AWS历次事故分析

& 对我们的启示

汪源

AWS历次事故

- 2011.4.21
  运维误操作和EBS系统故障US East Region的一个AZ中的13%的EBS卷和45%的Single-AZ RDS实例及2.5%Multi-AZ RDS实例，后续进而影响到整个Region的EBS管理操作。事故影响持续3天以上，并最终导致该AZ中0.07%的EBS卷和0.4%的Single-AZ RDS实例无法恢复。

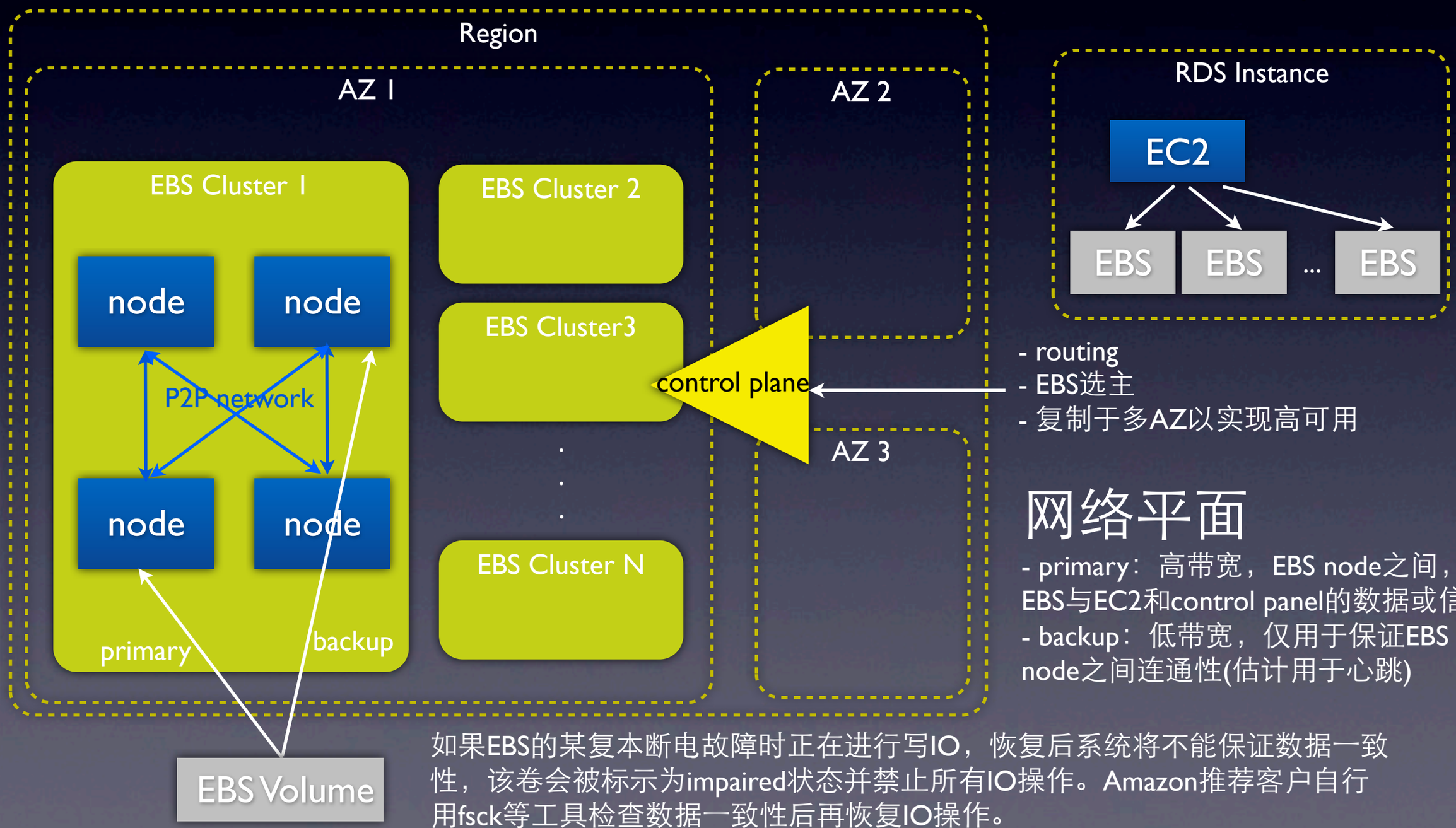
- 2012.6.29
  供电故障影响US East Region中约7%的EC2与EBS实例，电力恢复后的集中恢复过程导致EC2、EBS和ELB的管理服务中断1个多小时。EC2和EBS实例的恢复持续了几小时，部分EBS和Multi-AZ RDS实例不能自行恢复。

