本小节就生产部署CockroachDB集群的过程提供一些重要的建议。

集群拓扑

术语

下面就CockroachDB一些基础的术语进行简单的介绍。

| **英文** | **中文** | **简介** |
| --- | --- | --- |
| Cluster | 集群 | CockcroachDB部署的内容，在逻辑上视为一个独立的应用。 |
| Node | 节点 | 运行CockroachDB的一台单独的机器，多个节点组成一个集群。 |
| Range | - | CockroachDB所有用户数据（包括数据表、索引等等）和几乎所有的系统数据都存储在一个巨大有序的key-value集合。根据连续的key划分成多个区间，每个区间是一个Range，因此每个key只会出现在一个range中。 |
| Replica | 副本 | CockroachDB对每个Range做数据冗余（默认情况下是3份副本），并分别存储在不同的节点上。 |
| Range Lease | Range租期 | 每个Range所有副本当中只有一个副本持有Range租期，租约持有者接收和协调关于该Range的所有读写请求。 |

拓扑结构建议

一个物理机器只运行一个节点

在一台独立的机器上运行一个节点，由于CockroachDB跨节点进行数据冗余，单台机器上运行多个节点会增加由于机器故障而导致的数据丢失风险。若一台机器存在多个存储设备，用户希望充分利用多个存储设备提高IO并发度，可以使用--store的标签来配置单个节点多个存储设备，而不是一个存储设备启动一个节点。详见[启动节点](http://doc.cockroachchina.baidu.com/#deploy/start-a-node/)

部署在单数据中心

* 应对单节点故障

至少启动3个节点，使得默认情况下的[三副本数据冗余策略](http://doc.cockroachchina.baidu.com/#deploy/configure-replication-zones/#view-the-default-replication-zone)生效，从而保证单个节点丢失的情况下，集群绝大多数（此时为2/3）节点仍然存活，能够维持集群的可用状态。

* 应对双节点故障

至少启动5个节点，并配置[冗余副本数量到5](http://doc.cockroachchina.baidu.com/#deploy/configure-replication-zones/#edit-the-default-replication-zone)，从而保证双节点丢失的情况下，集群的3/5节点仍然存活，能够维持集群的可用状态。

部署在多数据中心

* 能够应对单数据中心故障  
  部署节点到至少3个不同的数据中心，并且每个节点设置--locality标记保证数据在跨数据中心之间均匀分布。这种情况下，若1个数据中心出现故障无法使用，另外2个数据中心仍能保证集群的可用性。
* --locality标记的更多应用细节  
  在启动节点的时候可以使用--locality指定节点的“位置”，例如--locality=region=west,datacenter=us-west-1。key-value对从最大到最小排序，并且key和key-value对的顺序在所有节点上必须相同。
  + 按照优先级顺序，CockroachDB根据--locality标记的“位置”情况，让数据的多个副本尽可能均分到不同位置。更多的副本分布策略可以查看：[复制区域](http://doc.cockroachchina.baidu.com/#deploy/configure-replication-zones/#replication-constraints)
  + 当“位置”之间存在较高延迟的时候，CockroachDB能够根据数据访问的集中程度，将Range租期优先分配给拥有更多请求的“位置”对应的节点，即 [Follow-the-workload](https://www.cockroachlabs.com/docs/stable/demo-follow-the-workload.html) 机制，进而减少网络开销、提升读性能。在多于3个数据中心的集群，为了充分发挥Follow-the-workload机制的优势，需要提高[冗余副本数量](http://doc.cockroachchina.baidu.com/#deploy/configure-replication-zones/#edit-the-default-replication-zone)，匹配数据中心的数量。
  + 配置--locality也是使用[表分区](http://doc.cockroachchina.baidu.com/#deploy/define-table-partitions-(enterprise)/)功能和[**节点地图**](http://doc.cockroachchina.baidu.com/#deploy/enable-node-map-(enterprise))企业特性的先决条件。
* 在运行5节点或更多节点的集群时，对于重要的内部数据增加到5份副本是最安全的方式。为保证集群的可用性，内部数据的range必须始终保持绝大多数，保证副本的可用性。

**TIPS:** 关于集群容错与自动恢复的更多内容，可以查看[Fault Tolerance and Automated Repair](https://www.cockroachlabs.com/docs/stable/training/fault-tolerance-and-automated-repair.html)

