云服务器 ECS

最佳实践

最佳实践

安全

在云端安全组提供类似虚拟防火墙功能，用于设置单个或多个 ECS 实例的网络访问控制，是重要的安全隔离手段。创建 ECS 实例时，您必须选择一个安全组。您还可以添加安全组规则，对某个安全组下的所有 ECS 实例的出方向和入方向进行网络控制。

本文主要介绍如何配置安全组的入网规则。

安全组相关的信息

在配置安全组的入网规则之前，您应已经了解以下安全组相关的信息：

- 安全组限制
- 安全组默认规则
- 设置安全组 In 方向的访问权限
- 设置安全组 Out 方向的访问权限

安全组实践的基本建议

在开始安全组的实践之前，下面有一些基本的建议：

- 最重要的规则：安全组应作为白名单使用。
- 开放应用出入规则时应遵循“最小授权”原则，例如，您可以选择开放具体的端口（如 80 端口）。
- 不应使用一个安全组管理所有应用，因为不同的分层一定有不同的需求。
- 对于分布式应用来说，不同的应用类型应该使用不同的安全组，例如，您应对 Web、Service、Database、Cache 层使用不同的安全组，暴露不同的出入规则和权限。
- 没有必要为每个实例单独设置一个安全组，控制管理成本。
- 优先考虑 VPC 网络。
- 不需要公网访问的资源不应提供公网 IP。
- 尽可能保持单个安全组的规则简洁。因为一个实例最多可以加入 5 个安全组，一个安全组最多可以包括 100 个安全组规则，所以一个实例可能同时应用数百条安全组规则。您可以聚合所有分配的安全规则以判断是否允许流入或留出，但是，如果单个安全组规则很复杂，就会增加管理的复杂度。所以，应尽可能地保持单个安全组的规则简洁。

- **调整线上的安全组的出入规则是比较危险的动作。如果您无法确定，不应随意更新安全组出入规则的设置。**阿里云的控制台提供了克隆安全组和安全组规则的功能。如果您想要修改线上的安全组和规则，您应先克隆一个安全组，再在克隆的安全组上进行调试，从而避免直接影响线上应用。

设置安全组的入网规则

以下是安全组的入网规则的实践建议。

不要使用 0.0.0.0/0 的入网规则

允许全部入网访问是经常犯的错误。使用 0.0.0.0/0 意味着所有的端口都对外暴露了访问权限。这是非常不安全的。正确的做法是，先拒绝所有的端口对外开放。安全组应该是白名单访问。例如，如果您需要暴露 Web 服务，默认情况下可以只开放 80、8080 和 443 之类的常用TCP端口，其它的端口都应关闭。

```
{ "IpProtocol" : "tcp", "FromPort" : "80", "ToPort" : "80", "SourceCidrIp" : "0.0.0.0/0", "Policy": "accept"} ,
{ "IpProtocol" : "tcp", "FromPort" : "8080", "ToPort" : "8080", "SourceCidrIp" : "0.0.0.0/0", "Policy": "accept"} ,
{ "IpProtocol" : "tcp", "FromPort" : "443", "ToPort" : "443", "SourceCidrIp" : "0.0.0.0/0", "Policy": "accept"} ,
```

关闭不需要的入网规则

如果您当前使用的入规则已经包含了 0.0.0.0/0，您需要重新审视自己的应用需要对外暴露的端口和服务。如果确定不想让某些端口直接对外提供服务，您可以加一条拒绝的规则。比如，如果您的服务器上安装了 MySQL 数据库服务，默认情况下您不应该将 3306 端口暴露到公网，此时，您可以添加一条拒绝规则，如下所示，并将其优先级设为100，即优先级最低。

```
{ "IpProtocol" : "tcp", "FromPort" : "3306", "ToPort" : "3306", "SourceCidrIp" : "0.0.0.0/0", "Policy": "drop",
Priority: 100} ,
```

上面的调整会导致所有的端口都不能访问 3306 端口，极有可能会阻止您正常的业务需求。此时，您可以通过授权另外一个安全组的资源进行入规则访问。

授权另外一个安全组入网访问

不同的安全组按照最小原则开放相应的出入规则。对于不同的应用分层应该使用不同的安全组，不同的安全组应有相应的出入规则。

例如，如果是分布式应用，您会区分不同的安全组，但是，不同的安全组可能网络不通，此时您不应该直接授权 IP 或者 CIDR 网段，而是直接授权另外一个安全组 ID 的所有的资源都可以直接访问。比如，您的应用对 Web、Database 分别创建了不同的安全组：sg-web 和 sg-database。在sg-database 中，您可以添加如下规则，授权所有的 sg-web 安全组的资源访问您的 3306 端口。

```
{ "IpProtocol" : "tcp", "FromPort" : "3306", "ToPort" : "3306", "SourceGroupId" : "sg-web", "Policy": "accept",
Priority: 2} ,
```

授权另外一个 CIDR 可以入网访问

经典网络中，因为网段不太可控，建议您使用安全组 ID 来授信入网规则。

VPC 网络中，您可以自己通过不同的 VSwitch 设置不同的 IP 域，规划 IP 地址。所以，在 VPC 网络中，您可以默认拒绝所有的访问，再授信自己的专有网络的网段访问，直接授信可以相信的 CIDR 网段。

```
{ "IpProtocol" : "icmp", "FromPort" : "-1", "ToPort" : "-1", "SourceCidrIp" : "10.0.0.0/24", Priority: 2} ,
{ "IpProtocol" : "tcp", "FromPort" : "0", "ToPort" : "65535", "SourceCidrIp" : "10.0.0.0/24", Priority: 2} ,
{ "IpProtocol" : "udp", "FromPort" : "0", "ToPort" : "65535", "SourceCidrIp" : "10.0.0.0/24", Priority: 2} ,
```

变更安全组规则步骤和说明

变更安全组规则可能会影响您的实例间的网络通信。为了保证必要的网络通信不受影响，您应先尝试以下方法放行必要的实例，再执行安全组策略收紧变更。

注意：执行收紧变更后，应观察一段时间，确认业务应用无异常后再执行其它必要的变更。

- 新建一个安全组，将需要互通访问的实例加入这个安全组，再执行变更操作。
- 如果授权类型为 **安全组访问**，则将需要互通访问的对端实例所绑定的安全组 ID 添加为授权对象；
- 如果授权类型为 **地址段访问**，则将需要互通访问的对端实例内网 IP 添加为授权对象。

具体操作指引请参见 经典网络内网实例互通设置方法。

本文将介绍安全组的以下几个内容：

- 授权 和 撤销 安全组规则。
- 加入安全组 和 离开安全组。

阿里云的网络类型分为 **经典网络** 和 **VPC**，它们对安全组支持不同的设置规则：

- 如果是经典网络，您可以设置以下几个规则：内网入方向、内网出方向、公网入方向和公网出方向。
- 如果是 VPC 网络，您可以设置：入方向 和 出方向。

安全组内网通讯的概念

本文开始之前，您应知道以下几个安全组内网通讯的概念：

- 默认只有同一个安全组的 ECS 实例可以网络互通。即使是同一个账户下的 ECS 实例，如果分属不同安全组，内网网络也是不通的。这个对于经典网络和 VPC 网络都适用。所以，经典网络的 ECS 实例也是内网安全的。
- 如果您有两台 ECS 实例，不在同一个安全组，您希望它们内网不互通，但实际上它们却内网互通，那么，您需要检查您的安全组内网规则设置。如果内网协议存在下面的协议，建议您重新设置。
    - 允许所有端口；
    - 授权对象为 CIDR 网段 (SourceCidrIp)：0.0.0.0/0 或者 10.0.0.0/8 的规则。

如果是经典网络，上述协议会造成您的内网暴露给其它的访问。

- 如果您想实现在不同安全组的资源之间的网络互通，您应使用安全组方式授权。对于内网访问，您应使用源安全组授权，而不是 CIDR 网段授权。

安全规则的属性

安全规则主要是描述不同的访问权限，包括如下属性:

- Policy：授权策略，参数值可以是 accept（接受）或 drop（拒绝）。
- Priority：优先级，根据安全组规则的创建时间降序排序匹配。规则优先级可选范围为 1-100，默认值为 1，即最高优先级。数字越大，代表优先级越低。
- NicType：网络类型。如果只指定了 SourceGroupId 而没有指定 SourceCidrIp，表示通过安全组方式授权，此时，NicType 必须指定为 intranet。
- 规则描述：
    - IpProtocol：IP 协议，取值：tcp | udp | icmp | gre | all。all 表示所有的协议。
    - PortRange：IP 协议相关的端口号范围：
        - IpProtocol 取值为 tcp 或 udp 时，端口号取值范围为 1~65535，格式必须是“起始端口号/终止端口号”，如“1/200”表示端口号范围为1~200。如果输入值为“200/1”，接口调用将报错。
        - IpProtocol 取值为 icmp、gre 或 all 时，端口号范围值为 -1/-1，表示不限制端口。
    - 如果通过安全组授权，应指定 SourceGroupId，即源安全组 ID。此时，根据是否跨账号授权，您可以选择设置源安全组所属的账号 SourceGroupOwnerAccount；
    - 如果通过 CIDR 授权，应指定 SourceCidrIp，即源 IP 地址段，必须使用 CIDR 格式。

授权一条入网请求规则

在控制台或者通过 API 创建一个安全组时，入网方向默认 deny all，即默认情况下您拒绝所有入网请求。这并不适用于所有的情况，所以您要适度地配置您的入网规则。

比如，如果您需要开启公网的 80 端口对外提供 HTTP 服务，因为是公网访问，您希望入网尽可能多访问，所以在 IP 网段上不应做限制，可以设置为 0.0.0.0/0，具体设置可以参考以下描述，其中，括号外为控制台参数，括号内为 OpenAPI 参数，两者相同就不做区分。

- 网卡类型（NicType）：公网（internet）。如果是 VPC 类型的只需要填写 intranet，通过 EIP 实现公网访问。
- 授权策略（Policy）：允许（accept）。
- 规则方向（NicType）：入网。
- 协议类型（IpProtocol）：TCP（tcp）。
- 端口范围（PortRange）：80/80。
- 授权对象（SourceCidrIp）：0.0.0.0/0。
- 优先级（Priority）: 1。

注意：上面的建议仅对公网有效。内网请求不建议使用 CIDR 网段，请参考 经典网络的内网安全组规则不要使

用 CIDR 或者 IP 授权。

禁止一个入网请求规则

禁止一条规则时，您只需要配置一条拒绝策略，并设置较低的优先级即可。这样，当有需要时，您可以配置其它高优先级的规则覆盖这条规则。例如，您可以采用以下设置拒绝 6379 端口被访问。

- 网卡类型（NicType）：内网（intranet）。
- 授权策略（Policy）：拒绝（drop）。
- 规则方向（NicType）：入网。
- 协议类型（IpProtocol）：TCP（tcp）。
- 端口范围（PortRange）：6379/6379。
- 授权对象（SourceCidrIp）：0.0.0.0/0。
- 优先级（Priority)：100。

经典网络的内网安全组规则不要使用 CIDR 或者 IP 授权

对于经典网络的 ECS 实例，阿里云默认不开启任何内网的入规则。内网的授权一定要谨慎。

为了安全考虑，不建议开启任何基于 CIDR 网段的授权。

对于弹性计算来说，内网的 IP 经常变化，另外，这个 IP 的网段是没有规律的，所以，对于经典网络的内网，建议您通过安全组授权内网的访问。

例如，您在安全组 sg-redis 上构建了一个 redis 的集群，为了只允许特定的机器（如 sg-web）访问这个 redis 的服务器编组，您不需要配置任何 CIDR，只需要添加一条入规则：指定相关的安全组 ID 即可。