- 2012.10.22
  程序bug导致US East Region某AZ EBS re-mirroring风暴，导致该AZ大多数EBS卷不能服务，进而影响该AZ的RDS和ELB服务，同时导致整个Region的EC2/EBS API操作。事务影响持续6个多小时。少量Multi-AZ RDS实例不能自行恢复。

- 2012.12.24
  运维误操作导致US East Region中6.8%的ELB实例无法正常工作近1天，同时导致期间不能进行ELB配置修改等管理操作。

2011.4.21

EBS & RDS架构示意


Region
AZ 1
EBS Cluster 1
node
node
P2P network
node
node
primary
backup
EBS Cluster 2
EBS Cluster3
EBS Cluster N
AZ 2
AZ 3
control plane
RDS Instance
EC2
EBS
EBS
...
EBS
- routing
- EBS选主
- 复制于多AZ以实现高可用
网络平面
- primary：高带宽，EBS node之间，EBS与EC2和control panel的数据或信令
- backup：低带宽，仅用于保证EBS node之间连通性(估计用于心跳)
EBS Volume
如果EBS的某复本断电故障时正在进行写IO，恢复后系统将不能保证数据一致性，该卷会被标示为impaired状态并禁止所有IO操作。Amazon推荐客户自行用fsck等工具检查数据一致性后再恢复IO操作。

事件回放(1)

- 4.21 0:47：网络扩容，一台primary网络的路由器下线，配置失误导致该路由器流量导入backup网络；
- backup网络带宽不足，导致受影响的EBS node跟外界失去联系（大量网络操作超时），从而导致大量EBS卷复制中断；
- 网络误操作及时发现并修复，primary和backup网络恢复；
- 之前复制中断的大量EBS卷开始分配新的backup，导致：1、re-mirroring storm占满网络带宽；2、所在EBS Cluster的空闲空间用完，导致阻塞了CreateVolume请求；3、13%的EBS卷一直在寻找新的backup不能读写；
- 因受影响的EBS Cluster阻塞了CreateVolume请求，EBS control plane的工作线程耗尽，导致整个Region不能进行EBS管理操作；
- 在这种特殊环境下，一个正常很难出现的bug频发，导致EBS node crash，进而导致更多的EBS需要re-mirror；
- 4.21 2:40：通过禁止故障EBS Cluster的CreateVolume操作暂时恢复了EBS control plane；
- 4.21 5:30：re-mirror过程中EC2和EBS node与control plane大量的选主协商请求再次导致control plane过载，整个Region的EBS管理操作再次不可用；

事件回放(2)

- 4.21 8:20：禁止故障EBS Cluster与control plane的一切通信，恢复了control plane；
- 4.21 11:30： hotfix：不过于积极的寻找新的backup，控制了事态不再恶化（在此之前由于之前提过的bug导致EBS node不停crash，时刻有恶化风险）；
- 4.21 12:04：事故控制完成，进入恢复阶段；
- EBS设计为在没有为crash的EBS node上的EBS卷完成重建复本之前不重用该node，为此恢复前必须为故障EBS Cluster增加大量存储容量，大量物理搬迁；
- 由于上述hotfix，新增存储容量不会很快被用起来，小心翼翼的调啊调；
- 4.22 2:00：存储容量加上去了，开始重建复本；
- 4.22 12:30：恢复了近9小时后，还剩2.2%的EBS卷未重建复本。但由于故障EBS Cluster与control plane的通信之前已被中断，已重建的EBS卷还不可用；
- 为了怕把control plane再次搞坏，整晚开发临时方案，第二天上午上线；
- 4.23 18:15：已重建的EBS卷恢复可用；
- 4.24 12:30：某些卷在发生事故时做了快照，用从S3快照恢复的方法恢复了一些卷，还剩1.04%的卷；
- 继续处理剩下的故障卷，最终0.07%的卷没能恢复；

