术语表

下面是一系列的 Ansible 文档中的术语。

在主界面查看文档，了解这些术语的上下文环境，不过这里是一个好的资源测试你对 Ansible 组件的了解和更好的理解他们是如何组合起来的。你可能在想回顾 Ansible 知识的时候看看这里，在邮件中出现一些术语的时候参考一下这个文档。

动作(Action)

一个动作属于一个任务的一部分，指定运行的模块，然后传递参数给此模块。每个任务之一一个动作，但是它可以有不同的参数。

Ad Hoc

指的是使用 /usr/bin/ansible 运行 Ansible 直接执行一些命令，而不是使用 /usr/bin/ansible-playbook 执行剧本。一个 ad-hoc 命令例子，可以是在你的基础设施里面重启50台机器。任何你可以做的东西， ad-hoc 都可以实现通过写一个剧本, 剧本肯定也组合了其它的一些操作。

Async

指的是一个任务配置为运行在后台，而不是等它完成。如果你有一个很长的任务要执行，而且时长可能超出 SSH 登录时长， 那么运行那个任务在 async 方式比较有意义。Async 方式可以每隔一段时间 poll 一次，等待此任务完成。它可以调整为把任务踢出去，然后不再理会它，以便后来使用。Async方式可以在 /usr/bin/ansible 和 /usr/bin/ansible-playbook下面。

Callback Plugin

指一些用户编写的代码可以从 Ansible 运行结果获取数据,并做出一些处理。 一些提供的在 Github 项目上的事例实现了自定义日志，发邮件，甚至播放声音效果。

Check Mode

指的是运行 Ansible 使用 `--ckeck` 选项，但是系统本身却不作出任何改变，仅仅输出可能发生的改变。这就像在其它系统上叫做“dry run”的方式， 用户应该被警告因为这个方式没考虑到命令失败的问题，或者冲突影响。使用这个可以知道哪些东西可能会发生，但是这不是一个好的替代 staging 环境。

Connection Type, Connection Plugin

Ansible 默认用可插拔的库和远端系统通信。 Ansible 支持天然的 OpenSSH (‘ssh’) 或者 Python 实现的 ‘paramiko’ 库。如果你在使用最近的 Ansible 版本，最好使用 OpenSSH ， 同时支持 Kerberos 和 jump hosts。这在文档开始部分就有提到。也有一些加速方式的连接类型，但是必须 bootstrapped 基于SSH类型的连接，但是它非常快，就像在本地系统上运行一样。用户也可以写他们自己的连接类型插件。

Conditionals

一个条件式是根据一个表达式正确或错误判断是否在一个机器上执行给定的任务。 Ansible 的条件表达式由 ‘when’ 提供，在playbook文档里面有讨论。

Diff Mode

`--diff``标识可以传递给 Ansible 来展示模板文件如何改变的，或者使用 ``--check` 模式时它们可能发生的改变 。这些 diffs 统一为diff格式。

Facts

Facts 是发现远端节点的信息。当它们被用在模板的时候， facts只能被引用，而不能被设置。Facts是当运行 plays 时候执行内部的’setup’模块自动收集的。你不需要明确的调用 setup 模块，它自己运行，但是当你想节省时间的时候你可以禁止它。为了方便用户转向其他系统配置工具， fact 模块可以拉取 facts 从 Chef的’ohai’ 和 Puppet的’facter’工具。

Filter Plugin

过滤插件式大多数用户从来不需要了解的东西。这允许创在新的 Jinja2 过滤，而这只对那些知道什么是 Jinja2 过滤的人有帮助。如果你需要他们，你可以从 API docs 部分学习如何写他们。

Forks

Ansible 与远端节点交流是通过并行的机制，并行机制的方式可以通过传递 ``–forks``参数设置，或者在配置文件里面编辑。默认是保守的5个线程。如果你有足够的内存，你可以很容易的设置为50或者更多值。

Gather Facts (Boolean)

上面已经提到了Facts。有时候在运行多个 playbook ， 可能不想收集一些fact ， 而且以后也不会用到这些值。在playbook里面设置 *gather_facts:False* 指示跳过收集 facts。

Globbing

Globbing 是一个一种基于通配符的方式挑选许多主机，而不是明确指定主机的名字，或者它们的组名。例如，使用 “www*”，来匹配所有以 “www” 开头的所有主机。这个理念直接被吸收进 Func 。除此之外，不同的 set 操作也可以通过 globbing 实现。