- 网卡类型（NicType）：内网（intranet）。
- 授权策略（Policy）：允许（accept）。
- 规则方向（NicType）：入网。
- 协议类型（IpProtocol）：TCP（tcp）。
- 端口范围（PortRange）：6379/6379。
- 授权对象（SourceGroupId）：sg-web。
- 优先级（Priority）：1。

对于 VPC 类型的实例，如果您已经通过多个 VSwitch 规划好自己的 IP 范围，您可以使用 CIDR 设置作为安全组入规则；但是，如果您的 VPC 网段不够清晰，建议您优先考虑使用安全组作为入规则。

将需要互相通信的 ECS 实例加入同一个安全组

一个 ECS 实例最多可以加入 5 个安全组，而同一安全组内的 ECS 实例之间是网络互通的。如果您在规划时已经有多个安全组，而且，直接设置多个安全规则过于复杂的话，您可以新建一个安全组，然后将需要内网通讯的 ECS 实例加入这个新的安全组。

安全组是区分网络类型的，一个经典网络类型的 ECS 实例只能加入经典网络的安全组；一个 VPC 类型的 ECS

实例只能加入本 VPC 的安全组。

这里也不建议您将所有的 ECS 实例都加入一个安全组，这将会使得您的安全组规则设置变成梦魇。对于一个中大型应用来说，每个服务器编组的角色不同，合理地规划每个服务器的入方向请求和出方向请求是非常有必要的。

在控制台上，您可以根据文档 加入安全组 的描述将一个实例加入安全组。

如果您对阿里云的 OpenAPI 非常熟悉，您可以参考 使用 OpenAPI 弹性管理 ECS 实例，通过 OpenAPI 进行批量操作。对应的 Python 片段如下。

```
def join_sg(sg_id, instance_id):
request = JoinSecurityGroupRequest()
request.set_InstanceId(instance_id)
request.set_SecurityGroupId(sg_id)
response = _send_request(request)
return response

# send open api request
def _send_request(request):
request.set_accept_format('json')
try:
response_str = clt.do_action(request)
logging.info(response_str)
response_detail = json.loads(response_str)
return response_detail
except Exception as e:
logging.error(e)
```

将 ECS 实例移除安全组

如果 ECS 实例加入不合适的安全组，将会暴露或者 Block 您的服务，这时您可以选择将 ECS 实例从这个安全组中移除。但是在移除安全组之前必须保证您的 ECS 实例已经加入其它安全组。

注意：将 ECS 实例从安全组移出，将会导致这个 ECS 实例和当前安全组内的网络不通，建议您在移出之前做好充分的测试。

对应的 Python 片段如下。

```
def leave_sg(sg_id, instance_id):
request = LeaveSecurityGroupRequest()
request.set_InstanceId(instance_id)
request.set_SecurityGroupId(sg_id)
response = _send_request(request)
return response

# send open api request
def _send_request(request):
request.set_accept_format('json')
try:
response_str = clt.do_action(request)
```

```
logging.info(response_str)
response_detail = json.loads(response_str)
return response_detail
except Exception as e:
logging.error(e)
```

定义合理的安全组名称和标签

合理的安全组名称和描述有助于您快速识别当前复杂的规则组合。您可以通过修改名称和描述来帮助自己识别安全组。

您也可以通过为安全组设置标签分组管理自己的安全组。您可以在控制台直接 设置标签，也通过 API 设置标签。

删除不需要的安全组

安全组中的安全规则类似于一条条白名单和黑名单。所以，请不要保留不需要的安全组，以免因为错误加入某个 ECS 实例而造成不必要的麻烦。

在安全组的使用过程中，通常会将所有的云服务器放置在同一个安全组中，从而可以减少初期配置的工作量。但从长远来看，业务系统网络的交互将变得复杂和不可控。在执行安全组变更时，您将无法明确添加和删除规则的影响范围。

合理规划和区分不同的安全组将使得您的系统更加便于调整，梳理应用提供的服务并对不同应用进行分层。这里推荐您对不同的业务规划不同的安全组，设置不同的安全组规则。

区分不同的安全组

公网服务的云服务器和内网服务器尽量属于不同的安全组

是否对外提供公网服务，包括主动暴露某些端口对外访问（例如 80、443 等），被动地提供（例如云服务器具有公网 IP、EIP、NAT 端口转发规则等）端口转发规则，都会导致自己的应用可能被公网访问到。

2 种场景的云服务器所属的安全组规则要采用最严格的规则，建议拒绝优先，默认情况下应当关闭所有的端口和协议，仅仅暴露对外提供需要服务的端口，例如 80、443。由于仅对属于对外公网访问的服务器编组，调整安全组规则时也比较容易控制。

对于对外提供服务器编组的职责应该比较明晰和简单，避免在同样的服务器上对外提供其它的服务。例如 MySQL、Redis 等，建议将这些服务安装在没有公网访问权限的云服务器上，然后通过安全组的组组授权来访问。

如果当前有公网云服务器已经和其它的应用在同一个安全组 SG_CURRENT。您可以通过下面的方法来进行变更。

梳理当前提供的公网服务暴露的端口和协议，例如 80、443。

新创建一个安全组，例如 SG_WEB， 然后添加相应的端口和规则。

说明：授权策略：允许，协议类型：ALL， 端口: 80/80，授权对象: 0.0.0.0/0， 授权策略:允许，协议类型:ALL，端口: 443/443 授权对象: 0.0.0.0/0。

选择安全组 SG_CURRENT， 然后添加一条安全组规则，组组授权，允许 SG_WEB 中的资源访问 SG_CURRENT。

说明：授权策略：允许，协议类型：ALL，端口：-1/-1，授权对象：SG_WEB，优先级：按照实际情况自定义[1-100]。

将一台需要切换安全组的实例 ECS_WEB_1 添加到新的安全组中。

i. 在 ECS 控制台中，选择 **安全组管理**。
ii. 选择 **SG_WEB** > **管理实例** > **添加实例**，选择实例 ECS_WEB_1 加入到新的安全组 SG_WEB 中，确认 ECS_WEB_1 实例的流量和网络工作正常。

将 ECS_WEB_1 从原来的安全组中移出。

i. 在 ECS 控制台中，选择 **安全组管理**。
ii. 选择 **SG_CURRENT** > **管理实例** > **移出实例**，选择 ECS_WEB_1 ，从 SG_CURRENT 移除，测试网络连通性，确认流量和网络工作正常。
iii. 如果工作不正常，将 ECS_WEB_1 仍然加回到安全组 SG_CURRENT 中，检查设置的 SG_WEB 暴露的端口是否符合预期，然后继续变更。

执行其它的服务器安全组变更。

不同的应用使用不同的安全组

在生产环境中，不同的操作系统大多情况下不会属于同一个应用分组来提供负载均衡服务。提供不同的服务意味着需要暴露的端口和拒绝的端口是不同的，建议不同的操作系统尽量归属于不同的安全组。

例如，对于 Linux 操作系统，可能需要暴露 TCP（22）端口来实现 SSH，对 Windows 可能需要开通 TCP(3389) 远程桌面连接。

除了不同的操作系统归属不同的安全组，即便同一个镜像类型，提供不同的服务，如果之间不需要通过内网进行访问的话，最好也划归不同的安全组。这样方便解耦，并对未来的安全组规则进行变更，做到职责单一。

在规划和新增应用时，除了考虑划分不同的虚拟交换机配置子网，也应该同时合理的规划安全组。使用网段+安全组约束自己作为服务提供者和消费者的边界。

具体的变更流程参见上面的操作步骤。

生产环境和测试环境使用不同的安全组

为了更好的做系统的隔离，在实际开发过程中，您可能会构建多套的测试环境和一套线上环境。为了更合理的做网络隔离，您需要对不同的环境配置使用不通的安全策略，避免因为测试环境的变更刷新到了线上影响线上的稳定性。

通过创建不同的安全组，限制应用的访问域，避免生产环境和测试环境联通。同时也可以对不同的测试环境分配不同的安全组，避免多套测试环境之间互相干扰，提升开发效率。

仅对需要公网访问子网或者云服务器分配公网 IP

不论是经典网络还是专有网络 (VPC) 中，合理的分配公网 IP 可以让系统更加方便地进行公网管理，同时减少系统受攻击的风险。在专有网络的场景下，创建虚拟交换机时，建议您尽量将需要公网访问的服务区的 IP 区间放在固定的几个交换机(子网 CIDR)中，方便审计和区分，避免不小心暴露公网访问。

在分布式应用中，大多数应用都有不同的分层和分组，对于不提供公网访问的云服务器尽量不提供公网IP，如果是有多台服务器提供公网访问，建议您配置公网流量分发的负载均衡服务来公网服务，提升系统的可用性，避免单点。

对于不需要公网访问的云服务器尽量不要分配公网 IP。专有网络中当您的云服务器需要访问公网的时候，优先建议您使用 NAT 网关，用于为 VPC 内无公网 IP 的 ECS 实例提供访问互联网的代理服务，您只需要配置相应的 SNAT 规则即可为具体的 CIDR 网段或者子网提供公网访问能力，具体配置参见 SNAT。避免因为只需要访问公网的能力而在分配了公网 IP(EIP) 之后也向公网暴露了服务。

最小原则

安全组应该是白名单性质的，所以需尽量开放和暴露最少的端口，同时尽可能少地分配公网 IP。若想访问线上机器进行任务日志或错误排查的时候直接分配公网 IP 或者挂载 EIP 虽然简便，但是毕竟会将整个机器暴露在公网之上，更安全的策略是建议通过跳板机来管理。

使用跳板机

跳板机由于其自身的权限巨大，除了通过工具做好审计记录。在专有网络中，建议将跳板机分配在专有的虚拟交换机之中，对其提供相应的 EIP 或者 NAT 端口转发表。

首先创建专有的安全组 SG_BRIDGE，例如开放相应的端口，例如 Linux TCP(22) 或者 Windows RDP(3389)。为了限制安全组的入网规则，可以限制可以登录的授权对象为企业的公网出口范围，减少被登录和扫描的概率。

然后将作为跳板机的云服务器加入到该安全组中。为了让该机器能访问相应的云服务器，可以配置相应的组授权。例如在 SG_CURRENT 添加一条规则允许 SG_BRIDGE 访问某些端口和协议。

使用跳板机 SSH 时，建议您优先使用 SSH 密钥对而不是密码登录。

总之，合理的安全组规划使您在扩容应用时更加游刃有余，同时让您的系统更加安全。

本文档从云服务器ECS使用的角度出发，结合相关产品和运维架构经验，介绍如何打造云端的数据安全。

适用对象

本文档适用于刚开始接触阿里云的个人或者中小企业用户。

主要内容

- 定期备份数据
- 合理设计安全域
- 安全组规则设置
- 登录口令设置
- 服务器端口安全
- 系统漏洞防护
- 应用漏洞防护
- 安全情报收集

定期备份数据

数据备份是容灾的基础，目的是降低因系统故障、操作失误、以及安全问题而导致数据丢失的风险。云服务器ECS自带有快照备份的功能，合理运用ECS快照功能即可满足大部分用户数据备份的需求。建议用户根据自身的业务情况，制定适合自己的备份策略，您可以选择**手动创建快照**，或者**创建自动快照策略**，并**将此策略应用到指定磁盘**。推荐每日做一次自动快照，每次快照最少保存7天。养成良好的备份习惯，在故障发生时，有利于迅速恢复重要数据，减少损失。

合理设计安全域

基于SDN（Software Defined Network）技术研发的VPC专有网络，可以供用户构建自定义专属网络，隔离企业内部不同安全级别的服务器，避免互通网络环境下一台服务器感染后影响到其它应用服务器。