改进措施

- 加强网络维护操作的审核和自动化；
- 预备更多空闲容量以便发生大规模故障时能满足大量re-mirror的需求；
- 复制中断时，更积极的尝试与原复本重连而非重新分配新复本，重新分配时也加入backoff策略；
- 修复那个会导致EBS node crash的bug；
- 改进control plane的处理逻辑防止被故障AZ拖死；
- 下放部分control plane处理逻辑到EBS Cluster；
- 让更多的服务（如VPC）能支持多AZ；
- 举行一系列在线课程，教育用户使用多个AZ来避免SPOF；
- 加强EBS恢复过程的透明性、可控性和自动化水平；
- 加强故障期间的客户沟通；
- 怎么避免最终0.07%的坏盘呢？没说

2012.6.29

事件回放

- 6.29 19:24：雷暴导致US East Region其中一个AZ电力不稳，按设计切换到备用发电机回路，但其中一个数据中心的发电机故障（US East Region包含10多个数据中心，每个AZ有多个数据中心）；
- 6.29 20:04：故障数据中心的UPS电池耗尽，服务器全部宕机。Region内约7%的EC2和EBS实例受影响；
- 6.29 20:24：发电机正常工作，电力恢复；
- 大面积恢复EC2和EBS。EC2恢复时存在boot瓶颈，花了几小时。
- 6.30 0:25：EBS实例都起来了，但故障时有进行中的IO写操作的EBS实例还不能读写，要交给用户判断状态是否一致；
- 6.30 2:45：90%可能不一致的EBS实例交付给用户；
- 6.29 20:04~21:10：EC2和EBS的control plane过载，整个Region内EC2和EBS管理操作不能进行。这同时还因为control plane存储元数据的datastore要手工来切换；
- 大量服务器上线触发了ELB control plane的一个未知bug，使之决定升级大量ELB实例，导致ELB control plane过载。而ELB control plane是用queue来调度任务的，这样用户的ELB操作请求也被长时间阻塞。因为ELB操作不能完成，用户想通过创建新EC2实例并修改ELB配置来切换业务流量的努力也不能成功；
- 由于一个未知bug，部分Multi-AZ RDS实例未能自动failover。

改进措施

- 加强对柴油发电系统的测试
- 改进EC2/EBS的control plane，在datastore故障时能自动切换
- ELB control plane改用多个queue
- 实现不依赖于ELB control plane的DNS负载均衡机制，在某AZ失效时快速撤离到该AZ的流量
- 修复导致部分Multi-AZ RDS实例不能自动failover的bug

2012.10.22

事件回放

- 10.22 10:00：EBS监控agent的bug导致内存泄漏占用了EBS server太多内存，从而无法响应请求，导致US East Region中某AZ内的一小部分EBS卷复制中断（这一bug只在agent上报的服务器故障时才会触发，而之前一台该服务器故障后更新DNS记录有遗漏）；
- 10.22 11:00：re-mirror风暴导致大多数EBS卷都stuck了；
- 10.22 11:10：紧急降低了failover的速度；
- 10.22 11:35：系统开始恢复；
- 10.22 13:40：60%的受影响EBS卷恢复；
- 10.22 15:10：发现问题根源是agent占用内存过高；
- 10.22 16:15：几乎所有EBS卷都恢复了；
- 期间整个Region的EC2与EBS的API服务也受了比较大的影响，原因是对describe类API的throttling策略过于激进；
- 因为两个bug，少量的Multi-AZ RDS实例未能自动failover（其中一个是因为复制已经中断，failover到standby将导致丢事务）；
- 因为ELB服务用EBS存储配置信息，故障AZ内的部分Single-AZ ELB实例也中断服务。因为ELB使用EIP，恢复过程中耗尽了EIP资源，导致ELB的恢复持续到21:50。Multi-AZ ELB实例本来应该将流量从故障AZ切走，但因为一个软件bug，一小部分Multi-AZ ELB实例未能及时切走到故障AZ的流量。

