**原型系统说明文件**

## 实现对Openstack的监控

Openstack的监控分为两个部分，分别为传统的物理机及服务监控、虚拟资源监控。该监控系统使用到的开源软件包括Nagios、Ganglia、Graphite、Graphios、Grafana。虚拟资源监控方面由于Openstack的Telemetry项目的发展，我们可以使用其中的Ceilometer子项目来采集性能数据。

该监控系统的核心是Nagios。不同于其他监控软件把所有能够想到的监控项目都写到系统里，组成一个巨大的应用程序，认为能够满足用户的所有需求，并提供了一个多个层级嵌套的、复杂的用户界面。Nagios没有内部监控逻辑，并假设对用户的监控环境或方式不甚了解。Naigos是一个框架，其核心是一个调度程序。无论用户的小监控程序使用何种语言编写的，都可以将它们完全托管到Nagios中。用户只需要配置很少的参数，Naigos就能帮你调度那些小的监控程序（或插件），收集它们返回的状态，确保通知发送给正确的人，并根据规则按需升级。Nagios同时能够对所有插件的状态情况进行跟踪，记录状态改变的日期和时间，并内置了历史报告的功能。Nagios提供了很多挂钩，使数据的输入和输出都很方便，所以它能够向图形程序（比如RRDTool和MRTG）提供实时数据，并且能够很方便地与其他监控系统协作，无论是向上集成（输出数据）或是向下集成（接收数据）。**[3]**

主机和服务是Nagios最重要的两个对象。任何能通过TCP协议通信的设备都可以被定义为主机。每个主机都可以定义多个服务。通常，一台独立的主机会运行多个应用—至少有多个元素应当被监视，但是主机自身只会处于启动或是宕机状态，或者说是不是可用。因此，Nagios允许用户对每台主机定义唯一的主机检测机制以及多个服务检测。Nagios通过这些主机和服务检测的定义，对插件进行调用以获得主机或服务的状态。通常一台主机宕机，其上面的服务必定不可用。对于主机/服务的假设使Nagios能够很容易地搞清楚服务与主机的依赖关系。当Nagios运行插件对某台主机上的某个服务进行检测时，如果插件返回了一个错误状态，Nagios首先就会对该主机进行检测。如果主机检测也返回了错误状态，则表示主机不可用，此时，Nagios只会通知用户主机宕机，同时推迟所有服务检测，直到主机可用为止。**[3]**

由于Nagios自带的插件只支持少量的监控指标，我们需要对Nagios的插件进行扩展。编写Nagios的插件十分简单。字面上来说，程序作为插件唯一要遵守的原则就是提供退出代码。任何能够提供退出代码的编程或脚本语言都可以用来编写Nagios插件。插件的具体工作步骤如下所述：

1. 从目标主机上收集信息，比如当前负载或者它的首页文件index.html；

2. 将收集的信息与期望状态或阈值进行比对；

3. 根据比对结果，提供对应退出代码。

服务定义将指定使用的插件以及其他相关信息，比如服务名称、检测执行的频率等。Nagios负责调度并执行服务定义中指定的插件，并有选择地为插件提供阈值进行比较。当插件完成了它的“工作”后，会返回4个退出代码之一：0表示“正常”，1表示“警告”，2表示“严重”，3表示“未知”。Nagios解析插件的退出代码并据此做响应。当Nagios接收到有问题的代码时，它会更新服务状态并根据配置决定是否联系相关人员。

通常，除了退出代码，插件还会通过标准输出提供一行文本信息，Nagios将上述信息作为可读的汇总信息。汇总信息将会全部显示在Web界面相应的状态字段中。对于不喜欢解释退出代码含义的人类而言，这个功能能够很方便地回传服务的相关信息。此外，插件的标准输出还可以包含性能数据。性能数据跟在文本信息和管道符之后。

对于物理设备的监控，Ganglia提供了优秀的实现方案。Ganglia在每台物理机上安装一个gmond守护进程，该进程采集物理机的性能信息并发给其他节点，并只占用了少量的系统资源，不对性能造成明显的影响。Ganglia的安装包带有一个叫做check\_ganglia的脚本，该文件是用python编写的获取Ganglia性能信息的Nagios插件，可以直接使用。Ceilometer是Openstack的一个子项目，负责收集Openstack内部的性能信息，提供给监控、计费等服务使用。该项目经过简单的安装和配置之后就能够使用。但虚拟机的有些性能信息需要额外的配置，并在虚拟机有相关活动后（如发送网络包）才会收集。Ceilometer提供了SDK可供Python开发者使用，开发者需要查阅官方文档并编写对应的Nagios插件。

