ZStack 技术白皮书精选

ZStack--标签系统

扫一扫二维码，获取更多技术干货吧

版权声明

摘要

大道至简·极速部署，ZStack 致力于产品化私有云和混合云。

ZStack 是新一代创新开源的云计算 IaaS 软件，由英特尔、微软、CloudStack 等世界上最早一批虚拟化工程师创建，拥有 KVM、Xen、Hyper-V 等成熟的技术背景。

ZStack 创新提出了云计算 4S 理念，即 Simple （简单）、Strong （健壮）、Smart （智能）、Scalable （弹性），通过全异步架构，无状态服务架构，无锁架构等核心技术，完美解决云计算执行效率低，系统不稳定，不能支撑高并发等问题，实现 HA 和轻量化管理。

ZStack 发起并维护着国内最大的自主开源 IaaS 社区——zstack.io，吸引了 6000 多名社区用户，对外公开的 API 超过 1000 个。基于这 1000 多个 API，用户可以自由组装出自己的私有云、混合云，甚至利用 ZStack 搭建公有云对外提供服务。

ZStack 拥有充足的知识产权储备，积极申报多项软著和专利，参与业内标准、白皮书的撰写，入选云计算行业方案目录，还通过了工信部云服务能力认证和信通院可信云认证。ZStack 面向企业用户提供基于 IaaS 的私有云和混合云，是业内唯一一家实现产品化，并领先业内首家推出同时打通数据面和控制面无缝混合云的云服务商。选择 ZStack，用户可以官网直接下载、1 台 PC 也可上云、30 分钟完成从裸机的安装部署。

目前已有 1000 多家企业用户选择了 ZStack 云平台。

ZSTACK--标签系统

ZStack 中的标签不仅帮助用户聚集资源，也帮助控制软件行为。ZStack 有一套完整的规范，用以定义标签的类别、形式和用法。除了用户外，插件也可以创建自己的标签，以记录元数据和拓展现有的资源属性；通过这些手段，标签可以帮助插件引入新的特性，而不改变 ZStack 的数据库结构，消除了在软件升级对数据库迁移的需求。

动机

随着云中资源的不断增长，用户可能会想要有一种方式，使用人类可读的标签，去分组相似的资源。举个例子，所有 Web 服务器的虚拟机都可以有一个标签*'web-tier-vm'*，这样可以从 UI 和 CLI 把它们作为一个组来获取。对于 IaaS 本身，预先定义的业务逻辑也许从来都不能满足用户的需求。以创建虚拟机为例，默认的选择目标主机的算法是，从主机池中随机选择一个，但用户可能需要各种各样的算法来满足它们的使用情景。比如说选择内存超过 8G 的主机，选择拥有 SR-IOV 硬件的主机，或选择一个有当前用户的运行中虚拟机的主机。IaaS 软件几乎不能为所有无止境的、不可预知的需求提供单独的 API，必须有一种机制允许基础 API（如 APICreateVmInstanceMsg）携带额外信息。

根据各自的业务逻辑，插件可以选择是否创建数据库表。比如，Open vSwitch L2 Network 插件，由于需要创建一种新的类型的资源，可能需要添加一张新表；然而，一个允许主机保留内存的插件可能不需要添加一张新表，而仅需在主机上附加一点数据。如果 IaaS 软件没有为插件提供一种附加数据，它们将开始创造新的、琐碎的模式或添加现有模式的列从而修改现有的模式，导致软件升级时数据库迁移的难处理的情况。

最后，对于建立在 ZStack 上的第三方软件，允许它们将信息存储到 ZStack 的数据库可以避免数据完整性问题，并使得它们可以使用 ZStack 的全部查询 API（详见“查询 API”）。

问题

大多数 IaaS 软件都有着标签的概念。然而，它们并不是都为不同场景定义了一个详尽

的标签规范。例如，一些 IaaS 使用标签是为了用户聚合资源，一些 IaaS 是为了内部业务逻辑。ZStack 则为不同场景的标签的每一个层面都精心设计了标签规范。

标签系统

在 ZStack 中，标签本质上是携带了少量资源相关信息的字符串。一个标签通常由以下几个字段组成：

FIELD	DESCRIPTION
uuid	标签的 UUID
resourceUuid	标签所关联的资源的 UUID
resourceType	标签所关联的资源的类型
Tag	一个包含了有意义信息的字符串
Type	标签类型：System 或者 User

