What

项目概述

我们这个项目是基于 2021 年的 DisGraFS 展开，DisGraFS 是一个分布式图文件系统，但是缺少必要的监控组件。本项目将致力于在 DisGraFS 上安装分布式监控系统，了解系统的运行状态，保证服务的正确且稳定运行。

分布式监控

什么是分布式监控?

首先，分布式监控是部署在分布式系统内的监控组件，它可以监视和显示集群中各节点的状态信息，它有运行在各个节点的进程，可以采集不同节点之间的通信消息，采集各个节点的资源利用率，最后将采集到的数据汇总到一个数据库，进行分析处理后以直观的图形化界面进行呈现。

什么是 DisGraFS?

DisGraFS 是一个分布式图文件系统，他将图结构的思想应用于分布式文件系统上面，使其兼顾图文件系统方便用户快速搜索、模糊搜索、查找相关文件的特点，以及分布式文件系统的海量文件存储、云存储的特点。

他共分为五个部分，对于它的运行过程与执行逻辑，我们将会在 How 这一部分进行具体说明。

DisGraFS 存在的问题

首先是他们的远程存储集群实际没有实现。按照 DisGraFS 本来的设想，当整个分布式文件系统搭建运行起来以后，用户只需要进入到 DisGraFS 提供的网页，登陆验证通过后，即可看到整个图文件系统，并在网页上实现文件的上传、移动和删除。而 DisGraFS 在具体实现中，为了简化工作量——他们的工作量确实非常大——他们将远程存储节点进行了简化，远程存储集群仅有一台服务器，而本地用户需要将这个存储集群挂载到本地，才能实现文件的传输。我们计划完成他们最初的设想，重新搭建存储集群，这将是一个真正具有多个存储节点的、远程的集群，用户的所有操作仅需要在网页端执行。

第二，DisGraFS 并不具备可运维性，也没有进行可用性的保障，它的日志文件分散，开发人员很难同时获得整个系统的运行状态，也很难看到各个节点的资源使用情况，甚至无法及时了解某个物理节点是否还处于连接状态，这使得它的可用性不能得到保证。这也是我们要搭建监控组件的原因。

Why

针对上面提到的问题，为了提供一个用户友好、使用起来优雅便捷的分布式文件系统，我们要修改目前 DisGraFS 的存储集群，把它部署到远程。这样实现后，我们要做的监控将实现更大的作用，原本位于本地的、只有一台的存储节点变成了远程的、多节点的存储集群，对于它们运行状态的获取离不开实时监控系统，而利用监控到的运行信息，将有助于索引服务器根据用户的可用性要求，对文件进行切片存储。

当然，监控的意义不止如此，监控在整个系统的运行维护中发挥着比较大的作用，它能帮助运维工作者掌握系统的实时状态，发现影响性能和稳定性的模块，预知可能出现的故障，以及发生故障时的及时反应。

对于分布式系统而言尤为如此，分布式系统是由多台机器协作共同完成任务，集群的内部关系复杂，一旦出现状况，问题的定位是很困难的。虽然已经有很多日志监控文件，但是集群中的日志分散、数据量大，查找起来并不容易。

对于 DisGraFS 而言，我们从去年参与这个项目的学长处得知，DisGraFS 存在系统稳定性方面的问题，他们去年在最后演示运行时出现过两次系统崩溃的情况。我们初步怀疑问题出在节点间通信以及不同节点的内容同步上，尤其是指存储集群和图数据库的内容同步。所以，我们也希望通过在 DisGraFS 上部署监控系统，发现影响系统稳定性的因素，进行优化处理。

此外，在复现 DisGraFS 的过程中，我们除了感受到这个项目的创新性以及学长们工作量的巨大，也感受到 DisGraFS 部署的复杂，原项目并没有提供一个完整清晰的部署文档，好多细节需要自己摸索尝试，手动模拟其源代码的执行流程。针对这一点，我们还将致力于整理出一份更完善的部署文档。