Group

组由几个主机组成，可以方便的当做一个目标看待，同时可以共享变量。

Group Vars

group_vars 文件位于一个目录下面，同时在 inventory 旁边，有一个可选的文件名在每个组后面。这是一个方便的位置来存放变量，提供给每个组，由其是复杂的数据结构，因此这些变量不需要嵌入在 inventory 文件或 playbook 文件里面。

Handlers

Handler 仅仅是普通的任务在Ansible playbook里面(请参考tasks)。但是仅仅当任务包含 “notify” 指令和指示它改变了一些东西的时候才运行。例如，如果一个配置文件改变了，然后任务引用这个配置文件模板通知服务器重启 handler 。这意味着服务可以被反弹仅仅他们需要重启的时候。Handler 不仅仅可以用于重启服务，但是重启服务是最通用的用法。

Host

一个host 只是简单的 Ansible 管理的远端机器。它们可以被分配私有的变量，可以被组织为一个组。所有的组有可以访问一个名字，也可以是IP地址，如果他们在默认的SSH端口不能访问，可以指定一个一个可选的端口号

Host Specifier

每一个 Play 映射为一系列的 tasks (可以是定义的role，purpose，或系统指令) 到一些系统的集合

“hosts:” 指令在每个play中通常叫做主机指定。

它可以挑选一个系统，一个或更多组，甚至一些主机在其他组不在某个组里面，但是在另外一个组里面。

Host Vars

就像”Group Vars”，一个名称为 “host_vars/” 的目录在 inventory 文件旁，可以在 invetory 文件的主机名后面包含这个文件，使用 YAML 格式。这提供一个方便的位置分配变量给这个主机而不要在 inventory 文件里面嵌入太多变量。Host Vars 文件还可以用于定义复杂的在 inventory 文件里面不断出现的数据结构。

Lazy Evaluation

总的来说， Ansible 评估任何变量在 playbook 内容在最新的可能的时间里，也就是意味着如果你定义了一个数据结构，这个数据结构自身也可以定义变量值在里面，然后每件事情就像你期望的那样工作。 这也意味着 变量字符串可以包含其它的变量在字符串里面。

Lookup Plugin

一个查询插件是从外界得到数据进入 Ansible 。这些东西就像 “with_items” ，一个基础的循环插件，但是也有其它的查询插件就像 “with_file”, 从文件加载数据，甚至有一些逡巡环境变量， DNS 文本记录，或者键值存储。 查询插件也可以被 templates 访问 ， `{{ lookup('file','/path/to/file') }}` .

Multi-Tier

IT 系统不是一次在同一时间只管理一个系统，而是在多个系统之间交互，一组系统，在一个定义好的顺序里面。例如，一个 web server 可能需要在数据库服务器之前更新，web server 的部分内容又要在 *THAT* 数据库服务之后更新，同时不同的负载均衡器和监控服务器也需要被联系到。 Ansible 看待系统为整个工作流和拓扑，而不是简单的一次一个系统。

Idempotency

改变类的命令仅仅在他们需要使用的时候才被使用，最好描述系统的状态而不是如何到达系统某个状态的过程。打个比方，从美国的卡罗莱纳州到加利福尼亚州包括驾驶很长一段距离的车，但是如果我是在阿拉斯加州，则需要乘坐地铁。 Ansible的资源就像你说，“把我放到加利福尼亚”然后决定如何到达那里。如果你已经在加利福尼亚，没有什么会发生，然后他会让你知道什么都没有发生，不需要改变什么东西。

Includes

Playbook 文件可以包含其它的 plays，任务列表也可以扩展在其它文件的外部任务，就像处理器。 Include 可以被参数化的，也就是装载文件可以传递变量。例如，一个Include 表演设置 Wordpress 博客站点，需要传递”user”参数,然后这个表演(play)可以 include 多于一次的博客站点，例如叫做 “alice” 和 “bob”

Inventory

一个描述主机和组的 Ansible 文件。Inventory 可以通过 “Inventory Script” 提供，有时也叫做 “External Inventory Script”

Inventory Script

一个简单的从外部资源寻找主机,主机组的成员，和变量信息的程序 – 可以是个 SQL 数据库，一个 CMDB 解决方案，或者是 LDAP。这个概念来自 Puppet (叫"External Nodes Classifier")，工作方式也是类似的。