建议用户**创建专有网络**，选择自有 IP 地址范围、划分网段、配置路由表和网关等。用户可以将比较重要的数据存储在一个跟互联网网络完全隔离的内网环境，日常运维可以用弹性IP（EIP）或者跳板机的方式，对数据进行管理。

安全组规则设置

安全组是重要的网络安全隔离手段，用于设置单台或多台云服务器的网络访问控制。用户通过安全组设置实例级别的防火墙策略，可以在网络层过滤服务器的主动/被动访问行为，限定服务器对外/对内的的端口访问，授权访问地址，从而减少攻击面，保护服务器的安全。

例如Linux系统默认远程管理端口22，不建议向外网直接开放，可以通过**设置安全组配置ECS公网访问控制**，只

授权本地固定IP对服务器进行访问；您可以查看其它应用案例，加深对安全组的熟悉程度。对访问控制有更高要求的用户或者也可以使第用三方VPN产品，对登录行为进行数据加密，更多软件尽在云市场。

登录口令设置

弱口令一直是数据泄露的一个大症结，因为弱口令是最容易出现的也是最容易被利用的漏洞之一。服务器的口令建议至少8位以上，从字符种类上增加口令复杂度，如包含大小写字母、数字和特殊字符等，并且要不定时更新口令，养成良好的安全运维习惯。

服务器端口安全

服务器只要给互联网提供服务，就会将对应的服务端口暴露在互联网，从安全管理的角度来说，开启的服务端口越多，就越不安全。建议只对外开放提供服务的必要端口，并修改常见端口为高端口（30000以后），再对提供服务的端口做访问控制。

例如数据库服务尽量在内网环境使用，避免暴露在公网；如果必须要在公网访问，则需要修改默认连接端口3306为高端口，并根据业务授权可访问客户端地址。

系统漏洞防护

系统漏洞问题这种长期都存在的安全风险，可以通过系统补丁程序，或者安骑士补丁管理来解决。Windows系统的补丁更新要一直开启，Linux系统要设置定期任务执行yum update -y来更新系统软件包及内核。

云盾旗下的安骑士产品，可以主动检测网站后门，第一时间打补丁修复漏洞，同时还能识别防御非法破解密码的行为，避免被黑客多次猜解密码而入侵，批量维护服务器安全。安骑士同时还提供针对服务器应用软件不安全的配置检测和修复方案，帮助用户成功修复弱点，提高服务器安全强度。强烈推荐用户使用。

应用漏洞防护

应用漏洞是指针对Web应用、缓存、数据库、存储等服务，通过利用渗透攻击而非法获取数据的一种安全缺陷。常见应用漏洞包括：SQL注入、XSS跨站、Webshell上传、后门隔离保护、命令注入、非法HTTP协议请求、常见Web服务器漏洞攻击、核心文件非授权访问、路径穿越等。这种漏洞不同于系统漏洞，修复存在很大难度，如果程序在设计应用之初，不能对这些应用安全基线面面俱到，服务器安全的堡垒，就往往在这最后一公里被攻破。所以我们推荐通过接入Web应用防火墙(Web Application Firewall, 简称 WAF)这种专业的防护工具，来轻松应对各类Web应用攻击，确保网站的Web安全与可用性。

安全情报收集

在当今暗流涌动的互联网安全领域，安全工程师和黑客比拼的就是时间，云盾态势感知可以理解为一种基于大数据的安全服务，即在大规模云计算环境中，对能够引发网络安全态势发生变化的要素进行全面、快速和准确地捕获和分析。然后把客户当前遇到的安全威胁与过去的威胁进行关联回溯和大数据分析，最终产出未来可能发生的威胁安全的风险事件，并提供一个体系化的安全解决方案。

所以，技术人员除了在做好日常安全运维的同时，还要尽可能掌握全面的信息，提升预警能力，在发现安全问题的时候可以及时进行修复和处理，才能真正保证云服务器ECS的数据安全闭环。

云服务器 ECS 实例是一个虚拟的计算环境，包含了 CPU、内存、操作系统、磁盘、带宽等最基础的服务器组件，是 ECS 提供给每个用户的操作实体。

我们基本可以理解为一个实例就等同于一台虚拟机，那么我们在本地维护的虚拟机一般会做虚拟机实例级别的安全防护，以防止虚拟机被攻击和入侵等。同样的，云上的ECS实例也需要做安全性防护。

ECS实例放置在云上，除了置身于阿里云自身的安全平台外，用户也需要根据实际的需求进一步定制化安全，所以说ECS的安全是阿里云和用户共同构建的。如果ECS实例没有安全的防护，可能会带来不少不良的影响，比如遭受到DDoS而导致业务中断，比如受到Web入侵而导致网页被篡改、挂马，比如被注入而导致信息和数据泄漏等，影响ECS的使用和无法正常提供服务。

一般可以通过设置安全组、AntiDDoS、态势感知、安装安骑士、接入Web应用防火墙等方式提高ECS实例的安全性。下面就从实例层面分别讲解一下如何提高ECS实例的安全性。

设置安全组

安全组是一个逻辑上的分组，这个分组是由同一个地域（Region）内具有相同安全保护需求并相互信任的实例组成。每个实例至少属于一个安全组，在创建的时候就需要指定。同一安全组内的实例之间网络互通，不同安全组的实例之间默认内网不通。可以授权两个安全组之间互访。

设置安全组的好处

安全组是一种虚拟防火墙，具备状态检测包过滤功能。安全组用于设置单台或多台云服务器的网络访问控制，它是重要的网络安全隔离手段，用于在云端划分安全域。安全组规则可以允许或者禁止与安全组相关联的云服务器 ECS 实例的公网和内网的入出方向的访问。

如果没有很好地设置安全组或者安全组规则过于开放，则降低了访问的限制级别，在一定程度上为攻击者敞开了大门。

操作步骤

1、登录 云服务器管理控制台。

2、单击左侧导航中的 **安全组**。

3、选择地域。

4、单击**添加安全组规则**。

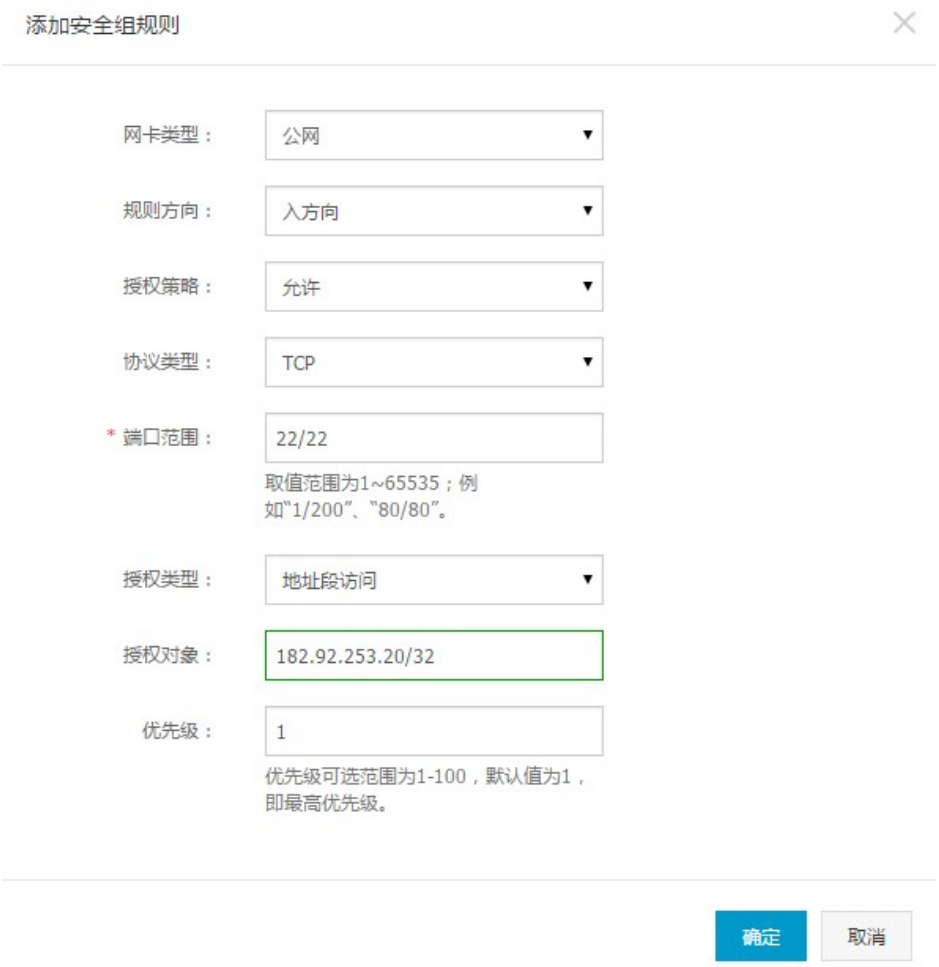
5、在弹出的对话框中，分别设置网络类型、规则方向、授权策略、协议类型、端口范围、授权类型、授权对象和优先级。

6、点击 **确定**，成功为该安全组授权一条安全组规则 。

下面结合一个案例来阐述一下，比如只允许特定IP远程登录到实例。

通过配置安全组规则可以设置只让特定 IP 远程登录到实例。只需要在公网入方向配置规则就可以了，以 Linux 服务器为例，设置只让特定 IP 访问 22 端口。

- 添加一条公网入方向安全组规则，允许访问，协议类型选择 TCP，端口写 22/22，授权类型为地址段访问，授权对象填写允许远程连接的 IP 地址段，格式为 x.x.x.x/xx，即 IP地址/子网掩码，本例中的地址段为 182.92.253.20/32。优先级为 1



- 再添加一条规则，拒绝访问，协议类型选择 TCP，端口写 22/22，授权类型为地址段访问，授权对象写所有 0.0.0.0/0，优先级为 2

最终的效果如下：

来自 IP 182.92.253.20 访问 22 端口优先执行优先级为 1 的规则允许。

来自其他 IP 访问 22 端口优先执行优先级为 2 的规则拒绝了。

AntiDDoS

阿里云云盾可以防护SYN Flood，UDP Flood，ACK Flood，ICMP Flood，DNS Flood，CC攻击等3到7层DDoS的攻击。DDoS基础防护免费为阿里云用户提供最高5G的默认DDoS防护能力。

阿里云在此基础上，推出了安全信誉防护联盟计划，将基于安全信誉分进一步提升DDoS防护能力，用户最高可获得100G以上的免费DDoS防护资源。

为什么需要AntiDDoS

DDoS（Distributed Denial of Service）即分布式拒绝服务。攻击指借助于客户/服务器技术，将多个计算机联合起来作为攻击平台，对一个或多个目标发动DDoS攻击，从而成倍地提高拒绝服务攻击的威力，影响业务和应用正常对用户提供服务。