硬件

硬件建议

每个节点需要配备必要的CPU、内存、网络和存储等资源，在部署集群前需要检查各个硬件设备。 最低限度的情况下，每个节点应该有2GiB的RAM和一个完整的内核。数据量大、复杂的工作负载、高并发和高性能场景下，需要更多的硬件资源。

**WARNING:** 尽量避免使用配置shared-core或"burstable" VMs。

追求最好的性能：

* 存储设备方面相比于HDD，更加推荐使用SSD
* 增加CPU的核心数量往往比增大内存获得更高的性能收益

追求最好的恢复能力：

* 增加节点的数量而不是质量，有利于在节点丢失的时候减少数据重新均衡分布的代价
* 不同于集群部署在云端，云服务商往往会提供必要的存储备份服务，如果集群部署在用户本地机器上，则推荐通过[配置复制区域](http://doc.cockroachchina.baidu.com/#deploy/configure-replication-zones/)将冗余副本数量从默认的3提升到5，因为本地机器需要冒着更大风险、付出更多代价去应对节点故障导致的数据丢失。

**Warning:** 改变默认的[复制区域](http://doc.cockroachchina.baidu.com/#deploy/configure-replication-zones/#edit-the-default-replication-zone)，不会自动应用于已经存在的复制区域。为保证集群的可用性，内部数据的系统range必须始终保持绝大多数，保证副本的可用性。因此，如果要增加默认的复制因子，就要确保对重要内部数据的复制因子也增加。

针对云端部署的建议

以下的建议是Cockroach Labs成员内部在云端测试后得出的配置建议，仅供参考，一切以实际测试结果为准。

AWS

* 使用m(general purpose)、c(compute-optimized) 或 I(storage-optimized) 的机器，Cockroach Labs内部测试使用的是m3.large机器，每台机器配备了2 vCPUs和7.5 GiB的内存。
* **不推荐**使用"burstable"的t2机器（单核配置）
* 使用 [Provisioned IOPS SSD-backed (io1) EBS volumes](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html#EBSVolumeTypes_piops) 或是 [SSD Instance Store volumes](https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ssd-instance-store.html)

Azure

* 使用存储优化的[Ls-series](https://docs.microsoft.com/en-us/azure/virtual-machines/linux/sizes-storage)VMs，Cockroach Labs内部测试使用的是Standard\_L4sVMs，每台机器配备了4 vCPUs和32 GiB的内存。
* 使用[Premium Storage](https://docs.microsoft.com/en-us/azure/virtual-machines/windows/premium-storage) 或是 配备ex4文件系统的SSD存储（不要使用Windows的ntfs文件系统），注意[Premium Storage的空间大小会对IOPS有影响](https://docs.microsoft.com/en-us/azure/virtual-machines/windows/premium-storage#premium-storage-disk-limits)。
* 如果选择本地SSD存储，要注意重启机器以后文件系统将重置为ntfs，需要确保自动监控生效、并重新配置磁盘的文件系统。
* **不推荐**使用"burstable"的[B-series](https://docs.microsoft.com/en-us/azure/virtual-machines/linux/b-series-burstable)机器（单核配置）。此外Cockroach Labs曾经在A-series VMs上测试遇到数据损坏的问题，在D-series VMs上测试遇到磁盘性能不稳定的问题，值得用户关注。

Digital Ocean

* 使用除了只有1 GiB内存以外（不满足最低硬件要求）的[droplets](https://www.digitalocean.com/pricing/)机器。所有Digital Ocean的机器均配备的是SSD存储。

GCE

* 使用n1-standard或n1-highcpu [predefined VMs](https://cloud.google.com/compute/pricing#predefined_machine_types) 或 [custom VMs](https://cloud.google.com/compute/pricing#custommachinetypepricing)。Cockroach Labs内部测试使用的是custom VMs，每台机器配备了8 vCPUs和 16 GiB内存。
* **不推荐**使用f1或g1的shared-core机器（单核配置）
* 使用[Local SSDs](https://cloud.google.com/compute/docs/disks/#localssds)或是[SSD持久设备](https://cloud.google.com/compute/docs/disks/#pdspecs)，注意[SSD持久设备的IOPS受到存储空间大小和机器CPU数量的影响](https://cloud.google.com/compute/docs/disks/performance#optimizessdperformance)

