面向云存储安全的可自愈拜占庭 Quorum 系统

王亚文,郭云飞,刘文彦,霍树民,杨 超
(国家数字交换系统工程技术研究中心,河南郑州 450002)

摘 要: 针对云存储系统中的数据安全问题,提出了一种面向云存储安全的可自愈拜占庭 Quorum 系统. 该系统以虚拟机作为后端存储设备构建虚拟存储节点,利用虚拟机多样化操作系统,以及动态迁移、快速部署等机制构建动态异构的存储系统架构. 在拜占庭容错门限的基础上,提出自愈门限的概念,并设计相应系统安全协议,实现存储节点的自动化异常检测和状态复原. 实验结果表明,提出的云存储系统具有较高的鲁棒性,能有效提高存储数据的安全性.

关键词: 云存储; 数据安全; 拜占庭 Quorum 系统; 自愈机制
中图分类号: TP309.2 **文献标识码:** A **文章编号:** 0372-2112 (2020)04-0675-07
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2020.04.009

A Self-healing Byzantine Quorum System for Cloud Storage Security

WANG Ya-wen, GUO Yun-fei, LIU Wen-yan, HUO Shu-min, YANG Chao
(*National Digital Switching System Engineering R&D Center, Zhengzhou, Henan* 450002, *China*)

Abstract: For the problem of data security in the cloud storage system, a self-healing Byzantine Quorum system for the cloud storage security is proposed. In this system, the virtual machines are used as back-end storage devices to construct the virtual storage node. The diverse operating systems, dynamic migration and rapid generation mechanism of virtual machines are introduced to build dynamic and heterogeneous storage system architecture. On the basis of the Byzantine fault tolerance threshold, the concept of self-healing threshold is presented and several security protocols are devised to achieve automated anomaly detection and storage node recovery. The experimental results show that the proposed cloud storage system is greatly robust and can effectively improve the security of stored data.

Key words: cloud storage; data security; Byzantine Quorum system; self-healing mechanism

1 引言

随着计算机和网络技术的飞速发展,人类进入了数据大爆炸时代,我们的生活充斥着各种各样的数据. 为了方便人们快速有效的存储和获取数据,存储即服务的商业理念被提出[1,2],在此理念下,各大公司纷纷建立了公有云存储系统,如 Amazon 公司的 S3, Microsoft 的 Azure 等,面向公众提供服务. 除此之外,学校,银行,军事机构等也纷纷建立了自己的私有云存储系统来管理工作数据.

数据安全研究主要分为数据可用性研究、数据完整性研究以及数据机密性研究[3],本文重点研究云存储系统中的数据可用性问题. 数据可用性问题来源于云存储系统的不可控性,因此研究者们往往从系统框架层面进行改进来提高数据的可用性,其主流思想是构建冗余系统架构,实现多节点数据备份,避免单点故障[4]. 典型的系统有 HDFS、Ceph 以及 Swift 等. HDFS (Hadoop Distribute File System)[5]将数据分割成多个数据块,并冗余备份在多个数据节点上. 但是 HDFS 采用管道(pipeline)的形式将数据依次存入数据节点. 假设系统采用 n 重数据备份,那么只有当第 n 个节点数据写入成功后,此次数据存储操作才算结束. 这种流水线式的存储方式会带来较大的时间开销. Ceph[6]采用分布式主从结构,需要至少一个 Ceph Monitor 和多个 Ceph OSD 组成存储集群. Ceph OSD 主要负责数据存储,一份数据需要保存在至少两个 Ceph OSD 中以保障数据的可用性. Ceph Monitor 作为 Ceph 的监控器,其功能是维护整个集群的健康状态并提供一致性决策. 但是在分

收稿日期:2019-04-04;修回日期:2019-07-27;责任编辑:李勇锋
基金项目:国家重点研发计划(No. 2018YFB0804004);国家自然科学基金创新群体(No. 61521003)