使用AntiDDoS，无需采购昂贵清洗设备，可以在受到DDoS攻击不会影响访问速度，带宽充足不会被其他用户连带影响，保证业务可用和稳定。

操作步骤

1、进入阿里云官网，登录到 **管理控制台**。

2、输入用户名密码。

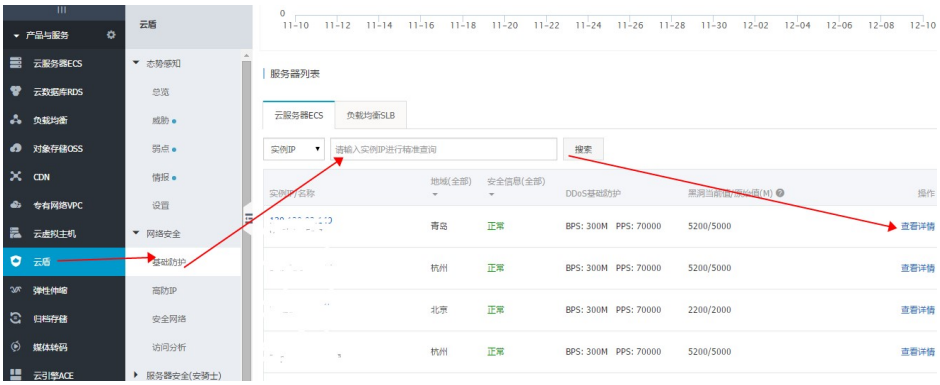
3、通过**云盾>DDOS防护>基础防护**，查看基础防护配置。

4、可以加入安全信誉防护联盟。勾选服务条款，点选加入安全信誉防护联盟加入联盟。如下图所示。

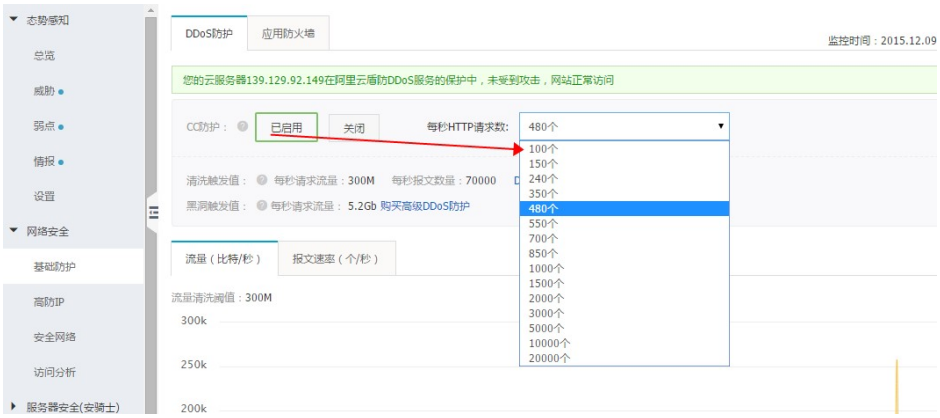
云盾DDoS基础版提供不大于5G的DDoS防护，在此基础上推出了安全信誉防护联盟计划，您可通过加入此联盟，在获得原默认防护能力基础上，会得到免费增量防护带宽机会。

加入联盟后，可查看自己的安全信誉分，并查看安全信誉组成，维护安全信誉，获得更大的防护能力。加盟成功后在基础防护界面显示如下信誉界面。

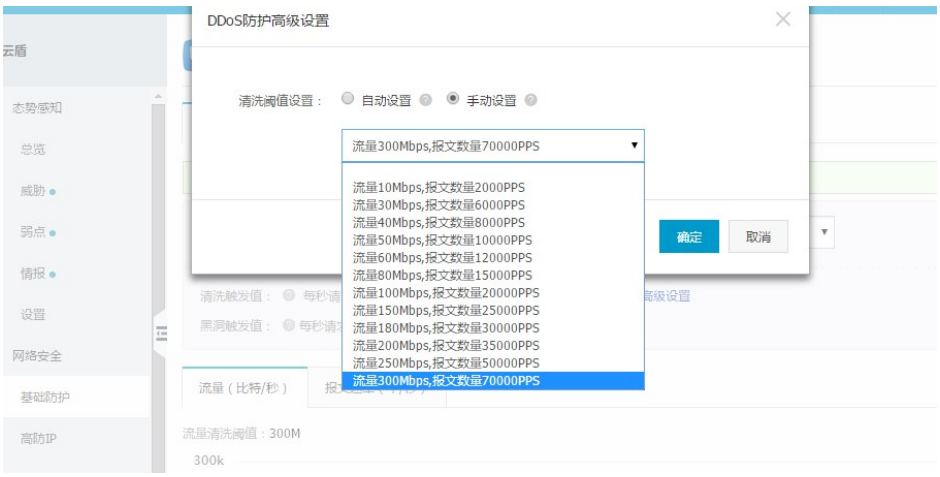
5、【基础防护】点击对应ECS服务器的【查看详情】，如果服务器数量比较多，可以在【云服务器ecs】列表中通过【实例IP】和【实例名称】搜索服务器，再点击对应服务器的【查看详情】。



6、进入页面后，可以在【CC防护】点击【已启用】开启CC防护，点击【关闭】则关闭CC防护功能，在【每秒HTTP请求数】可以对每秒http请求数设置清洗阈值，达到阈值后便会触发云盾的清洗。



7、如果购买了高级DDoS防护，可以点击【DDoS防护高级设置】可以设置清洗阈值，选择【自动设置】后系统会根据云服务器的流量负载动态调整清洗阈值，选择【手动设置】可以手动对流量和报文数量的阈值进行设置，当超过此阈值后云盾便会开启流量清洗(建议如果网站在做推广或者活动时适当调大)。



态势感知

态势感知态势感知提供的是一项SAAS服务，即在大规模云计算环境中，对那些能够引发网络安全态势发生变化的要素进行全面、快速和准确地捕获和分析。然后，把客户当前遇到的安全威胁与过去的威胁进行关联回溯和大数据分析，最终产出未来可能产生的安全事件的威胁风险，并提供一个体系化的安全解决方案。

态势感知的优势

对“渗透攻击”有所感知，以云计算数据平台支撑，因此具有强大的安全数据分析能力，对各种常见类型的攻击可以实时分析和展示。

操作步骤

1、在阿里云用户控制台-《云盾》-《态势感知》中点击免费开启服务，即可使用态势感知。

2、通过紧急时间、威胁、弱点、情报、日志等方面，辅以直观的可视化的分析，让安全一目了然。

安装安骑士

服务器安全(安骑士)是云盾推出的一款服务器安全运维管理产品。通过安装在服务器上的轻量级Agent插件与云

端防护中心的规则联动，实时感知和防御入侵事件，保障服务器的安全。

安装安骑士的好处

安骑士是很轻量的，服务器上运行的Agent插件，正常状态下只占用1%的CPU、10MB内存。安骑士可以自动识别服务器的Web目录，对服务器的Web目录进行后门文件扫描，支持通用Web软件漏洞扫描和Windows系统漏洞扫描，对服务器常见系统配置缺陷进行检测，包括可疑系统账户、弱口令、注册表等进行检测。

我们可以将安骑士理解为ECS实例上的防病毒软件，如果没有安骑士，相当于少了一个可靠的卫士，我们ECS实例的健康性水平也会相应降低。

操作步骤

1、服务器安全(安骑士)Agent插件目前集成于安全镜像中，在购买ECS后，一般都已经默认安装，您可以进入**安骑士控制台-配置中心**，查看每台服务器的在线状态。

2、若不在线，请按照如下方式下载并安装。

1) 进入服务器安全(安骑士)控制台-设置-安装Agent页面，根据页面提示获取最新版本下载地址，以管理员权限在服务器上运行并安装。

2) 对于非阿里云服务器，在安装过程中会提示输入验证Key，这个验证Key用于关联阿里云账号，通过阿里云账号在安骑士控制台使用相关功能，验证key会显示在安装页面中。

3) 大约安装完成2分钟后在云盾·服务器安全(安骑士)控制台-配置中心里查看到在线数据，阿里云服务器将会从离线变成在线，非阿里云机器会新增在服务器列表中。

接入Web应用防火墙

云盾Web应用防火墙(Web Application Firewall, 简称 WAF)基于云安全大数据能力实现，通过防御SQL注入、XSS跨站脚本、常见Web服务器插件漏洞、木马上传、非授权核心资源访问等OWASP常见攻击，过滤海量恶意CC攻击，避免您的网站资产数据泄露，保障网站的安全与可用性。

接入Web应用防火墙的好处

无需安装任何软、硬件，无需更改网站配置、代码，它可以轻松应对各类Web应用攻击，确保网站的Web安全与可用性，淘宝天猫都在用。除了具有强大Web防御能力，还可以指定网站的专属防护，背后是大数据的安全能力。适用于在金融、电商、o2o、互联网+、游戏、政府、保险、政府等各类网站的Web应用安全防护上。

如果缺少WAF，光靠前面提到的防护措施会存在短板，例如在面对如数据泄密、恶意CC、木马上传篡改网页等攻击的时候，就不能拿很好地防护了，可能会导致Web入侵。

操作步骤

1、控制台配置。

1) 登录阿里云控制台，找到云盾->Web应用防火墙->域名配置，点击“添加域名”按钮。

2) 弹出的对话框中输入相关信息：

3) 获取CNAME。配置好域名后，WAF会自动分配给当前域名一个CNAME，可点击域名信息来查看：

4) 上传HTTPS证书和私钥（仅针对HTTPS站点）。如果防护HTTPS站点，必须上传服务器的证书和私钥到WAF，否则访问HTTPS站点会有问题。勾选HTTPS后，会看到红色的“异常”字样，提示当前证书有问题，点击“上传证书”来上传：

5) 接入状态异常排查，刚添加完域名时，接入状态可能会提示异常。这是正常的，待修改DNS使用CNAME解析接入WAF后，或者是有正常流量经过WAF以后会变成正常的。

2、放行回源IP段。

3、本地验证。

1) 以前面步骤中添加的域名 “www.aliyundemo.cn” 为例，hosts文件应该添加如下内容，其中前面的IP地址为对应的WAFIP地址，WAF的IP可以通过ping提供的CNAME来获得。

```
# localhost name resolution is handled within DNS itself.
#       127.0.0.1       localhost
#       ::1             localhost

[redacted]58.255 www.aliyundemo.cn
```

2) 修改hosts文件后保存。然后本地ping一下被防护的域名，预期此时解析到的IP地址应该是刚才绑定的WAF IP地址。如果依然是源站地址，可尝试刷新本地的DNS缓存（Windows的cmd下可以使用ipconfig/flushdns命令）。

3) 确认hosts绑定已经生效（域名已经本地解析为WAF的IP）后，打开浏览器，输入该域名进行访问，如果WAF的配置正确，网站预期能够正常打开。

4) 尝试一下手动模拟一些简单的web攻击命令，如www.aliyundemo.cn/?alert(xss) 预期WAF能够弹出阻拦页面：

4、通过DNS供应商或者其他域名解析系统，修改DNS解析。

阿里云给我们ECS实例的安全性提供了这么多的安全产品保驾护航，我们可以根据实际需要选择相应的产品，加强对系统和数据的防护，减少ECS实例接受到的侵害，使其稳定、持久地运行。