改进措施

- 完善监控，后续再发生类似的内存泄漏时报警
- 完善DNS更新流程，同时部署监控在DNS更新未能及时完成时报警
- 调整EBS failover策略，防止过快failover导致系统快速恶化
- 不使用对describe类操作过于激进的throttling策略
- 改善监控，监控用户级的throttling行为
- 保证有充足的EIP资源用于ELB的恢复
- 解除ELB与EBS的耦合，使得同AZ的EBS故障时，ELB也可恢复
- 修复agent程序内存泄漏bug
- 修复导致Multi-AZ RDS实例不能自动failover的bug
- 修复Multi-AZ ELB实例未能将到故障AZ的流量切走的bug

2012.12.24

事件回放

- 12.24 12:24：部分ELB元数据被开发人员不小心删除（已有ELB实例不做修改时不受影响，一修改就会不能正常工作或性能下降；创建新的ELB实例不受影响）；
- 几小时后，发现问题的根源；
- 12.24 17:02：为防止更多的ELB实例受影响，禁止了修改ELB配置的一些管理操作。此时6.8%的ELB实例受影响；
- 尝试恢复被删除的ELB元数据（应该是基于日志的PIT恢复），但第一个方法工作了几个小时后没能恢复数据；
- 12.25 5:40：ELB元数据恢复成功，开始修正之前无操作的ELB实例并逐步开放ELB管理操作；
- 12.25 12:05：系统完全恢复。

改进措施

- 加强对ELB配置数据访问权限的控制，不对开发人员开放长期权限；
- 改进恢复算法，更快的恢复丢失的ELB配置数据；
- 改进ELB control plane算法，使之在故障后能自行恢复；

启示(I)

- 研发高可伸缩、高可靠、高可用的系统非常困难，在系统与用户规模增加之后，看似安全的地方往往最危险。因此，我们在开发此类系统时，要谨记：所有可能出问题的地方都一定会出问题
- 稳定可靠是系统的生命。1次严重故障就可能对我们系统的推广应用带来灾难性的后果
- 计划采取以下三方面措施
  - 在需求、设计、开发、测试中强化对系统质量的关注
  - 强化试运营机制，计划利用博客搬迁替换服务器新开辟用于开发测试的试运营环境
  - 强化运维制度，减少人为误操作

启示(2)

- 需求
  - 给予核心但偶现bug的处理、系统质量测试验证任务更高优先级
  - 功能稳定后各sprint持续安排提高测试覆盖率的任务
- 设计
  - 强制总体设计评审会机制，强制要求各项目持续维护与实现一致的总体设计文档（团队、主管、资深架构师等审阅此文档可能发现很多问题的改进）
  - 强制进行并发正确性、故障处理流程正确性、性能（包括大面积灾难恢复时的性能）、资源限制throttling等项目的论证
  - 预留带宽给控制通道：每台服务器上的数据通道占用的带宽要实施总体控制
  - 复制机制设计应积极重连，谨慎重建，设计使得重连后的恢复远快于重建
  - 采取机制避免操作风暴，建议通过集中控制与调度、请求退避、限流、工作流独立、多进程等机制来预防操作风暴的产生

启示(3)

- 开发
  – 强制实施code review机制
- 测试
  – 加强整体测试充分度，对测试覆盖率实施更高的要求（整体80-90%，核心模块90-95%）。测试覆盖率与功能发布同步
  – 强制实施单元测试机制
  – 强制实施联调环境与线上环境自动回归测试机制
  – 加强大面积故障恢复测试与论证
  – 加强系统各组件的压力测试，防止高并发时偶现的bug