Jinja2

Jinja2 是 Ansible 模板的首选语言。它非常简单，很容易阅读和书写。

JSON

Ansible 从远端机器上返回的数据使用 JSON 类型。这使得模块可以使用任何语言编写，而不仅仅是Python。

Library

许多模块的集合供 /usr/bin/ansible 或 Ansible Playbook 使用。

Limit Groups

通过传递 `--limit somegroup` 参数给 ansible 或 ansible-playbook ， 命令可以限制为一些主机的子集 。 例如这可以使目标为全部的服务器到只允许一个服务器运行 playbook 。

Local Connection

通过在 playbook 中使用 "connection:local" ， 或者传递 "-c local" 给 /usr/bin/ansible ， 这指明了我们正在管理本地主机而不是远端机器。

Local Action

local_action 指令在 playbook 意味着给予的步骤仅仅会在本地机器上运行， 但是这变量 '{{ ansible_hostname }}'可以被传递到远端机器引用。这可以被用于触发器，例如，rsync 操作。

Loops

通常来说， Ansible 不是一个编程语言。它跟喜欢声明，尽管不同的结果像 "with_items" 使得指定的任务重复的实验多个 items 在一个列表里面。特定的模块，例如 yum 和 apt ， 对这更喜欢，可以安装多个包，然后加速了配置的总时间。

Modules

Module 是 Ansible 运行远端机器的单元。模块可以使用通过 /usr/bin/ansible 或者 /usr/bin/ansible-playbook 。模块可以通过任何语言编写包括 Perl，Bash，Ruby，但是使用 Python 可以利用一些有用的社区库代码。模块仅仅返回一些 JSON 格式数据或简单的

key=value 集合。一旦模块在远端执行之后，他们就被移除了，隐私不需 daemon 长时间运行。Ansible 把模块的集合看做 'library'

Notify

等级改变的事件和通知处理任务需要在 play 的最后运行。如果一个 handler 被多个任务通知，它会仍然仅仅运行一次。 Handler仅仅按照列表运行一次，而不是他们被notified 的顺序。

Orchestration

一些软件自动化系统使用这个单词意味着不同的事情。 Ansible使用它作为一个导演执导一个曲子。一个数据中心或云架构充满多个系统，表演很多角色 – web servers，database servers，负载均衡器，监控系统， 持续集成系统等。在具体表演过程中，必须要安排好特定的步骤。一些系统执行一些步骤，然后其它系统，然后先前的系统执行更多的步骤。同时，发送邮件也可能是需要的到 web service 联系人。 Ansible 编排了所有过程的模型。

paramiko

默认， Ansible 管理机器使用 SSH。而 Ansible 默认使用的 python 提供的库叫 paramiko。paramiko库非常的快和很容易管理，渴望支持 Kerberos 或 jump Host 的用户转向使用 SSH 作为连接类型了。在他们的 playbook里面使用 "-c ssh" 选项即可。

Playbooks

Playbooks 是一种语言，Ansible 用于编排，配置，管理和部署吸引。他们被叫做 Playbooks 的部分原因是依据它行为的类比，使用它应该是一件有趣的事情。他们不是 工作书。

Plays

一个 playbook 就是一系列的 plays。一个 play 就是在一些主机中挑选指定的主机和主机组，然后运行任务在这些主机上，定义这些主机的角色和他们会怎么样表演。

Pull Mode

Pull 模式是节点每隔 N 分钟检查特定的主机。它使用 ansible-pull 程序，pull模式有很多选择性。Ansible-pull 在任务计划中检查配置指令熟悉怒，使用连接插件，在本地管理机器。

Push Mode

push 模式是 Ansible 的默认模式。事实上，这也不算是个模式 – 你不去想它的时候 ansible 就是这么工作的。Push 方式通过复杂的编排进程，而不要等到节点检查，对节点有个很好的粒度控制。

Register Variable

Ansible 运行的结果可以存储在一个变量里面以便模板或条件语句使用，用于定义这个变量的关键字叫做 'register'。你可以定义无限制的变量名用于 registertion.