修改 Windows 服务器默认远程端口

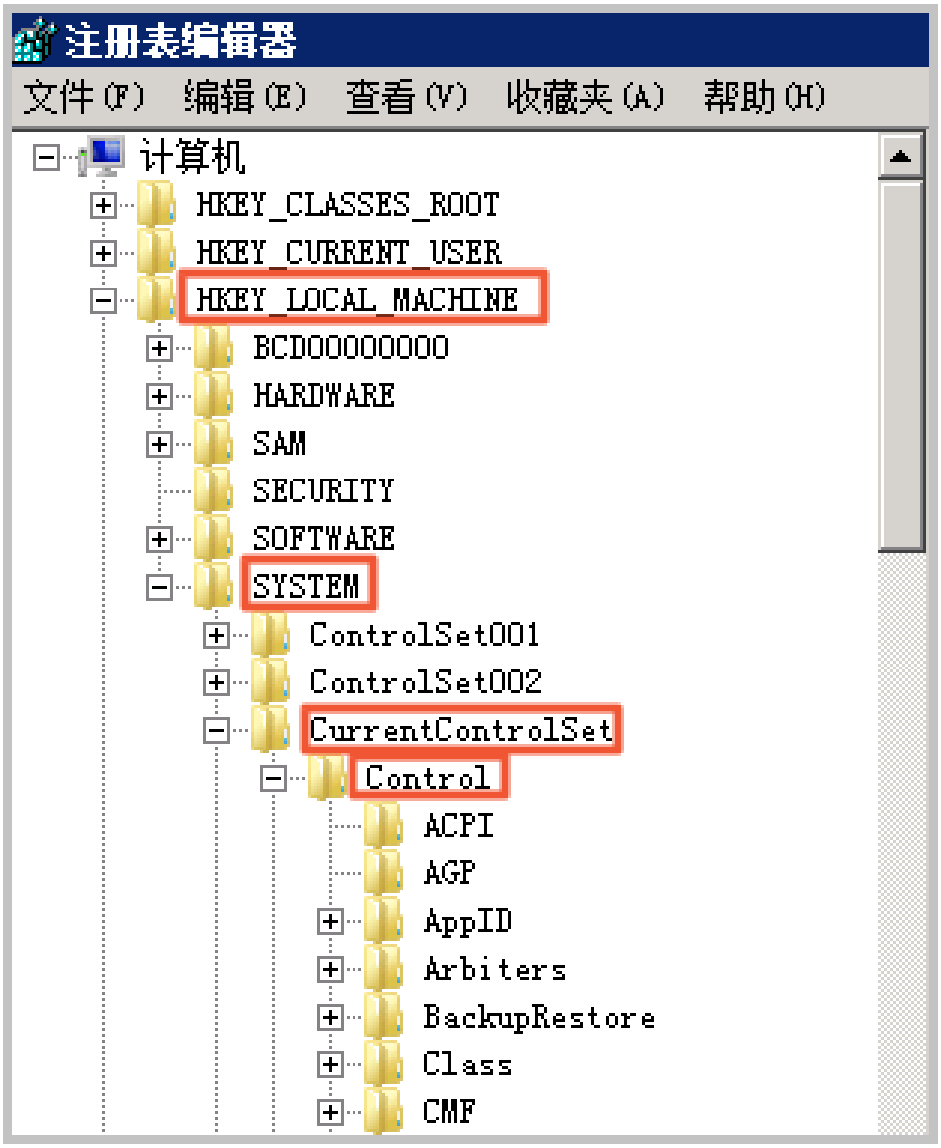
操作步骤

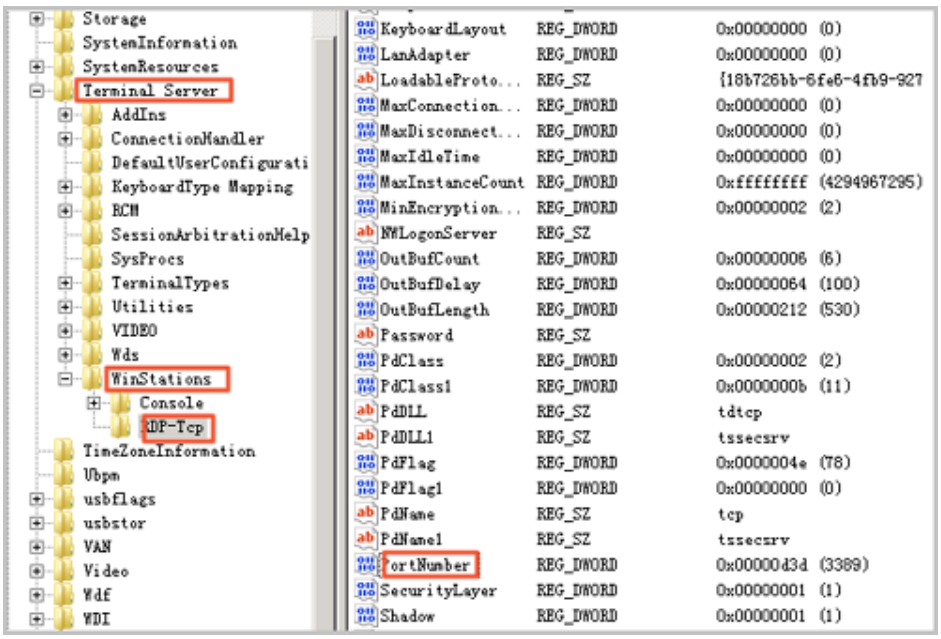
远程连接并登录到 Windows 实例。

运行regedit.exe打开注册表编辑器.

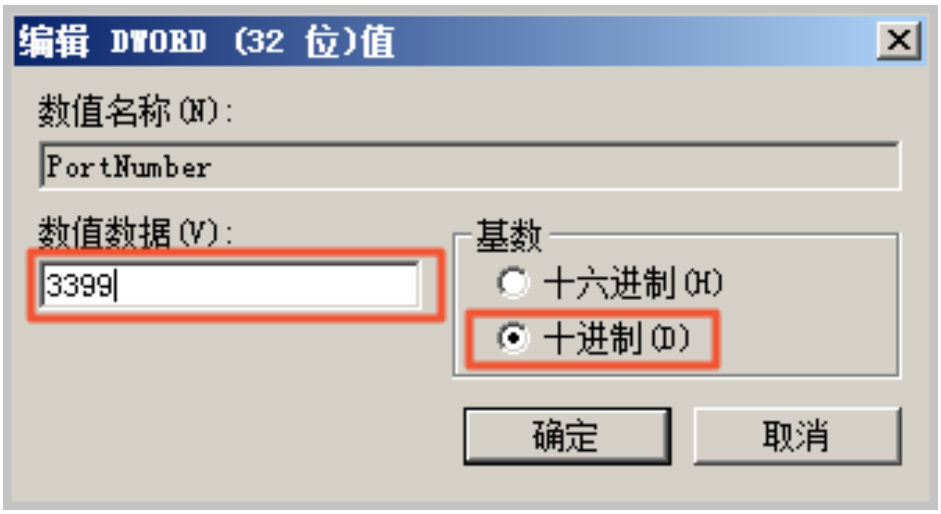
找到如下注册表子项：

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Terminal Server\WinStations\RDP-Tcp\PortNumber





在弹出的对话框中，选择**十进制**，在**数值数据**中输入新的远程端口号，在本例中即 3399。单击**确定**。



（可选）如果您开启了防火墙，需要将新的端口号添加到防火墙并设置允许连接。具体方法参见**设置 ECS 实例远程连接防火墙**中的**添加端口规则**章节。

登录 ECS 管理控制台，找到该实例，选择**更多>重启**。

实例重新启动后，在实例的右侧单击**管理**，进入**实例详情**页面。选择**本实例安全组**。

在**安全组列表**页面，找到相应的安全组，单击**配置规则**。

在**安全组规则**页面，单击**添加安全组规则**。根据实际的使用场景来定义安全规则，允许新配置的远程端口进行连接。关于如何设置安全组参见**添加安全组规则**。

以上步骤完成后，远程访问服务器，在远程地址后面添加新远程端口号即可连接实例。例如：192.168.1.2:3399。

注意：调整 3389 端口后，使用 Mac 的远程桌面连接客户仅支持默认的 3389 端口。

修改 Linux 服务器默认远程端口

本节以 CentOS 6.8 为例介绍如何修改 Linux 服务器默认远程端口。

操作步骤

远程连接并登录到 Linux 实例。

运行 vim /etc/ssh/sshd_config命令。

在键盘上按 I 键，进入编辑状态。将 22 端口修改成目标端口，本节以 1022 端口为例。在Port 22下输入Port 1022。

在键盘上按 **ESC**，输入：wq退出编辑状态。

执行以下命令重启实例，之后您可以通过 22 端口和 1022 端口 SSH 登录到 Linux 实例。

```
/etc/init.d/sshd restart
```

（可选）配置防火墙。使用 CentOS 7 以前的版本并开启默认防火墙 iptables 时，应注意 iptables 默认不拦截访问，如果您配置了 iptables 规则，需要执行iptables -A INPUT -p tcp --dport 1022

-j ACCEPT配置防火墙。然后执行service iptables restart 重启防火墙。

> **说明**：CentOS 7 以后版本默认安装 Firewalld。如果您已经启用 firewalld.service，需要放行 TCP 1022 端口：运行命令 firewall-cmd —add-port=1022/tcp —permanent。返回结果为 success 即表示已经放行 TCP 1022 端口。

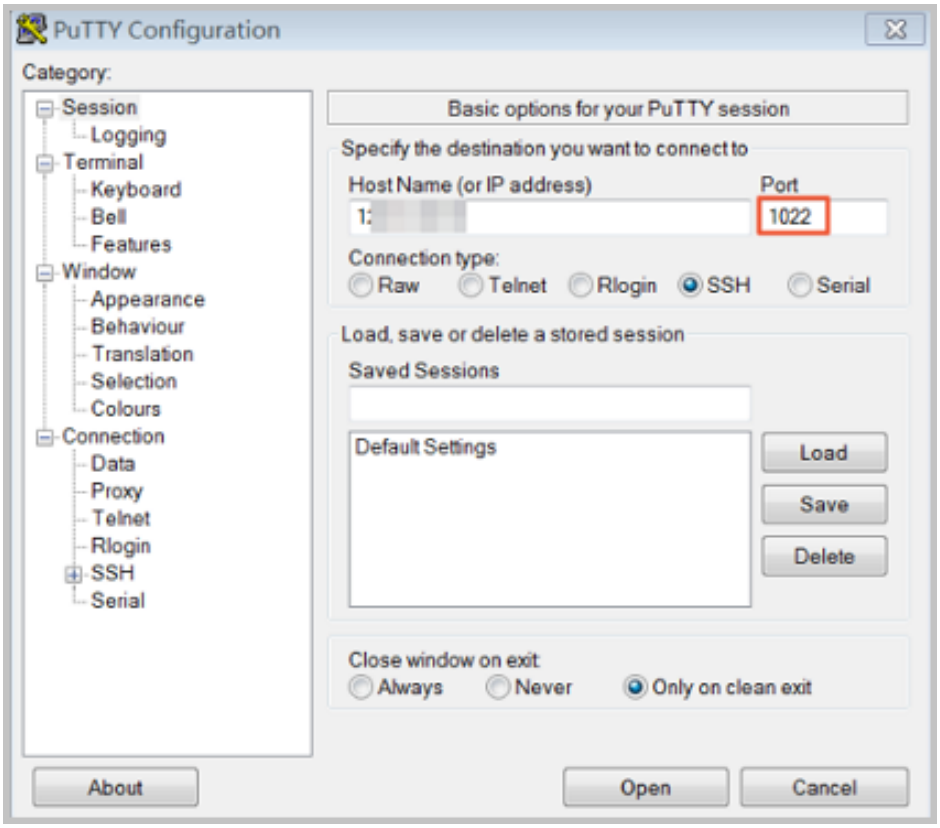
登录 ECS 管理控制台，找到该实例，选择**管理**。

进入**实例详情**页面。选择**本实例安全组**。

在**安全组列表**页面，找到相应的安全组，单击**配置规则**。

在**安全组规则**页面，单击**添加安全组规则**。根据实际的使用场景来定义安全规则，允许新配置的远程端口进行连接。关于如何设置安全组参见**添加安全组规则**。