安全性

非安全模式的集群存在很大的风险：

* 集群对任何客户端是开放的，能够访问集群任意节点的IP地址。
* 任何用户，甚至root用户都能够不需要密码访问集群。
* 任何用户均可以以root用户接入集群，可读写集群中的任何数据。
* 没有网络加密或认证，因此欠缺机密性。

因此我们强烈建议使用TLS加密技术去验证节点、客户端的身份，同时需要加密节点和客户端之间传输的数据。使用[cockroach cert](http://doc.cockroachchina.baidu.com/#deploy/create-security-certificates/)命令或是[openssl](https://www.cockroachlabs.com/docs/stable/create-security-certificates-openssl.html)生成部署所用的安全证书。以上两种方式都需要以下文件：

* 一对CA证书和密钥文件，用于加密其他认证。
* 为节点创建独立的证书和密钥文件，通用名为node。
* 为客户端和用户创建独立的证书和密钥文件，通用名设置为用户的名字，默认用户为root。

CockroachDB还提供了[密码验证](http://doc.cockroachchina.baidu.com/#deploy/access-management/manage-users/#user-authentication)，但是从安全上考虑不推荐使用。

网络

网络相关flags

启动节点的命令包括两个主要的flags：

* --host代表对客户端和其他节点暴露的一个或多个IP访问地址
* --advertise-host 其他节点使用该地址访问本节点

这两个flag相互之间组合的效果如下：

| **---** | **不指定--host** | **指定--host** |
| --- | --- | --- |
| 不指定--advertise-host | 节点使用所在机器的所有网卡IP地址供其他节点和客户端访问，同时通知其他节点将规范hostname作为访问前者的地址 | 节点使用--host指定的IP地址或hostname，并将该地址通知给其他节点 |
| 指定--advertise-host | **【推荐】**节点使用所在机器的所有网卡的IP地址供其他节点和客户端访问，同时通知其他节点使用作--advertise-host指定的hostname为访问地址 | 节点使用--host指定的IP地址或hostname，同时通知其他节点使用--advertise-host指定的主机名作为前者的访问地址 |

**TIPS:** 当使用hostname的时候，无论是--host或是--advertise-host指定的，务必确保该hostname能够被DNS或是etc/hosts文件正确解析。

单一网络建立集群

根据网络是否为私有网络，有不同的部署建议。在私有网络中，机器的IP地址受限，不能接入公网。如果该网络为外部访问受限的私有网络，集群将获得更高的安全性和更小的网络延迟。

| **私有与否** | **部署建议** |
| --- | --- |
| 是 | 启动节点时将--host设置为私有网络地址，不要指定--advertise-host，其他节点将使用该节点的私有网络地址去访问该节点，同时私有网络里的负载均衡器也将使用该地址。 |
| 否 | 启动节点时将--advertise-host设置为一个静态公网IP地址，不要指定--host，其他节点将使用该公网IP地址去访问该节点，而私有网络里的负载均衡器则可使用任意能访问到该节点的IP地址。  如果负载均衡器使用集群外部的网络，可能需要配置防火墙，使得来自该外部网络的访问请求能够访问到集群。 |

跨网络建立集群

根据不同网络之间的节点能不能相互之间访问，有不同的部署建议。

| **节点之间能够跨网络访问** | **部署建议** |
| --- | --- |
| 是 | 在同一个云环境的网络下，是一种典型的情况，可参考[单一网络配置建议](http://doc.cockroachchina.baidu.com/#deploy/production-checklist/#cluster-on-a-single-network) |
| 否 | 在不同的云环境下，是另一种典型的情况，此时我们需要配置[VPN](https://en.wikipedia.org/wiki/Virtual_private_network)、[VPC](https://en.wikipedia.org/wiki/Virtual_private_cloud)、[NAT](https://en.wikipedia.org/wiki/Network_address_translation)或是其他能够打通两个网络的技术方案。之后为每个节点配置--advertise-host，不需要指定--host，其他节点将使用该公网IP地址去访问该节点，而私有网络里的负载均衡器则可使用任意能访问到该节点的IP地址。 |

负载均衡

在集群当中每个CockroachDB节点都是一个平等的SQL访问网关。在负载均衡的时候，需要考虑节点的性能和可靠性：

* 性能

均衡各个节点的客户端访问量，避免单个节点处理过多的访问请求以至于影响集群的性能。

* 可靠性

负载平衡器将客户端健康与单个CockroachDB节点的健康解耦。为了避免客户端请求路由到故障或是未准备好处理请求的节点，负载均衡器需要使用[CockroachDB 's readiness health check](http://doc.cockroachchina.baidu.com/#deploy/monitoring-and-alerting/overview/#health-ready-1)