布式主从结构中,主节点异常带来的不良影响一定高于从节点,各节点地位的不平等性会降低系统整体安全性能. 针对此问题,Quorum 机制[7]被提出. 在该机制中,各节点的地位是平等的,通过对读写集合的交汇性进行约束,保证各节点存储数据一致性的同时,也提高了数据读写效率. 凭借这些优点,Quorum 机制常被用于设计分布式存储系统,其中 OpenStack Swfit[8]就是典型代表. 但是 Quorum 机制削弱了数据的一致性,当服务器故障造成 Quorum 无法交汇时,会出现用户无法获取最新存储数据的情况. 针对此问题,Malkhi[9]将拜占庭容错与 Quorum 相结合提出了拜占庭 Quorum 机制,对 Quorum 的交汇性作出更多的要求[10]. 文献[11]利用 4 个来自不同提供商的云系统并结合拜占庭 Quorum 机制设计了名为 DepSky 的容错云存储系统,能有效提高存储数据的可用性.

容错机制的目的是实现系统的高可用,但是目前云存储系统采用的容错机制无法实现系统长期的高可用[12],因为在缺少人为干预条件下,容错系统中的正常节点数量的减少是一个不可逆的过程,这就使得容错系统出现故障仅仅是时间长短的问题. 为了在分布式存储系统中实现存储节点的自愈,本文提出一种面向云存储安全的可自愈拜占庭 Quorum 系统,以虚拟机作为云存储系统后端存储设备,利用虚拟机多样化操作系统、虚拟机动态迁移以及快速部署等方法构建动态异构的分布式存储系统来提高系统的弹性. 并且在拜占庭 Quorum 机制的基础上提出一种系统自愈机制,在系统异常节点数超出容错门限时触发,实现自动化的存储节点状态复原.

2 可自愈拜占庭 Quorum 存储系统

本文在 OpenStack Swift 开源软件的基础上结合拜占庭 Quorum 机制实现了节点自愈的云存储系统. 本节将从系统架构、系统自愈门限、系统安全协议 3 个方面进行介绍.

2.1 动态分布式存储架构

在分布式存储系统中,物理服务器通常作为后端存储设备,但这种方式存在以下几方面问题:

存储节点数量固定. 在 Swift 软件中,最小的一个存储节点可以是一个磁盘分区. 通常情况下一个安装好操作系统的物理服务器的分区数是一定的,这也表明在一个物理服务器中能够部署的最大存储节点数是固定的.

存储节点状态绑定. 在一个物理服务器上部署的所有存储节点的工作状态是相互绑定的,它们或是全部在线或是全部离线.

存储节点位置固定. 当一个存储节点部署在某一个物理服务器上时,它的位置将始终固定在此物理服务器中,无法变更.

系统复原能力差. 当一个存储节点出现故障时,在不添加新的物理服务器的基础上,难以快速恢复故障节点.

针对这些问题,提出一种动态异构分布式存储架构,利用虚拟机作为后端存储设备,系统架构如图 1 所示.

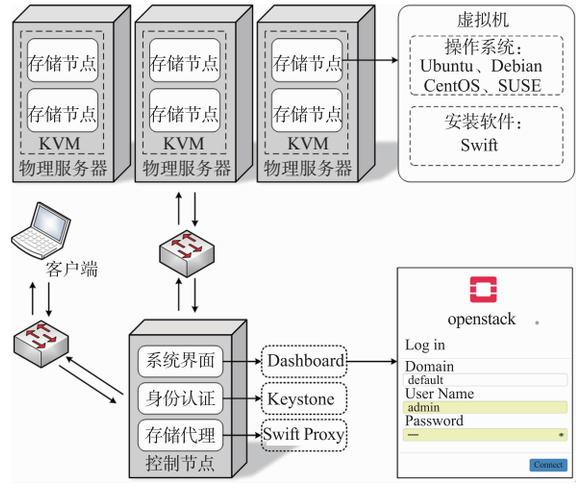


图1 动态分布式存储架构

在该架构中,将 Swift 服务部署在 KVM 虚拟机中,将虚拟机作为存储节点,我们将其称为虚拟存储节点,虚拟机操作系统可以是 CentOS, Ubuntu, SUSE, Debian 系统(目前 OpenStack Swift 只支持在这几种系统中安装). 用户登录 OpenStack Dashboard 界面进行数据存取操作,控制节点的 Swift Proxy 将用户请求转发给虚拟存储节点进行响应. 虚拟机与宿主机通过网桥相连,保证 Swift Proxy、虚拟存储节点、虚拟存储节点宿主机在同一网络下.