使用 SSH 工具连接新端口，来测试是否成功。登录时在 **Port** 一栏输入新修改的端口号，在本例中即 1022。



使用 1022 端口连接成功后，再次运行vim /etc/ssh/sshd_config命令，将 Port 22 删除。

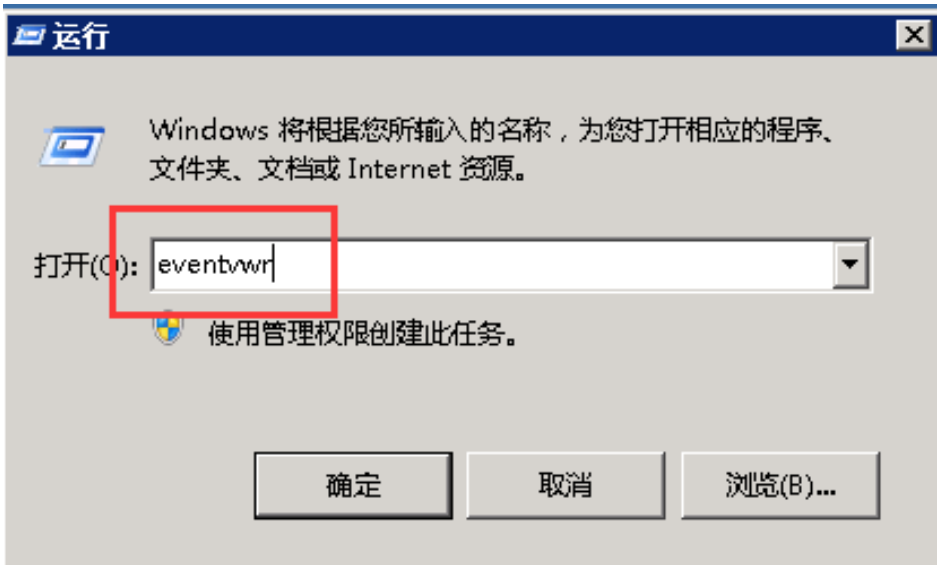
运行 /etc/init.d/sshd restart 命令重启实例，服务器默认远程端口修改完成。再次登录时使用新端口号登录即可。

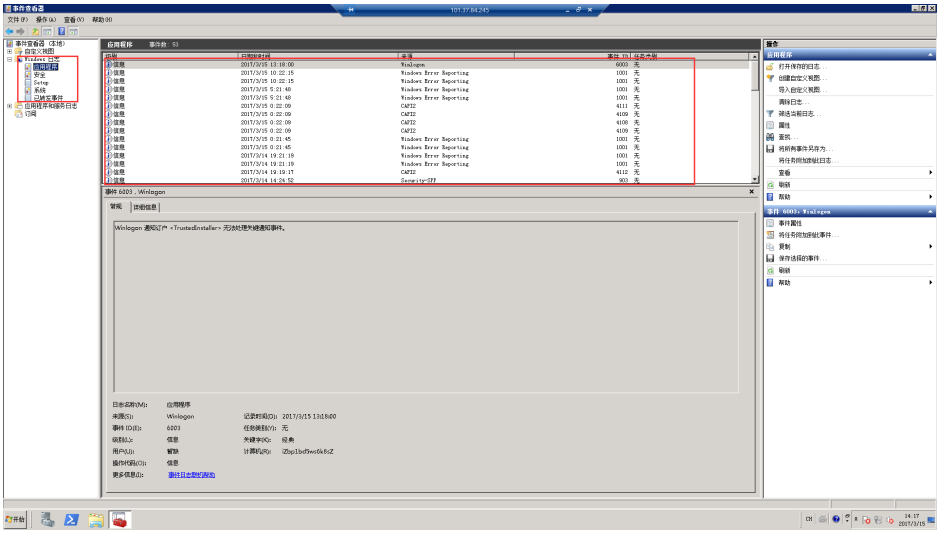
注意：请不要直接对 22 端口进行修改。之所以先设置成两个端口，测试成功后再关闭一个端口，是为了防止在修改配置文件及网络调试过程中，万一出现新端口无法连接的情况下，还能通过 22 端口进行登录调试。

简介

日志是记录系统中硬件、软件和系统问题的信息，同时还可以监视系统中发生的事件，当服务器被入侵或者系统（应用）出现问题时，管理员可以根据日志来迅速定位到问题的关键，然后对问题在进行快速的处理，这样才可以极大的提高我们的工作效率和服务器的安全性。Windows系统日志主要分为三种，分别是系统日志、应用程序日志和安全日志，还有应用程序和服务日志。接下来以Windows server 2008 R2为例来简单的介绍下四种日志的使用和简要分析。

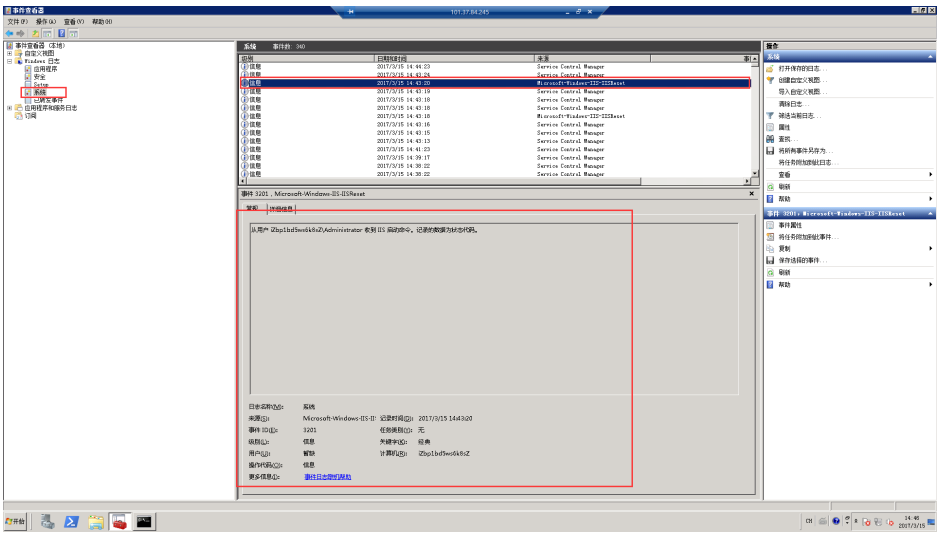
Windows查看系统日志的方法：开始>设置>控制面板>管理工具，中找到的“事件查看器”，或者Win+r键输入”eventvwr”也可以直接进入“事件查看器”。





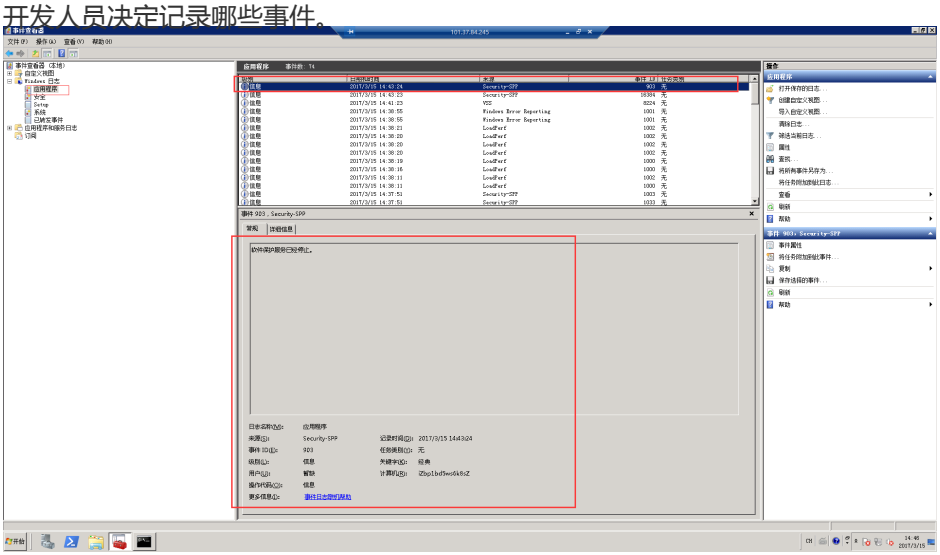
系统日志

系统日志包含 Windows 系统组件记录的事件。例如，在启动过程中加载驱动程序或其他系统组件失败将记录在系统日志中。系统组件所记录的事件类型由 Windows 预先确定。



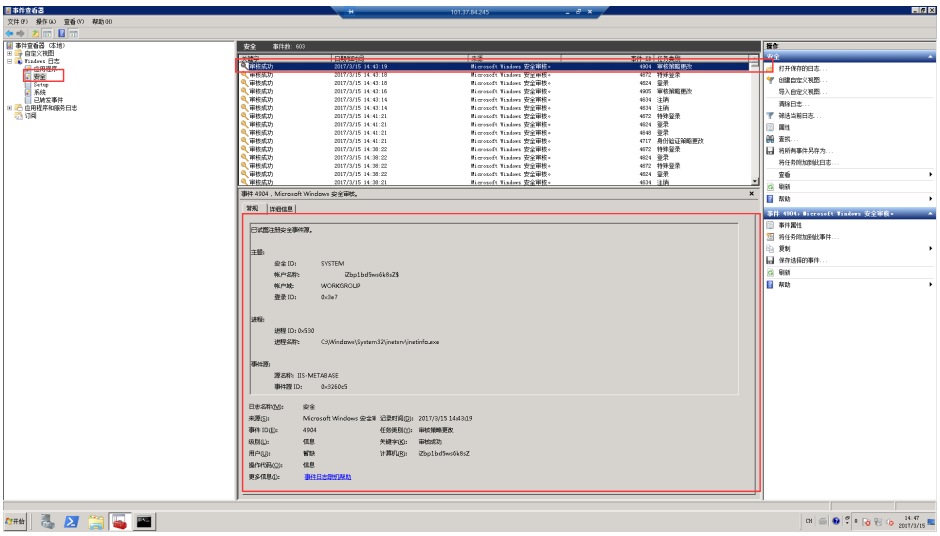
应用程序日志

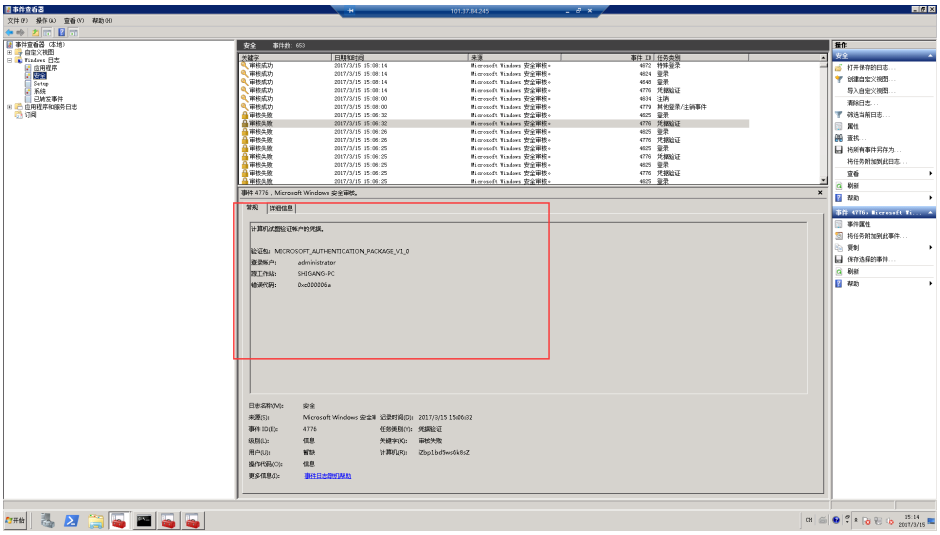
应用程序日志包含由应用程序或程序记录的事件。例如，数据库程序可在应用程序日志中记录文件错误。程序开发人员决定记录哪些事件。