在标签方面，ZStack 和其他 IaaS 软件的本质区别是 ZStack 将标签分为两类：用户（User）和系统（System）。

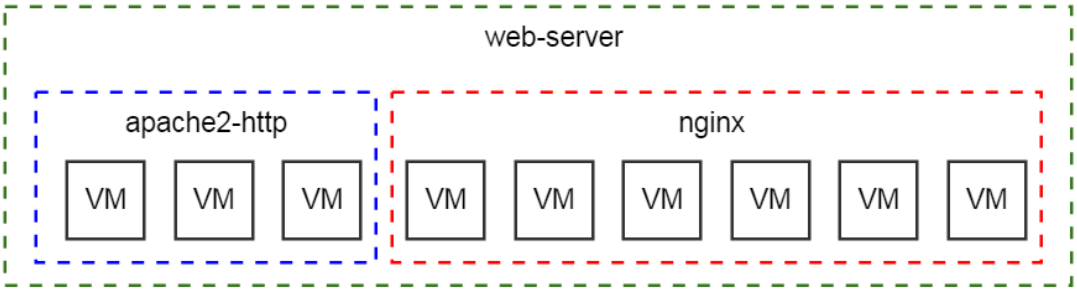
1.用户标签

用户标签，顾名思义，是用户为资源分组而创建的标签。例如，通过标签'apache2-http'将安装了 Apache2 HTTP 服务器的虚拟机分组，这样用户就可以通过查询 API 获取这些虚拟机，使用标签'apache2-http'作为查询条件即可：

```
QueryVmInstance __userTag__=apache2-http
```

备注：详见“查询 API”

这是最常见的标签使用方法，一个资源可以和多个标签相关联，并且被不同的逻辑组划分。



用户标签也可以通过和系统的标签一起使用来控制 ZStack 的行为。例如，如果一个用户标签 SSD 已经在主存储系统上创建好，那么一个系统标签可以指导 ZStack 在有用户标签 SSD 的主存储上去创建 VM 的根目录。在这种情况下，用户标签更像是用户输入的资源元数据。我们很快就会看到，插件也可以使用系统标签创建资源元数据。

2.系统标签

不像用户标签可以被用户在任意时间、以任意值创建，系统标签有固定的格式，并且是被 ZStack 的业务服务和插件提前定义好的，可以在以下场景被使用：

2.1 元数据

插件可以使用系统标签来记录资源的元数据。例如，主机的数据库表中没有列去记录如 hypervisor 版本，hypervisor SDK 版本这样的元数据；然而，衍生的主机插件，例如，KVM 主机插件，可能需要这些元数据来确定当前虚拟机管理程序是否有某些特征；例如，是否能对 KVM 在线快照是由 libvirt 和 QEMU 版本决定的。在 ZStack 中，当连接到后端主机时，KVM 主机插件将 OS 的版本，libvirt 版本，QEMU 的版本和 qemu-img 工具的版本作为系统标签保存。

```
QuerySystemTag fields=tag resourceUuid=d07066c4de02404a948772e131139eb4
```


```
{
  "inventories": [
      {
          "tag": "capability:liveSnapshot"
      },
      {
          "tag": "qemu-img::version::2.0.0"
      },
      {
          "tag": "os::version::14.04"
      },
      {
          "tag": "libvirt::version::1.2.2"
      },
      {
          "tag": "os::release::trusty"
      },
      {
          "tag": "os::distribution::Ubuntu"
```

```
    }
  ],
  "success": true
}
```

2.2 资源属性

插件也可以使用系统标签将新属性添加到资源中。例如，虚拟机的数据库模式中没有列来记录该使用什么 IP 分配算法，什么时候分配虚拟机网卡。这种额外的属性可以用系统标签实现。插件可以创建的系统标签的数量没有限制，附加的插件可以利用这点，并避免干扰数据库模式。

数据库表和系统标签：因为数据库表和系统标签都可以定义资源属性，有时会难以决定属性是应该为数据库模式中的一列，还是应该为一个单独的表中的系统标签。添加新列来修改一个现有的数据库模式，通常需要进行数据库迁移，这是 IaaS 软件升级的一个主要痛点。所以开发者可能更倾向使用系统标签来代表新属性。然而，滥用系统标签是一种错误的编程方式。按照 ZStack 的约定，只应该使用系统标签形式引入非固有的资源属性；系统的标签并不能拯救设计的很烂的数据库表。例如，如果 VM 的数据库表缺失集群 UUID（虽然不会），即使需要进行数据库迁移也必须补充回来；但为了私人使用而被用户创建的插件引入的部门 ID 应该作为一个系统标签实现。这种权衡有时候并不容易，我们会严格控制任何数据库结构的变化。