该架构继承了虚拟机动态灵活的特点,首先在一个物理服务器上可以实时部署任意数量的节点. 并且可以选择性的指定哪些虚拟存储节点在线,哪些虚拟存储节点离线. 比如当检测出某个虚拟存储节点异常时,能够快速的将该节点下线而不影响同一物理服务器上的其他存储节点. 同时,该架构还可与入侵检测系统相结合,当部署在物理服务器上的入侵检测系统出现入侵报警,可以将宿主机上的所有虚拟存储节点进行迁移,远离攻击者. 此外,虚拟机快速部署和销毁的特点也赋予了系统较强的复原能力.

2.2 系统自愈门限

当存储节点被攻击者挟持后大体会呈现两种表现,拒绝响应请求和错误响应请求. 采用哈希校验的方式可以有效应对错误响应问题,因此我们重点讨论拒

绝响应问题. 针对拒绝响应问题,本文在系统正常工作状态和系统瘫痪状态之间增加了自愈状态,当系统错误节点数超过了容错度 f_b 后,系统无法正常工作,但此时系统可以开始进行自愈,即复原异常节点. 如果系统持续受到攻击,当异常节点数超过自愈门限 f_h 时,系统就失去了自愈能力,陷入永久瘫痪状态. 我们下面讨论自愈门限 f_h 如何确定.

假设系统的拜占庭错误容忍度为 f_b,则系统存储节点数 N 需满足式(1),

$$N \geq 3f_b + 1 \tag{1}$$

读写 Quorum 的大小 Q 如式(2)所示,

$$Q = N - f_b \tag{2}$$

在写操作过后,会有至少 Q 个节点写上此次数据 d_i,反之,有最多 $N-Q$ 个节点没有写入数据 d_i. 因此只要系统中存在至少 $N-Q+1$ 个正常节点,数据 d_i 就一定会存在于系统之中. 因此我们提出自愈门限 f_h,其定义如式(3)所示,

$$f_h = N - (N - Q + 1) = Q - 1 \tag{3}$$

只要系统中异常节点数不超过自愈门限 f_h,系统中就一定包含完整的存储数据,这是系统能够实现自愈的必要条件. 结合式(1)、(2)和(3)可得到自愈门限 f_h 和拜占庭 Quorum 容错度 f_b 之间的关系,如式(4)所示,

$$f_h \geq 2f_b \tag{4}$$

式(4)说明自愈门限比拜占庭 Quorum 容错度要宽泛很多,从而可以进一步提高系统在受到网络攻击时的存活能力.

2.3　系统安全协议

本文设计的系统包含三种安全协议:(1)系统自愈协议;(2)错误累积替换协议;(3)存储节点转移协议.

2.3.1　系统自愈协议

当系统中拒绝响应请求的节点数超过了容错度 f_b 后,系统将无法正常工作. 但是只要异常节点数小于自愈门限 f_h,系统中就一定存在完整的存储数据,通过存储节点同步机制,即可确保每个节点都包含了完整的存储数据. 由于存储节点以虚拟机的形式存在,因此可以通过镜像复制的方法快速产生包含完整数据的全新节点实现系统自愈.

首先我们对自愈协议触发场景进行分析. 在拜占庭 Quorum 存储系统中,当用户发起数据上传请求时,系统只有将数据成功写入至少 Q 个节点中才算完成响应. 如果系统中实际异常节点数 f_p 超过 f_b,即 $f_p > f_b$,则系统实际能写入的节点数 Q_p 必小于 Q,如式(5)所示,

$$Q_p = N - f_p > N - f_b = Q \tag{5}$$

此时,系统无法顺利完成针对数据上传请求的响应操作,自愈协议就是在此情况下触发,其协议内容如算法 1 所示.