该监控系统将Nagios部署到一个主机上，称为监控主机，并修改Nagios的配置，为Nagios添加了准备好的check\_ganglia和check\_vm插件。监控系统在每个被监控的物理机上安装了Ganglia的gmond守护进程，在监控主机上安装了Ganglia的gmetad守护进程，并使用gmetad汇总所有的gmond守护进程发来的物理机性能数据，check\_ganglia插件再从gmetad获取性能数据。Check\_vm直接调用了Openstack的Ceilometer服务的API获取Ceilometer收集到的虚拟机数据。

Nagios并没有存储插件采集到的性能信息的历史数据，因此也无法绘制图表。事实上，性能数据对Nagios没有意义，Nagios通过执行service\_perfdata\_command命令关联的某一脚本来把性能数据输出到外部图形系统，如RRDTools和Graphite。该监控系统使用了Graphite作为外部的图形系统。Graphite是一个专门用于存储监控性能数据的开源软件，但不负责性能数据的采集，因而能与Nagios配合使用。Graphite存储按照时间顺序排列的数值数据，并且时间越久远，数据的密集度越低。将Nagios采集到的性能数据存储到Graphite的数据库里还需要安装Graphios，该工具能够将Nagios的性能数据发送Graphite、InfluxDB等多种存储系统。Nagios需要要存储性能数据的服务的定义里面加入\_graphiteprefix或\_graphitepostfix来定义该性能数据的层级结构，如host1.disk.free或host2.disk.free。性能数据会按照层级结构被存储到Graphite，层级结构决定了读取数据的方式。获取数据时用户可以使用host1.disk.free获取host1的磁盘空闲空间，使用\*.disk.free获取所有主机的磁盘空闲空间的数据。Graphite也提供了查看性能图表的功能，但是功能太过简单不能满足需求。为了实现一个令人满意的监控系统，一个交互更加简单、自定义能力更强的图表系统是必须的。

Grafana是一个基于Django框架的数据可视化系统，以交互简单、功能强大的方式提供创建和分享仪表盘的功能，并支持Graphite、InfluxDB等多种数据后端。Grafana的仪表盘指的是一个web页面上显示的图表。仪表盘可以包含多个面板，每个面板包含一张图表。可以为每个面板制定不同的数据来源以显示不同的监控信息，此外图表的样式也有很多的自定义选项，如柱状图、折线图、颜色、粗细等。仪表盘还支持模板变量简化仪表盘的创建。用户可以定义一个变量如host并指定host的取值范围，仪表盘会自动为host的不同取值创建多个面板。此外，Grafana还提供了API供开发者使用，开发者可以通过API创建和修改仪表盘。

这样监控系统所需的数据采集、存储、展示都齐全了。

## 配置管理系统

有了监控系统，用户还需要把要监控的主机和服务添加到系统中。这时用户必须手动编辑Nagios的配置文件，而数据中心的成百上千台主机的配置对管理员来说可能是灾难性的，因此管理员需要一个能简化配置且用户友好的管理系统。

系统的开发语言是Python。Python是一个能够高效开发的语言**[6][7]**。该监控系统使用Flask框架实现了一个web应用程序来解决这个问题。Flask是一个简单的Pyhton的web应用框架，使用扩展的方式增加其功能，具有很强的扩展性。**[8]**该应用程序提供定义主机和服务、配置监控主机、配置用于告警的邮件服务、查看告警的功能。Nagios的主机和服务的定义方式在主机数目增加时会变得相当繁琐，但却十分适合让程序来处理。当用户在管理系统定义好主机和服务并点击同步到Nagios后，管理系统会根据用户的定义生成Nagios的配置文件，并用这些文件覆盖掉原有配置。

监控主机之前必须在被监控主机安装和配置gmond守护进程，逐一地在主机上手动安装和配置将会是一件耗时耗力的工作。管理系统使用Ansible来自动化安装和配置。Ansible是一个开源的自动化运维工具，能对主机进行批量配置和部署。Ansible用YAML定义要处理的主机、任务和配置，具有良好的可读性。其操作具有幂等性，多次执行产生的结果与一次执行的产生的结果相同。Ansible的playbook定义了在每台主机上执行的任务和顺序。Ansible使用SSH连接到目标主机，因此还需要提供目标主机的SSH登录账号和密码。当用户点击部署主机时，管理系统就会使用目标主机的SSH登录账号和密码以及预先定义好的playbook对目标主机进行部署。**[9]**

Nagios在采集到性能数据后，会把数据的数值和警告、严重的阀值比较，当数值超过阀值时Nagios就会发出相应的通知。Nagios的通知的具体行为可以自定义。在该监控系统中，Nagios决定发送通知时，就会通知的内容以JSON的格式发送到管理系统的一个路径上，管理系统会将通知存储到数据库，管理员可以在管理系统上看到这些通知。如果设置邮件服务，管理系统也会将通知发送到管理员的邮箱。

管理系统还使用Grafana的API为用户定义的主机和服务创建了仪表盘。