Resource Model

Ansible 模块工作在资源上。例如，file 模块会挑选指定的文件然后确保资源的属性匹配指定的模型。例如，我们想改变 /etc/motd 的属主为 'root'，如果它还没设置为 root,或者设置权限为'0644',如果还没有设置为 0644 。资源模型是幂等性('idemotent')意味着改变命令不会运行除非需要的时候，Ansible会把系统变为期望的状态而不管当前的状态是什么。

Roles

一个 Role 可以包含特定的变量值，特定的任务，特定的触发器等东西。因为 Role 的文件结构，roles 可以是再次利用的单元，可以让你在其它 playbooks 中共享一些行为。

Rolling Update

一次处理某组主机的 N 个节点，避免一次全部更新导致系统离线。 例如，在一个 500 节点的 web 拓扑里，最好一次更新 10~20 台机器一次。Ansible 中的 'seria' 关键字控制 rolling updtae的池。默认是一次全部处理。OS 配置可以不使用 rolling update 模型，但是可以这么做。

Runner

Ansible 核心的组件是 /usr/bin/ansible 指令，它背后有强大的力量，激发 playbook 中的每个任务。 Runner 一般是 Ansible 开发者经常谈论的，但是它对用户来说不是经常用到的词汇。

Serial

参考 "Rolling Update".

Sudo

Ansible 不要求一定用 root 登录，它是无守护进程模式的(这可能是个安全问题，在敏感的环境里面)。 Ansible可以记录一些运行 sudo 命令的操作，可以运行无密码的和有密码的 sudo。一些操作不需要使用 sudo (像 scp 文件传输)可以通过 Ansible 的 copy,template,和 fetch 模块实现。

SSH (Native)

OpenSSH 作为 Ansible 的传输被指定使用 “-c ssh”，这可以很有用当你想登陆通过 Kerberized SSH 或者 SSH jump hosts 等待。在 1.2.1版本，ssh被用作默认，之前使用 ‘paramiko’ 作为默认。使用一个客户端 支持 ControlMaster 和 ControlPersist 是被推荐的对于管理大量主机。如果你不需要使用 Kerbers，jump hosts或者其它的特性， 选择 paramiko 是不错的选择。Ansible 会发出警告，如果它没有检测到 ControlMaster/ControlPersist 兼容性。

Tags

Ansible 允许给playbook里面的资源通过自定义的关键字打上标签，然后只运行与关键字一致的部分代码。 例如，可能有个完成的 OS 配置，然后特定的步骤标记为 “ntp” ，然后运行 “ntp” 步骤来重新配置时间服务器信息。

Tasks

Playbooks 包含 Tasks， Tasks 结合一个动作使用一个名称和一些可选的关键字。处理器也是 tasks，但是他们是特殊的 tasks 不运行，除非他们被通知一个 tasks 报道的远端吸引变化。

Templates

Ansible 很容易的传输文件到远端系统上面，但是它经常需要替换一些变量在其它的文件里面。变量可以来自 清单文件，Host Vars， Group Vars,或者 Facts。Templates 使用 Jinja2 模板引擎同样可以包含逻辑控制像循环和 if 语句。

Transport

Ansible 使用 “Connection Plugins” 定义可用的传输类型。这只是 Ansible 如何到达管理的系统。Rransports 包括 paramiko, SSH (using OpenSSH), 和 local.

When

一个可选的关键字来决定这个任务是不是应该指向，如果再 “when:” 关键字这里的表达式是是不正确的，这个任务会被忽略。

Van Halen

没有其它的原因，Michael 真的很喜欢他们，所有的 Ansible 版本代号都是以 Van Halen 的歌曲命名。

Vars (Variables)

和 Facts 相反， 变量是一些值，或字典，列表的名称(可以是标量值–整数，布尔型，或字符串，字典，列表)，然后变量可以应用在模板和剧本里面。他们是声明的东西，不是获取远程系统的当前状态或性质(这是Facts)

YAML

Ansible 不想强迫人们编写编程语言的代码实现自动化基础设施部署,所以 Ansible 使用YAML来定义剧本还配置语言和变量文件。YAML很棒因为它有很少的语法，然后非常干净,容易浏览。对人来说，这是一个很好的数据格式的配置文件,机器也可读。YAML非常流行在动态语言社区，编程语言也有库可用来序列化这种语言.

See also

常见问题
: 常问到的问题

Playbooks
: Playbook 介绍

最佳实践
: 最好的联系见习

User Mailing List
: Have a question? Stop by the google group!

irc.freenode.net
: #ansible IRC chat channel