算法 1　自愈协议

```
Swift Proxy 向所有节点发送查询请求〈request,n,o_q〉;//n 表示请求序号,o_q 表示查询各节点元数据的操作
第 i 个节点收到请求后向 Swift Proxy 回复消息〈reply,n,t_i[ ]〉;//t_i[j] 表示第 i 个节点上存储的第 j 条数据的时间戳信息
N = Swift Proxy 在规定的时间内收到的回复消息数量;
l = 0;
/* 存储节点同步机制 */
for k = 1:N
   for g = 1:Length(t_k[ ])
     if t_k[g] ⊄ t_p[ ]
       l = l + 1;
       t_p[l] = t_k[g];
     else
       continue;
     end
   end
end
for k = 1:N
   Swift Proxy 将 t_k[ ] 与 t_p[ ] 进行比较;
   Swift Proxy 负责转发节点 k 缺少的数据;
end
/* * * * * * * * * * * * * * * * * * * * * * */
Swift Proxy 向任意节点的宿主机发送自愈请求〈request,c,o_c〉;//c 表示存储节点配置信息,o_c 表示复制存储节点镜像的操作
如果 Swift Proxy 依然无法执行数据上传操作,重新执行自愈协议;
```

2.3.2　错误累积替换协议

Swift 软件自带故障处理机制来保证数据的一致性,其内设的 Auditor 会周期性的扫描存储文件的哈希值以验证数据是否完整,当检测出异常时,Replicator 会被激活进行相应的复制替换操作. 该机制能够一定程度上保证数据的安全性,但是当某一节点被攻击者挟持后,仅仅保证数据安全性是不够的,我们还应将被挟持节点替换掉. 因此我们在 Swift 自带故障处理机制基础上进行改进,设计了错误累积替换协议,其协议内容如算法 2 所示.

算法 2　错误累积替换协议

```
输入:N,Q,E(错误累积替换阈值)
/* 数据上传 */
h_u = Hash(d_u);//d_u 表示待上传数据
for i = 1:N
   return_value = Swift Proxy 向节点 i 发送写操作请求〈request,n,o_w,d_u,h_u〉;// n 表示请求序号,o_w 表示写操作
   if return_value! = SUCCESS
       C[i] + +;
     end
end
/* 数据下载 */
```

return_value[2]初始化;// *return_value*[0]存放数据,*return_value*[1]存放哈希值
for $i=1:N$
  return_value = Swift Proxy 向节点 i 发送读操作请求〈*request*, n, o_r, u〉;//n 表示请求序号,o_r 表示读操作,u 表示待下载数据的 url 信息
  if *return_value*[1] ! = Hash(*return_value*[0])
    $C[i]++$;
  end
end
/* Swift 自带故障处理机制 */
while *signal* ! = exit
  for $i=1:N$
    return_value = Auditor_scan(i);//Swift 自带后台进程 Auditor 周期性扫描节点 i
    if *return_value* ! = SUCCESS //检测到节点 i 中数据存在异常
      $C[i]++$;
    end
  end
end
/* */
for $i=1:N$
  if $C[i]>E$
    销毁节点 i;
    启动存储节点同步机制;//见自愈协议
    复制存储节点{$j|C[j]==\min(C[1],C[2],\cdots,C[N])$}的镜像并部署全新节点;
  end
end

2.3.3 存储节点转移协议

我们在各虚拟存储节点的宿主机上部署了 Snort 入侵检测器,其利用内部规则对网络数据包进行分析和预警. 当某宿主机检测出恶意数据包时,启动存储节点转移协议,将该宿主机上的所有虚拟存储节点进行迁移,协议内容如算法 3 所示.

算法 3 存储节点转移协议

输入:*snort_log*(Snort 报警日志),N
/* 虚拟存储节点宿主机端 */
while *signal* ! = exit
  return_value = Scan(*snort_log*);
  if *return_value* = = ALERT
    向 Swift Proxy 发送迁移请求〈*request*, n_m, *id*, *vm*[ ]〉;//n_m 表示请求序号,*id* 表示宿主机编号,*vm*[ ]表示需要迁移的虚拟机信息
  end
end
/* Swift Proxy 端 */
if 接收到迁移请求
  for $i=1:N$
    resource[i] = 向节点 i 发送查询请求〈*request*, n_q, o_q〉;//n_p 表示请求序号,o_q 表示查询剩余资源操作
  end
  strategy = 根据 *resource*[ ]和 *vm*[ ]的信息制定迁移策略;
  向对应的宿主机回复消息〈*reply*, n_m, *strategy*, o_m〉;//o_m 表示迁移操作
end

3 实验

本节对三种系统安全协议进行功能测试.