How

我们的项目分为四个步骤：

第一步，复现 DisGraFS，学习其代码逻辑，找到源码中可以部署监控的位置。

第二步，利用 Prometheus 部署监控，对监控到的信息进行处理，最后用 Grafana 进行图形化呈现。

第三步，搭建远程分布式存储集群，实现远程通信和远程文件传输

第四步，根据监控到的信息，对 DisGraFS 进行性能上的优化

其中，第二和第三步我们初步打算分工同时进行。

现在，我们已经基本完成了第一步骤，即复现 DiaGraFS。接下来，我简单讲一下 DisGraFS 最终成品的实现逻辑。

首先是**中央索引服务器**。DisGraFS 中央索引服务器与 Neo4j 图数据库共同构成中央节点，主要负责接收其他节点的消息和对图数据库的修改与维护。

消息接收来自三个方向：网页端、存储集群和计算集群。中央索引服务器通过对接收到的消息进行分解和处理。实际复现中，我们用一台服务器完成上述操作。

存储集群。在 DisGraFS 中，文件被存储在基于 JuiceFS 的分布式文件存储集群当中。JuiceFS 是一款面向云环境设计的高性能共享文件系统。它采用「数据」与「元数据」分离存储的架构，从而实现文件系统的分布式设计。创建完毕后，只要服务器处于开启状态，任何拥有 Redis 数据库密码的 PC 皆可利用 JuiceFS 客户端将已创建的文件系统挂载到本地，实现跨平台、跨地区的不同主机上同时挂载读写。这里也可以看出 DisGraFS 实际的实现还是要求用户要将存储节点挂载到本地，形成一个所谓的客户端，这一项增加了用户的使用难度，这也是我们要优化的部分。

网页端。web端主要分前置界面和主要功能页面。用户在前置界面登录，连接至中央服务器。登陆后跳转到主要功能页面，用户在此界面可以启动客户端、自动挂载 JuiceFS 以及连接到储存端服务器，也可以在此界面实现查看文件关系，打开文件，删除文件等操作。

计算集群。计算集群的实现，使用到了 Ray。Ray 是一个分布式高性能计算框架，它由一个 head 和多个 worker 节点构成，架构如这个图所示，它有 Global Scheduler 负责全局调度，将任务分发到不同的 worker。在 DisGraFS 中，利用 Ray 这样一个分布式计算框架，搭建分布式计算集群，对存储集群中的文件进行打标签。

Ray 的搭建需要所有节点在同一局域网内，而 Ray 的 head 节点需要保持运行，才能保证其他 worker 节点的连接，所以我们把 head 节点部署在一台服务器上，worker 节点目前位于我们本地电脑的虚拟机。局域网的实现是通过搭建 vpn，将本地接到服务器的内网。

接下来，我们将具体演示一下 DisGraFS 的运行。

这是我们项目三个步骤中的第一步，第二步，就是部署监控。

我们准备使用 Promethues，这是一个开源监控系统和警报工具包，它将实时的指标数据（metrics）记录并存储在通过 Http 拉取模型构建的时间序列数据库中，有着较灵活的询问功能和实时告警功能。它的架构如图：

Prometheus Server 本身就是一个时序数据库，将采集到的监控数据按照时间序列的方式存储在本地磁盘当中。但是本地存储也意味着 Prometheus 无法持久化数据，无法存储大量历史数据，同时也无法灵活扩展和迁移。

不过 Prometheus 定义了两个标准接口，让用户可以基于这两个接口对接将数据保存到任意第三方的存储服务中，我们选用 InfluxDB。InfluxDB 是一个由 InfluxData 开发的开源时序型数据库，着力于高性能地查询与存储时序型数据，它自带各种特殊函数，使数据统计和实时分析变得十分方便。最后，通过 Grafana 来实现监控数据的可视化。Grafana 是一个跨平台的开源的度量分析和可视化工具，可以通过将采集的数据查询然后可视化的展示，并及时通知。

我们项目的第三步，搭建远程分布式存储集群，实现远程通信和远程文件传输。我们将参考 2020 年的一个项目——dontpanic，这个项目实现了容器化服务器端、高效的文件传输以及高可用性，我们初步打算使用容器，部署在云服务器中，容器化的好处就在于启动和销毁一个存储节点比较便捷，方便了部署和维护。存储集群与中央索引服务器进行信息交互。

当用户在网页端发出文件上传命令后，执行对用户文件的分析，通过网络传递信息至索引服务器。索引服务器根据文件信息以及用户对可用性的需求，对已经存在的计算节点进行调度，最后，将用户文件进行切片上传，考虑到可用性的问题，这个过程中会对切片进行一定数量的复制，保证如果某几个存储节点不可用，用户仍然可以获取其所需文件。下载时，索引服务器根据文件的记录信息，找到存储位置，进行整合后下载给用户。

最后，是我们项目的第四步，我们将根据监控采集到的结果，定位哪些节点存在问题，优化 DisGraFS，主要是其稳定性，或者可能优化其架构。

Q&A