安全日志

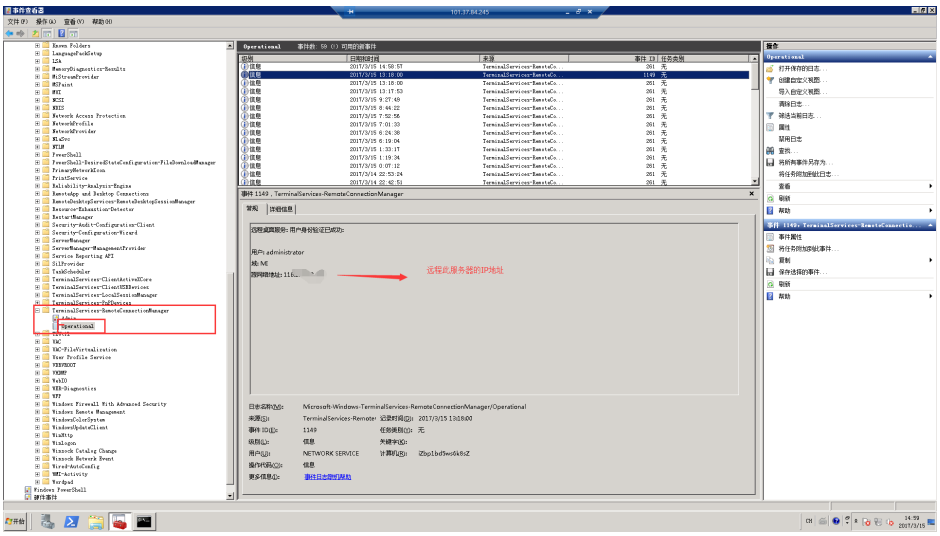
安全日志包含诸如有效和无效的登录尝试等事件，以及与资源使用相关的事件，如创建、打开或删除文件或其他对象。管理员可以指定在安全日志中记录什么事件。例如，如果已启用登录审核，则对系统的登录尝试将记录在安全日志中。





应用程序和服务日志

应用程序和服务日志是一种新类别的事件日志。这些日志存储来自单个应用程序或组件的事件，而非可能影响整个系统的事件。

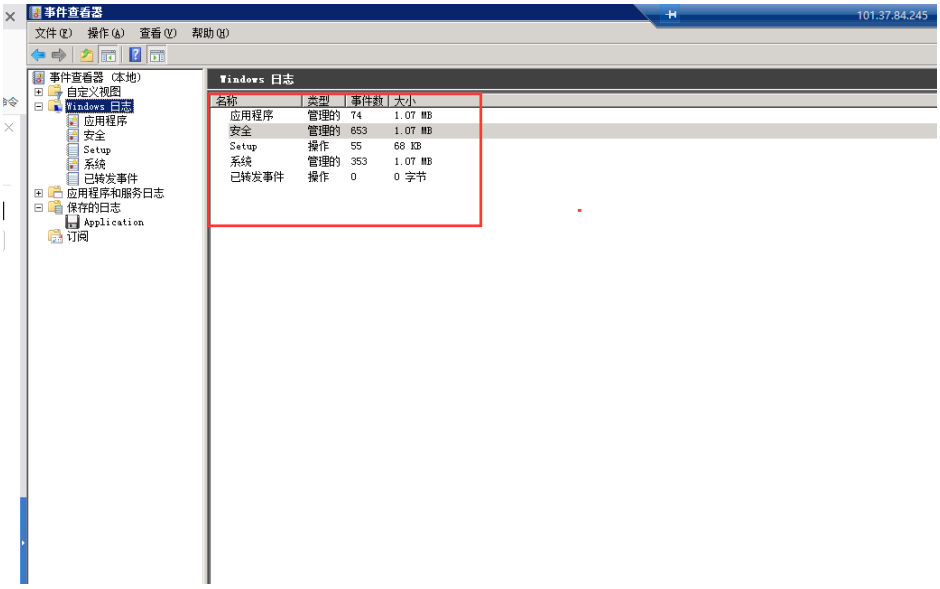


以上是四种日志的查看方法，可以针对所有错误日志的事件ID来对比微软知识库来找到解决方法。

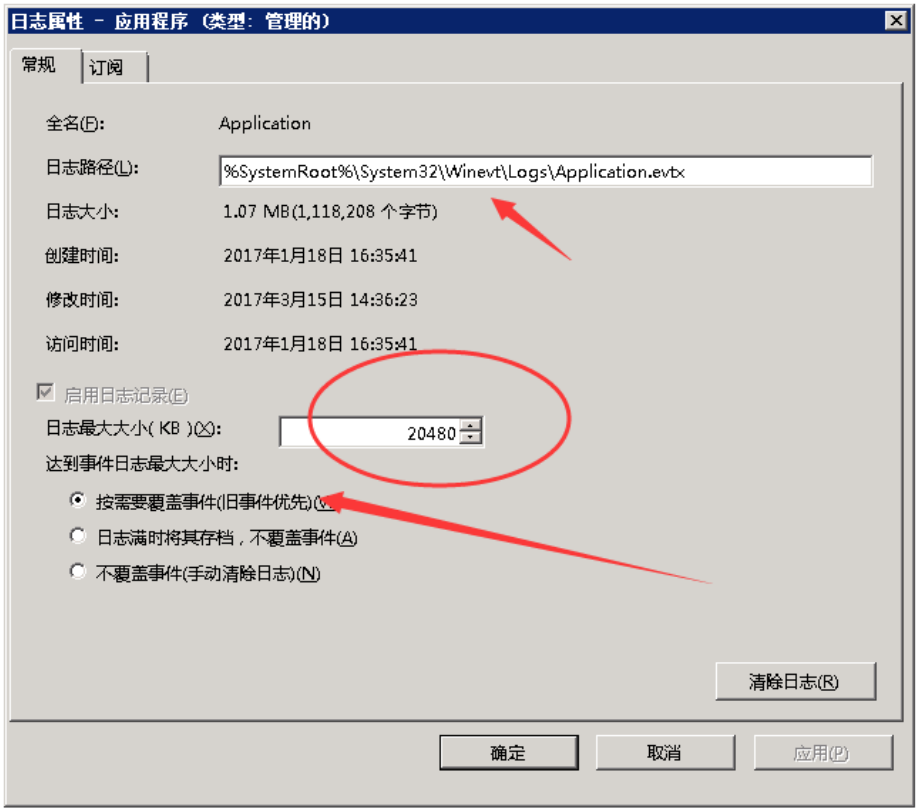
日志路径的修改和备份

日志默认保存在系统盘里面，日志最大值默认是20M,超过20M时会覆盖之前的事件，可以根据自己的需求修改

。



右键选择属性



相关链接

云服务器 ECS Windows 安全审计日志简要说明

简介

在Windows NT6.0之后微软推出了高级安全Windows防火墙(简称WFAS)，高级安全Windows防火墙是分层安全模型的重要部分， 通过为计算机提供基于主机的双向网络通讯筛选， 高级安全Windows防火墙 阻止未授权的网络流量流向或流出本地计算机。高级安全 Windows 防火墙 还是用网络感知，以便可以将相应安全设置应用到计算机连接到的网络类型。Windows 防火墙和 Internet 协议保护 (sec) 配置设置集成到名为 高级安全 Windows 防火墙 的单个 Microsoft 管理控制台 (MMC)，高级安全Windows防火墙也成为网络隔离策略的重要部分。

适用场景

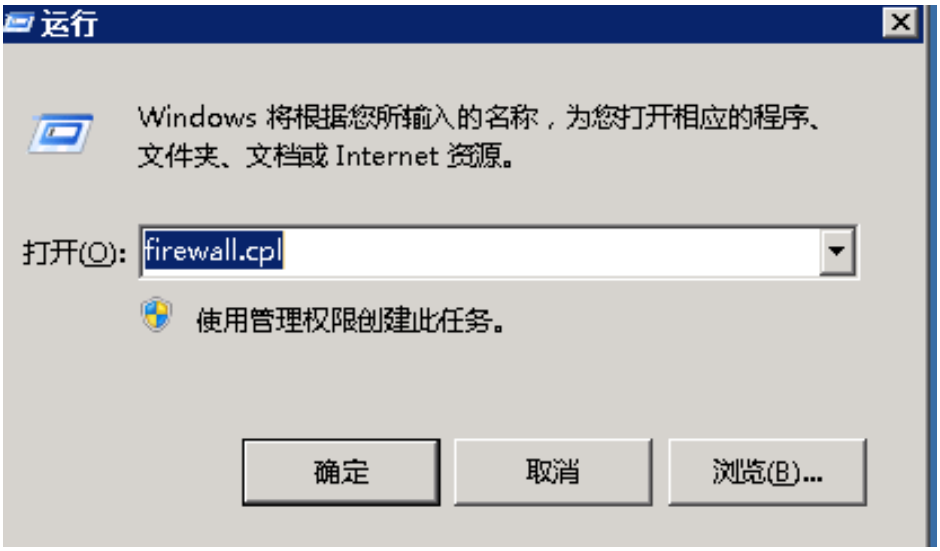
作为一个运维人员，越来越多的用户反映服务器被恶意攻击，密码被暴力破解等等，其实大多数原因都是自己给那些“入侵者”留的“后门”导致的。入侵者通过扫描主机开放的端口，一旦发现可以利用的端口，就会进行下一步的入侵，例如Windows的远程端口（3389）和Linux的远程端口（22）。既然知道了问题的关键，那么我们也有相应的对策，我们可以通过修改默认的远程端口以及限制远程的访问来关闭所谓的“后门”。那么

如何限制远程访问呢？接下来我们就以阿里云ECS实例Windows Server 2008 R2为例，来实现对远程桌面的限制。

操作步骤

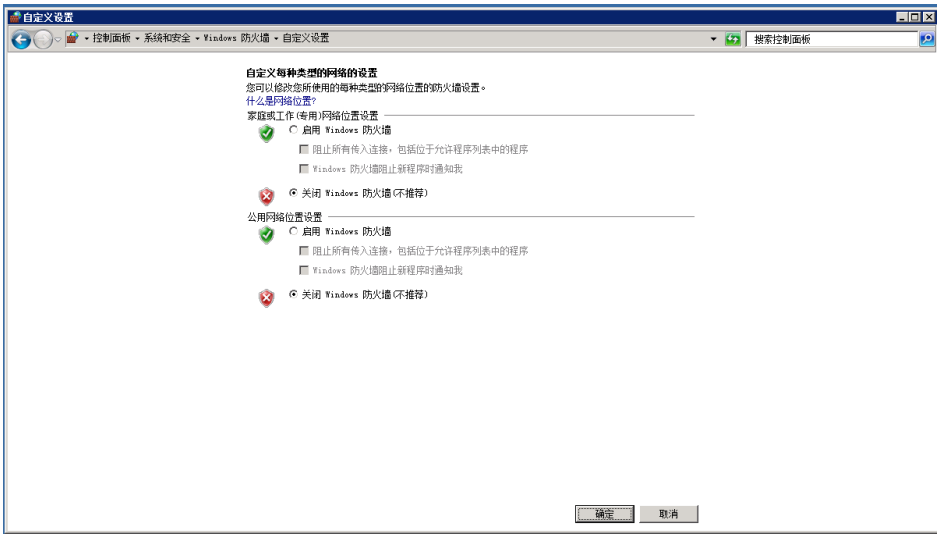
1.查看防火墙状态

阿里云ECS实例Windows Server 2008 R2防火墙默认是关闭的，键盘输入Win+R打开【运行】输入“firewall.cpl” 回车来打开Windows防火墙控制台，见下图。