3.1 测试环境

安全协议功能测试环境如图 2 所示. 我们利用四台服务器搭建了安全协议功能测试环境,其中一台作为控制节点,部署 Swift Proxy, Dashboard 和 Keystone 等 OpenStack 开源组件,另外三台作为存储节点宿主机,部署 Virt-manager 和 KVM 等软件. 在宿主机 1 和 2 上,我们分别部署若干台虚拟机并安装 Swift 软件提供存储服务,存储节点网卡和宿主机网卡通过网桥相连. 宿主机 3 用来测试存储节点转移协议,初始阶段不部署任何虚拟机. 三个宿主机与控制节点通过管理网络相连,控制节点通过内网穿透工具使客户端能够通过公网直接访问 Dashboard 页面,并进行数据上传和下载操作. 为了模拟系统受攻击的情景,假设攻击者存在于管理网络之中,并可通过 SSH(Secure Shell)接入到宿主机中.

3.2 自愈协议功能测试

在本节中,现有 4 个虚拟存储节点,平均部署在宿主机 1 和 2 上,Quorum 大小为 3. 攻击者通过 SSH 渗透到宿主机 1 中,并强制关闭两个存储节点,然后系统将进入自愈状态. 系统自愈主要分为以下几个部分,节点同步、镜像拷贝、镜像实例化以及代理节点重新生成 ring 文件并分发到各存储节点中. 其中主要的时间花费在镜像拷贝上,因此异常节点的恢复时间与其镜像的大小紧密相关. 我们测试了在不同镜像大小条件下的异常节点恢复时间,其结果如图 3 所示. 从图中可以看出,系统自愈所需时间与虚拟存储节点占用的存储空间成线性增长的关系.

之后,我们测试在不同攻击周期下的系统状态,并利用传统拜占庭 Quorum 容错机制进行对比. 假设每个虚拟存储节点占据 400G 磁盘空间. 根据图 3 结果可知系统自愈大约需要 140 分钟. 我们分别设定攻击周期为 100 和 200 分钟,并统计两种机制的正常节点数量随时间变化情况,其结果如图 4 所示. 从图 4 中可以看出,当攻击者攻击周期较短,小于系统自愈周期时,本文系统的自愈功能难以发挥作用. 但当攻击者周期大于系统自愈周期时,系统会在自愈状态和正常状态下持续转换.

3.3 错误累积替换协议功能测试

我们首先测试在系统处于正常工作状态时,系统中存在的拒绝响应异常节点平均需要多长时间能够被检测出. 本文在 2.3.2 节错误累积替换协议中提到,只

有当用户进行数据上传时,才有机会检测出拒绝响应异常节点,因此异常检测时间应该与用户执行数据上传的周期有关. 我们测试在不同数据上传周期下,系统检测出异常节点所需平均时间,其结果如图 5 所示. 可以看出,系统检测出异常节点所需平均时间与数据上传周期基本成线性增长关系.