**TIPS:** 单一负载均衡器能够处理节点故障的情况，但其本身对整个系统来说也是一个故障的风险点。因此需要设置多个负载均衡器，通过浮动IP或是DNS为客户端选择合适的负载均衡器。

关于负载均衡器的选择，我们有以下建议：

| **部署环境** | **选择建议** |
| --- | --- |
| 内部部署 | 使用HAProxy |
| AWS | 使用亚马逊提供的负载均衡服务 |
| Azure | 使用Azure提供的负载均衡服务 |
| Digital Ocean | 使用Digital Ocean提供的负载均衡服务 |
| GCE | 使用GCE提供的TCP代理负载均衡服务 |

监控和告警

尽管CockroachDB有多种[保障集群可用性的设计](https://www.cockroachlabs.com/docs/stable/high-availability.html)，对于集群健康和性能的持续监控仍然是有必要的，预先针对需要调查和干预的事件创建报警规则，以便出现问题的时候能够第一时间获知并及时处理。 详细的监控信息和告警设置，可参考[监控和报警](http://doc.cockroachchina.baidu.com/#deploy/monitoring-and-alerting/overview/)。

时钟同步

CockroachDB需要中等强度的时钟同步机制以维持数据的一致性。当一个节点检测到自身的机器时间与集群中至少50%节点的机器时间之间的误差值超过集群最大允许时间误差值（默认500ms）80%的时候，该节点会自动停止。这能够避免违反数据一致性导致读写旧数据的风险。通过在每个节点上运行NTP或其他时钟同步软件，来防止时钟漂移得太远是很重要的。

值得注意的一个罕见情况是节点的时钟在节点检测到之前突然跳过最大偏移量。虽然看起来极不可能，但是这可能发生，例如在VM内运行CockroachDB并且VM管理程序决定将VM迁移到具有不同时间的另一个硬件环境的时候。在这种情况下，节点的时钟变得不同步和节点自发停止之间可能会有一个小的时间窗口。在此窗口期间，客户端可以读取旧数据并基于旧数据写入衍生的数据。

以下是关于时间同步相关的部署建议：

| **部署环境** | **选择建议** |
| --- | --- |
| 内部部署 | 将NTP与Google的外部NTP服务配合使用 |
| AWS | 使用亚马逊时间同步服务 |
| Azure | 关闭Hyper-V时间同步，将NTP与Google的外部NTP服务配合使用 |
| Digital Ocean | 将NTP与Google的外部NTP服务配合使用 |
| GCE | 将NTP与Google的外部NTP服务配合使用 |

**TIPS**: 在多数情况下，我们推荐将NTP与Google的外部NTP服务配合使用，原因是Google提供了闰秒涂抹的技术。如果你采用了不提供闰秒涂抹的NTP服务，你必须手动配置客户端、并保证在每台机器上均以相同的方式运行。

缓存和SQL内存大小

（changed in v1.1）默认情况下，每个节点的缓存大小和SQL内存大小均为128 MiB，这是为了方便用户进行开发和测试、可在单物理机上部署多个节点。而生产环境中，一个物理机上只部署1个节点，此时推荐增加以下配置大小：

* 增加节点的缓存大小，以提高节点的读性能
* 增加节点的SQL内存大小，以提高节点并发处理客户端连接的数量（默认128 MiB大概能够支持6200个客户端连接），同时也能够提高节点在内存中处理ORDER BY、GROUP BY、DISTINCT、JOIN操作和窗口函数的能力。

具体配置方式是在启动节点的时候利用--cache和--max-sql-memory标识：

cockroach start --cache=.25 --max-sql-memory=.25 <other start flags>

文件句柄限制

CockroachDB会使用大量的文件句柄。一般情况下，系统默认的最大文件句柄数量不满足使用需求，因此需要用户手动进行系统配置提高该配置值。 对于每个CockroachDB节点：

