/openstack-monitoring/ [...]

   August 12, 201311:41

6. Yet another way to monitor OpenStack

[...] is some interesting ways that you could read on Internet. One of them is from Mirantis and explains that you could monitor API with check_http method. In my opinion, [...]

August 22, 201312:07

7. OpenStack Hacker养成指南 | UnitedStack Inc.

   [...] OpenStack 监控: http://www.mirantis.com/blog/openstack-monitoring/ [...]

   September 5, 201301:01

8. OpenStack Hacker养成指南 « OpenStack中国社区

   [...] OpenStack 监控: http://www.mirantis.com/blog/openstack-monitoring/ [...]

   September 8, 201317:00

9. OpenStack Ceilometer 简介 | UnitedStack Inc.

   [...] 虽然计量和监控他们的数据相似，但是对数据的要求却 大相径庭，前者重点在Allocated，而后者重点在Utilization，Allocated这些信息能够很容易通过消费其它服务的notification消息来获得，比如通过 捕获compute.instance.create.start, compute.instance.delete.end这些Event可以比较准确的知道一个instance是否创建成功，是否删除失败，但是像Instance 的Memory Utilization, Disk Space Utilization这些数据是监控非常关心的，然而想获取这些数据却不容易，没有现成的API可以使用，写Agent，兼容性是个头疼的问题， 这也是虚拟化平台监控的难点之一，目前Ceilometer在这方面还是空白，由此在邮件列表中也引发的一次不愉快的争论。但是关于OpenStack的监控，Mirantis给出了一个解决方案，使用sFlow 来解决虚拟资源监控的难点，有兴趣可以阅读一下。 [...]

   September 11, 201317:57

10. OpenStack Hacker养成指南 | 新世纪Linux社区

    [...] OpenStack 监控: http://www.mirantis.com/blog/openstack-monitoring/ [...]

    February 7, 201404:41

Post a comment

Some HTML is OK

Name *(required)*

Email *(required, but never shared)*

Web

Post comment

or, reply to this post via trackback.

GET TRAINED WITH OPENSTACK!

Boston, MA.
OpenStack Bootcamp, Aug 12-15

Raleigh, NC.
OpenStack Bootcamp, Aug 19-21

Paris, France.
OpenStack Bootcamp, Aug 20-22

Dallas, TX.
OpenStack Bootcamp, Aug 26-29

Seattle, WA.
OpenStack Bootcamp, Sep 16-19

New York, NY.
OpenStack Bootcamp, Sep 16-19

Chicago, IL.
OpenStack Bootcamp, Sep 23-26

Munich, Germany.
OpenStack Bootcamp, Oct 7-10

Paris, France - OPENSTACK SUMMIT.
OpenStack Bootcamp, Oct 29-31

MOST POPULAR BLOG POSTS

- Cloud Prizefight: OpenStack vs. VMware
- OpenStack Networking Tutorial: Single-host FlatDHCPManager
- Configuring Floating IP addresses for Networking in OpenStack Public and Private Clouds
- OpenStack Networking – FlatManager and FlatDHCPManager
- Openstack Networking for Scalability and Multi-tenancy with VlanManager
- An OpenStack guy takes CloudStack for a test drive

BLOG ARCHIVE

August 2014

July 2014

June 2014

May 2014

April 2014

March 2014

- Earlier

SOFTWARE

DOWNLOADS
CORE OPENSTACK
KEY RELATED PROJECTS
PLUG-INS & DRIVERS

SERVICES

SOLUTIONS ENGINEERING
CUSTOM DEPLOYMENT
CUSTOM SUPPORT
DRIVERS

TRAINING

OPENSTACK CERTIFICATION
OPENSTACK BOOTCAMP
ONSITE TRAINING
OUR INSTRUCTORS
ONLINE RESOURCES

OPENSTACK:NOW

BLOG
NEWS
TUTORIALS

USE CASES

SAAS/WEB VENDORS
SERVICE PROVIDERS
ENTERPRISE CLOUD

INFRASTRUCTURE VENDORS

COMPANY

OUR PHILOSOPHY
CUSTOMER SUCCESSES
CAREERS
LEADERSHIP
INVESTORS
BOARD OF DIRECTORS
PARTNERS
PRESS CENTER
PRIVACY POLICY

Mirantis Inc.
Pure Play OpenStack.

615 National Avenue, Suite 100 Mountain View, CA 94043
+1-650-963-9828

loading