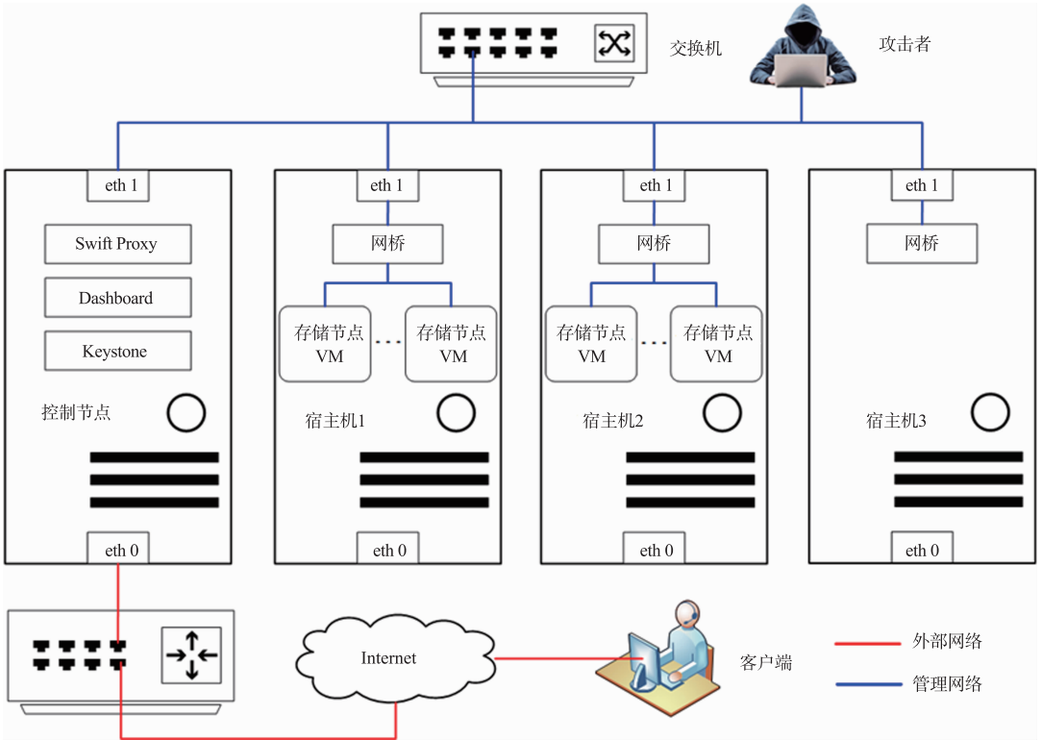


图2　安全协议功能测试环境

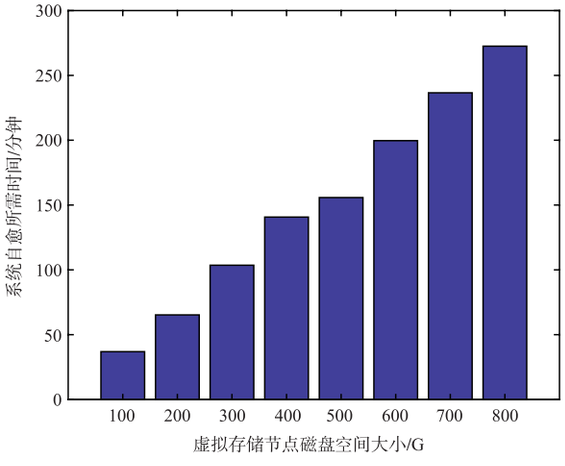


图3　系统自愈所需时间

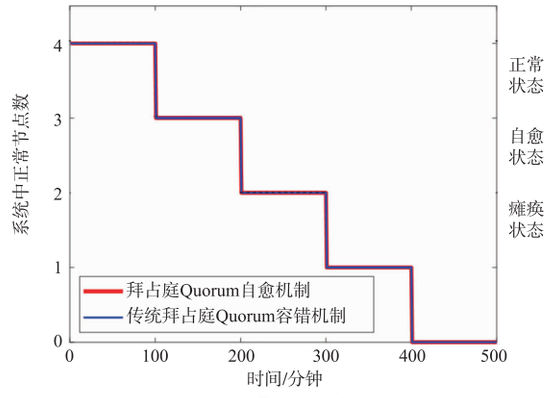
同3.2 节类似,我们测试在不同攻击周期下的系统状态,并利用传统拜占庭 Quorum 容错机制进行对比. 依然假设每个虚拟存储节点占据 400G 磁盘空间且用户平均上传数据周期为 40 分钟,则根据图 5 结果可知系统需大约240 分钟检测出异常节点. 异常节点替换所需时间与节点自愈时间相同,根据图 3 结果可知约 140 分钟. 我们设定攻击周期为 500 分钟,并统计两种机制的正常节点数量随时间变化情况,其结果如图 6 所示. 从图中可以看出,只要攻击周期大于异常节点检测和异常节点替换周期之和,即使系统受到持续攻击,也能保持正常的工作状态.