2.3 元编程

系统标签也可以标注资源以影响 ZStack 的执行流，它在某种程度上类似于 Metaprogramming（https://en.wikipedia.org/wiki/Metaprogramming）。

例如，管理员可以在 KVM 主机上创建一个系统标签 `reservedMemory::1G`，提示 ZStack 主机分配器从主机的可用内存保留 1G 内存；如果管理员改变心意，他可以通过删除标签来回收这 1G 内存。有很多类似的系统标签。

例如，在用户标签这一节中，我们提到了同时使用*用户标签* `SSD` 和*系统标签*来为 VM 的根云盘指定主存储。系统标签叫 `primaryStorage::allocator::userTag::{tag}::required`，如果一个*虚拟机实例规格*上有 `primaryStorage::allocator::userTag::SSD::required`，从该*虚拟机实例规格*上创建的虚拟机根云盘的任务，将只被分配到拥有*用户标签*为 `SSD` 的主存储上。有许多称之为解释点 `interpreting points` 的代码，将在执行过程中寻找特定的系统标签，可以改变代码的默认行为。

2.4 第三方软件集成

建立在 ZStack 上的第三方软件可以使用系统标签在 ZStack 的数据库中存储和资源关联的信息，这能有效避免第三方软件数据库和 ZStack 数据库的数据不一致性。

例如，一个私有软件可能需要记录虚拟机的部门 ID 来审计每个部门 IT 资源的使用情况，这个功能通常由一个私有的数据库完成，并迫使私有软件跟踪虚拟机的生命周期，因为它需要在数据库创建或销毁时，去更新自己的数据库。否则，数据将不会正确反映真实情况。

有了系统标签的帮助，私有软件可以使用系统标签，例如 `audit::departmentId::{id}`将信息存储在 ZStack 的数据库，将管理*部门 ID* 生命周期的责任转移给 ZStack。当一个虚拟机被销毁，它的*部门 ID*（例如 `audit::departmentId::1`）将在删除该虚拟机记录的数据库事务中被自动删除。此外，私有软件可以用它们的部门 ID 调用常规查询 API 检索虚拟机：

```
QueryVmInstance fields=uuid __sysTag__=audit::departmentId::1
```

注：在 ZStack 版本（0.6）中，我们还没开放允许定义任意系统标签的接口，所有的系统标签都是预先定义的。我们计划在下一个版本中开放这个接口，用户定义的系统标签可以在创建的时候添加一些系统允许的前缀，例如，`3rd::`。

和其他组件的关系

标签系统是 ZStack 核心组件之一；它不仅具有单独的 API 和服务，而且还和其他核心组件无缝集成。用户可以在资源创建时或在资源创建后创建标签。ZStack 所有创造型的 API 支持两个固有参数：`userTags` 和 `systemTags`，通过它们传递的标签将随着资源一起创建。例如:

```
CreateVmInstance name=testTag systemTags=hostname::web-server-1
l3NetworkUuids=6572ce44c3f6422d8063b0fb262cbc62

instanceOfferingUuid=04b5419ca3134885be90a48e372d3895

imageUuid=f1205825ec405cd3f2d259730d47d1d8
```

如果资源已经存在，用户可以使用*标签API*来创建或删除标签：

```
CreateUserTag resourceType=VmInstanceVO resourceUuid=613af3fe005914c1643a15c36fd578c6 tag=web

DeleteTag uuid=596070a6276746edbf0f54ef721f654e
```

资源被删除时，与资源相关联的标签将被自动删除。

资源可以通过使用两个特殊的查询条件进行查询：`__userTag__`and`__systemTag__`标签：

```
QueryVmInstance __userTag__=web zoneUuid=04b5419ca3134885be90a48e372d3895

QueryHost __systemTag__=capability:liveSnapshot
```

也有查询 API 专门用于分类：

```
QueryUserTag resourceUuid=0cd1ef8c9b9e0ba82e0cc9cc17226a26 tag~=web-server-%

QuerySystemTag resourceUuid=50fcc61947f7494db69436ebbbefda34
```

总结

在这篇文章中，我们展示了 ZStack 的标签系统。通过这个系统，用户、插件和第三方软件可以通过各种各样的方式使用标签，而不用改变代码和数据库表结构。这是又一个基本点，激发了一种潜力，使得 ZStack 在快速进化为一个成熟的、完整的云计算解决方案的同时，又能保持核心架构强壮稳定。