* 最大文件句柄数量最小需要设置为1956（其中每个store需要1700个文件句柄，，256用于网络相关），低于该阈值节点将无法启动CockroachDB。
* 推荐将最大文件句柄数量配置为unlimited，或者将该值设置为15000（其中每个store需要10000个文件句柄，5000用于网络相关）乃至更高的值以支持CockroachDB集群增长的性能需求。
* 当最大文件句柄数量不满足CockroachDB的使用需求的时候，在保证256个文件句柄用于网络的情况下，将至多10000个文件句柄用于文件存储，剩下的再分配给网络相关。

增加文件句柄限制

**ATTRIBUTIONS:** 本小节内容衍生自Riak LV 2.1.4 documentation的Open File Limits章节，在Creative Commons Attribution 3.0 Unported License下许可范围内使用。  
**NOTE:** 关于操作系统最大文件句柄值，CockroachDB只取硬限制的值，因此调整软限制的值不是必要的。

假设一个节点配备3个store，我们将最大文件句柄值的硬限制设置为35000（其中每个store需要10000个文件句柄，5000用于网络相关）：

Mac

Yosemite和之后的版本

* 查看文件句柄数量

launchctl limit maxfiles

maxfiles 10240 10240

最后两列分别为软限制和硬限制的值。如果硬限制设置的是未限制，则默认值是10240。

* 创建文件/Library/LaunchDaemons/limit.maxfiles.plist，文件所有者为root:wheel，文件权限至少为-rw-r--r--。添加如下内容：

**<?xml version="1.0" encoding="UTF-8"?>**

**<!DOCTYPE plist PUBLIC "-//Apple//DTD PLIST 1.0//EN" "http://www.apple.com/DTDs/PropertyList-1.0.dtd">**

<plist version="1.0">

<dict>

<key>Label</key>

<string>limit.maxfiles</string>

<key>ProgramArguments</key>

<array>

<string>launchctl</string>

<string>limit</string>

<string>maxfiles</string>

<string>35000</string>

<string>35000</string>

</array>

<key>RunAtLoad</key>

<true/>

<key>ServiceIPC</key>

<false/>

</dict>

</plist>

* 保存文件，重启系统使新配置生效
* 检查配置是否生效

launchctl limit maxfiles

maxfiles 35000 35000

之前的版本

* 查看文件句柄数量

launchctl limit maxfiles

maxfiles 10240 10240

最后两列分别为软限制和硬限制的值。如果硬限制设置的是未限制，则默认值是10240。

* 编辑（或是创建）/etc/launchd.conf，修改或是添加以下设置：

limit maxfiles 35000 35000

* 保存文件，重启系统使新配置生效
* 检查配置是否生效

launchctl limit maxfiles

maxfiles 35000 35000

Linux

每个进程限制

* 确保/etc/pam.d/common-session和/etc/pam.d/common-session-noninteractive均包含以下内容：

session required pam\_limits.so

* 编辑/etc/security/limits.conf，添加以下内容，星号可换成实际运行CockroachDB服务的用户名：

\* soft nofile 35000

\* hard nofile 35000

* 保存文件，重启系统使新配置生效
* 检查配置是否生效

ulimit -a

除了上述配置方式以外，还可以使用[systemd](https://en.wikipedia.org/wiki/Systemd)工具来配置：

* 编辑服务配置，设置最大的文件句柄数量：

[Service]

...

LimitNOFILE=35000

* 利用systemctl命令使配置生效：

systemctl daemon-reload

系统范围限制

需要确保系统范围的最大句柄限制值至少是上述单个进程限制值的10倍

* 查看系统范围内的文件句柄数量：

cat /**proc**/sys/fs/file-max

* 根据需要提高系统范围限制值：

echo 350000 > /**proc**/sys/fs/file-max

编排系统 / Kubernetes

当使用Kubernetes运行CockroachDB，为了更好的性能和系统可靠性，至少需要做出如下的配置：

* 相比于传统的HDDs，推荐使用SSDs。详细查看[SSDs instead of traditional HDDs](http://doc.cockroachchina.baidu.com/#deploy/orchestrated-deployment/kubernetes-performance-optimization/#resource-requests-and-limits)。
* 根据[resource requests and limits](http://doc.cockroachchina.baidu.com/#deploy/orchestrated-deployment/kubernetes-performance-optimization/#disk-type)配置CPU和内存资源

更多配置建议，请查看[CockroachDB Performance on Kubernetes](http://doc.cockroachchina.baidu.com/#deploy/orchestrated-deployment/kubernetes-performance-optimization/)