3.4　存储节点转移协议功能测试

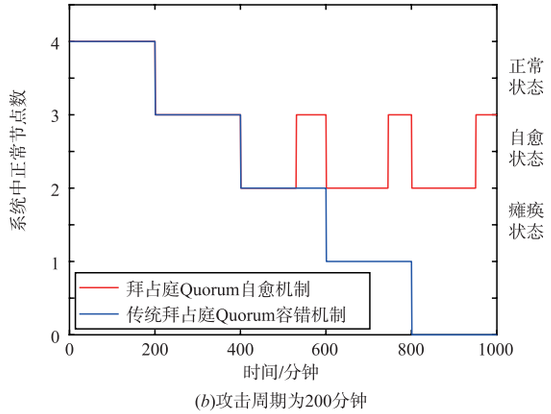
为进行存储节点转移协议的功能测试,我们在宿主机 1 上部署两个虚拟存储节点,宿主机 3 不部署任何虚拟存储节点. 为了方便展示系统功能,我们登录宿主机 1,并打开 Virt-manager 和 Snort 入侵检测器界面,其中 Virt-manager 远程连接宿主机 3 以查看虚拟存储节点是否迁移成功,Snort 入侵检测器加载一条检测 SSH 登录的规则并对网桥流量进行监控. 攻击前的 Virt-manager 和 Snort 运行情况如图 7(*a*)所示. 当攻击者通过 SSH 入侵到宿主机 1 后,Snort 会产生报警,触发存储节点转移协议,经过一系列热迁移操作后,从图 7(*b*)中看到虚拟存储节点已经成功的转移到宿主机 3 上.

4　总结与展望

现有的云存储系统缺少针对存储节点的自我复原机制,难以抵抗持续的网络攻击,因此提出面向云存储



(*a*)攻击周期为100分钟



(*b*)攻击周期为200分钟

图4 不同攻击周期下的自愈协议功能测试结果

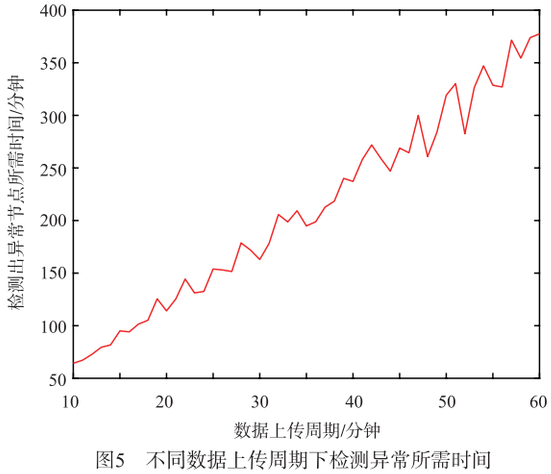


图5 不同数据上传周期下检测异常所需时间

安全的可自愈拜占庭 Quorum 系统. 该系统首先以虚拟机作为云存储系统后端存储设备,利用虚拟机多样化操作系统,虚拟机动态迁移、动态部署等方法构建动态异构的云存储架构. 然后在拜占庭 Quorum 容错门限的基础上,提出了自愈门限的概念,并根据系统异常节点数对系统状态进行了划分,即正常状态、自愈状态和瘫痪状态. 最后设计了三种安全协议,根据当前系统所处

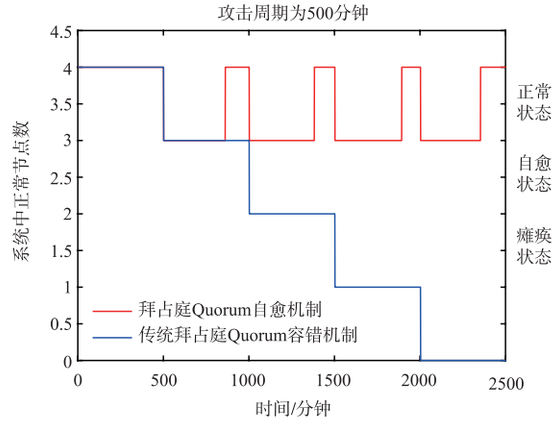
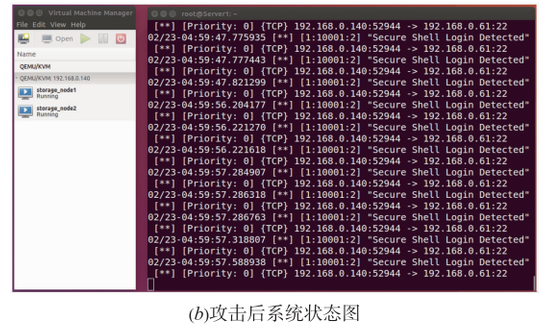


图6 特定攻击周期下的错误累积替换协议功能测试结果

(*a*)攻击前系统状态图



(*b*)攻击后系统状态图

图7 存储节点转移协议功能测试结果

状态,自适应的采取相应的安全机制进行存储节点状态复原. 实验结果表明,本文系统能够有效提高云存储系统抵御网络攻击的能力.

参考文献

[1] 周恩光,李舟军,郭华,等. 一个改进的云存储数据完整性验证方案[J]. 电子学报,2014,42(1):150 – 154.
Zhou En-guang, Li Zhou-jun, Guo Hua, et al. An improved data integrity verification scheme in cloud storage system [J]. Acta Electronica Sinica, 2014, 42(1): 150 – 154. (in Chinese)

[2] 周洪丞,杨超,马建峰,等. 云端数据异地容灾验证方案研究[J]. 电子学报,2016,44(10):2485 – 2494.

Zhou Hong-cheng, Yang Chao, Ma Jian-feng, et al. A study off-site disaster recovery performance of cloud data[J]. Acta Electronica Sinica, 2016, 44(10): 2485 – 2494. (in Chinese)

[3] Xiao Z, Xiao Y. Security and privacy in cloud computing [J]. IEEE Communications Surveys & Tutorials, 2013, 15(2): 843 – 859.

[4] Liu J, Shen H. A low-cost multi-failure resilient replication scheme for high data availability in cloud storage[A]. IEEE 23rd International Conference on High Performance Computing, Data and Analytics[C]. Hyderabad, India: IEEE Press, 2016. 242 – 251.

[5] Karun A K, Chitharanjan K. A review on hadoop—HDFS infrastructure extensions[A]. IEEE Conference on Information & Communication Technologies (ICT)[C]. Tamil Nadu, India: IEEE Press, 2013. 132 – 137.

[6] Zhang J, Wu Y, Chung Y C. PROAR: A weak consistency model for Ceph[A]. IEEE 22nd International Conference on Parallel and Distributed Systems (ICPADS)[C]. Wuhan, China: IEEE Press, 2016. 347 – 353.

[7] Lea T E, Jehl L, Meling H. Towards new abstractions for implementing quorum-based systems[A]. IEEE 37th International Conference on Distributed Computing Systems (ICDCS)[C]. Atlanta, USA: IEEE Press, 2017. 2380 – 2385.

[8] Chekam T T, Zhai E, Li Z, et al. On the synchronization bottleneck of OpenStack swift-like cloud storage systems [A]. IEEE 35th International Conference on INFOCOM [C]. San Francisco, USA: IEEE Press, 2016. 1 – 9.

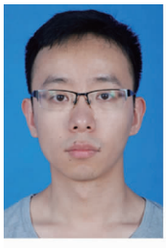
[9] Malkhi D, Reiter M. Byzantine quorum systems[J]. Distributed computing, 1998, 11(4): 203 – 213.

[10] 范捷,易乐天,舒继武. 拜占庭系统技术研究综述[J]. 软件学报,2013,6:1346 – 1360.
Fan Jie, Yi Le-tian, Shu Ji-wu. Research on the technologies of Byzantine system[J]. Journal of Software, 2013, 6: 1346 – 1360. (in Chinese)

[11] Bessani A, Correia M, Quaresma B, et al. DepSky: dependable and secure storage in a cloud-of-clouds[J]. ACM Transactions on Storage (TOS), 2013, 9(4): 31 – 45.

[12] Psaier H, Dustdar S. A survey on self-healing systems: approaches and systems[J]. Computing, 2011, 91(1): 43 – 73.

作者简介



王亚文　男,1990 年 8 月出生,河南郑州人. 国家数字交换系统工程技术研究中心博士研究生,主要研究方向为云安全.
E-mail:15738321455@163.com

郭云飞　男,1963 年出生,河南郑州人. 国家数字交换系统工程技术研究中心教授,博士生导师,主要研究方向为电信网安全.