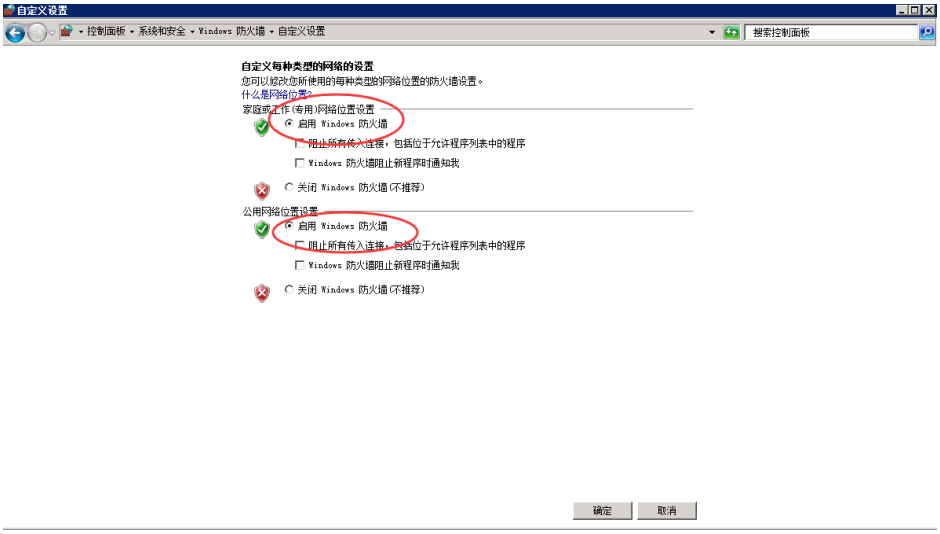
选择打开或关闭Windows防火墙。

如下图，我们看到防火墙是默认关闭的。



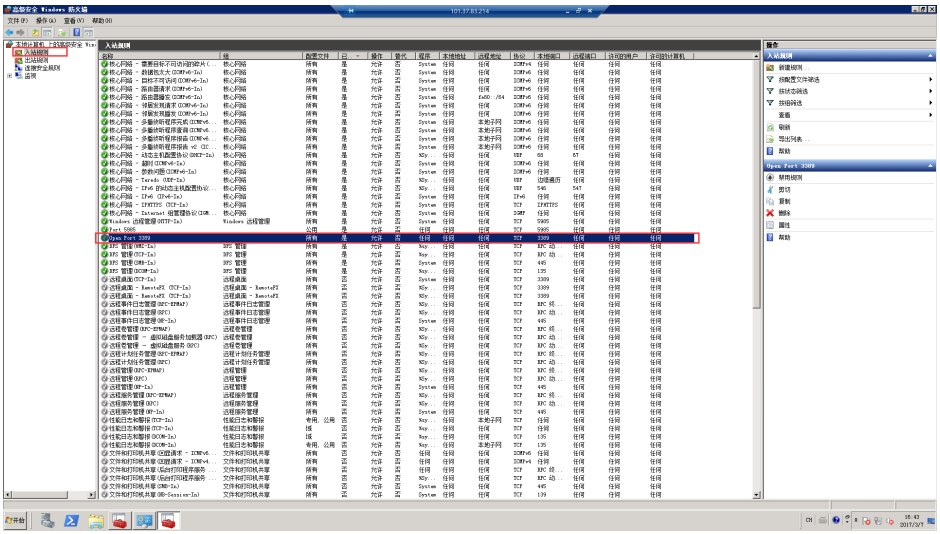
2.启用防火墙

还是通过上面的步骤开启防火墙，见下图。



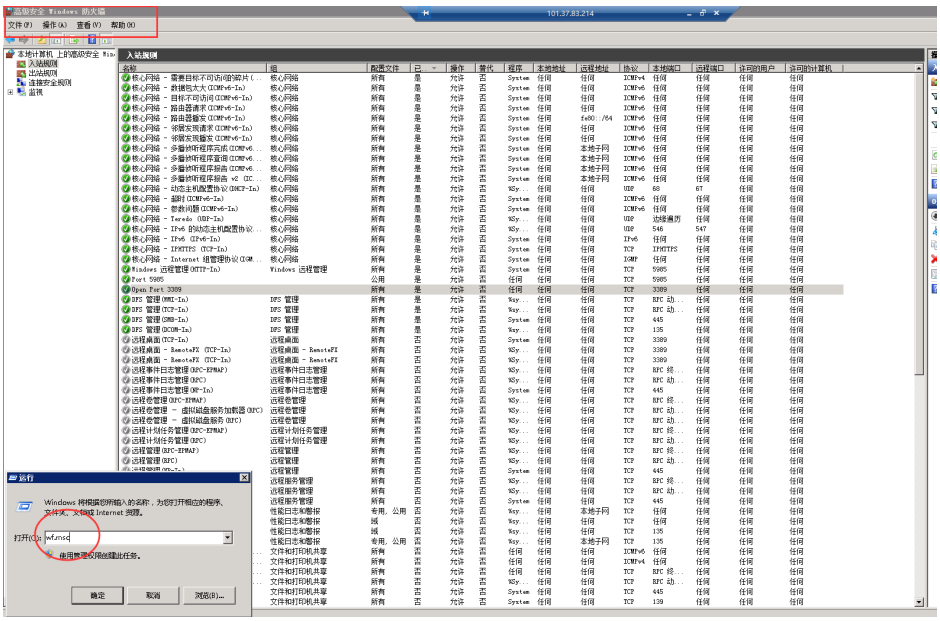
这里需要注意一点的是：启用之前请确认远程端口已经在里面，否则自己也将无法远程，不过高级安全Windows防护墙入站规则默认是放行3389端口的选择高级设置。

选择入站规则，我们看到open port 3389这条入站规则默认是放行3389端口的。

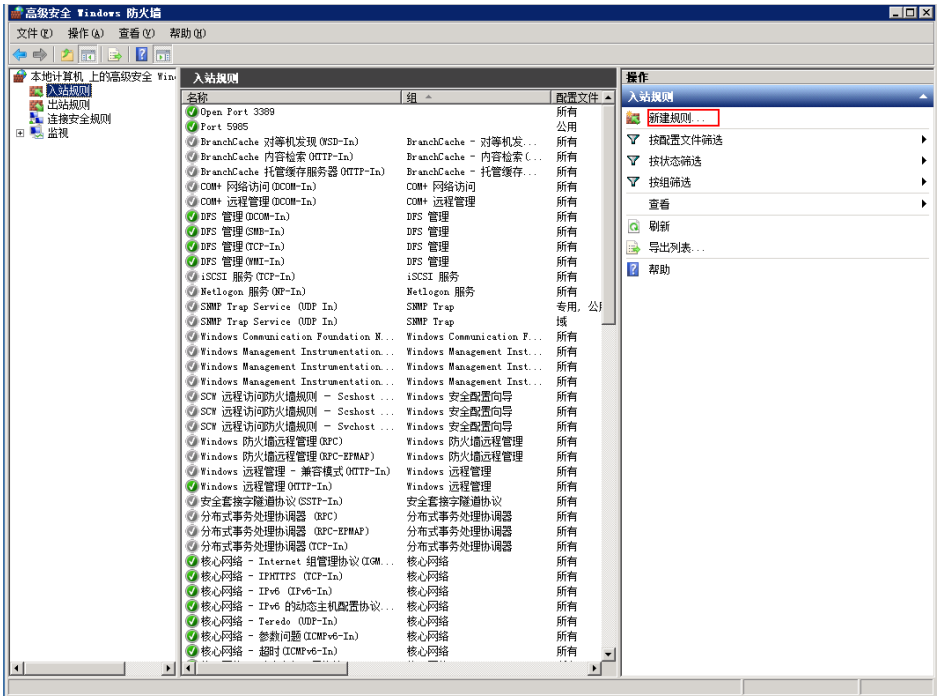


3.配置高级安全Windows防火墙

键盘输入Win+R打开【运行】输入“wf.msc” 回车来打开高级安全Windows防火墙，如下图。

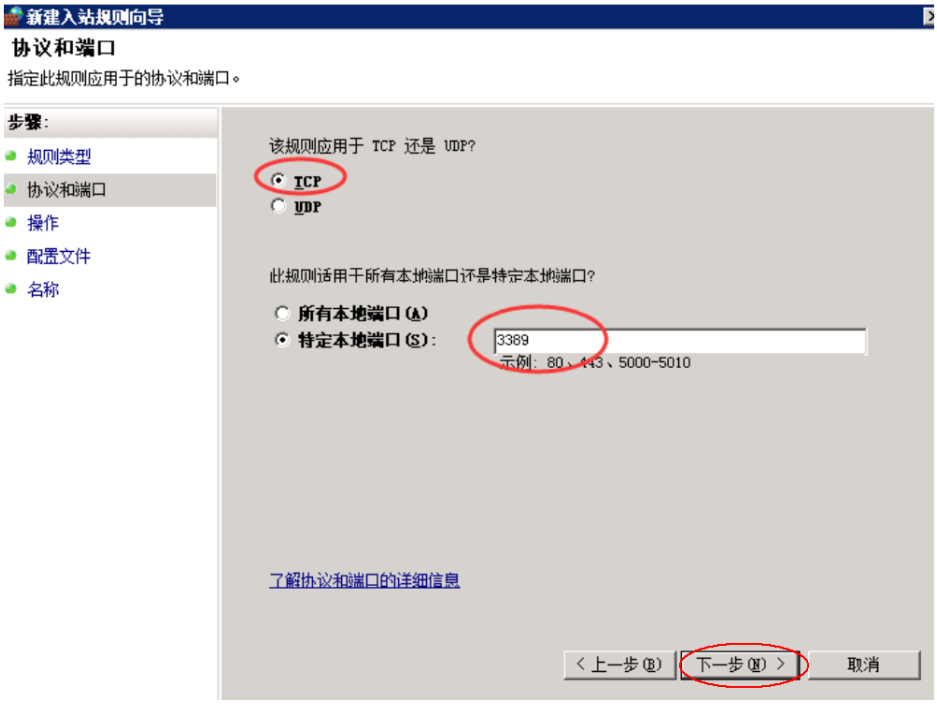


（1）通过手工新建入站规则

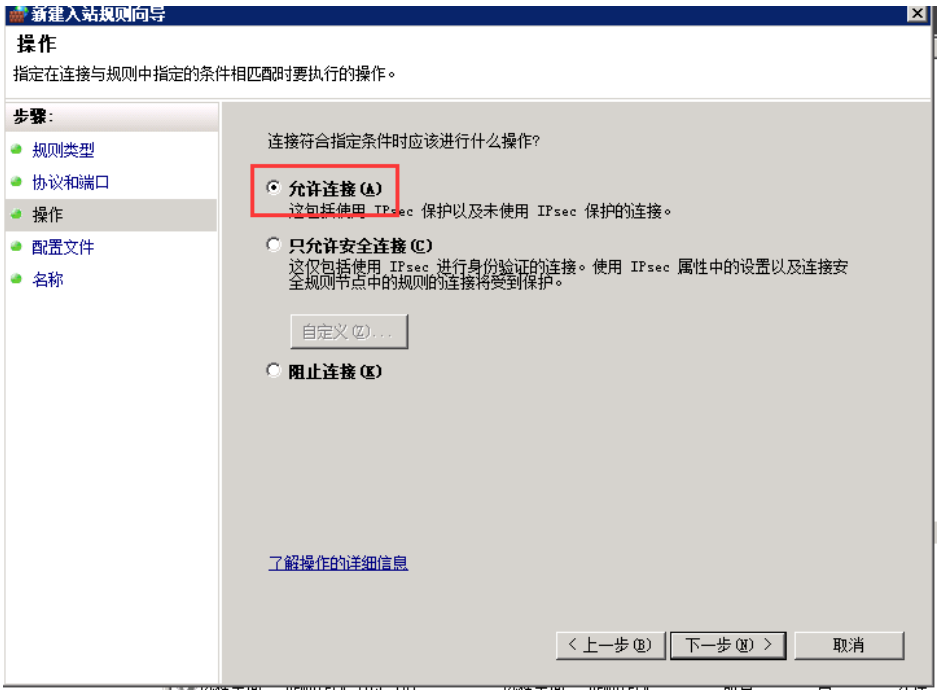


在弹出的新建入站规则向导窗口，选择 端口 然后鼠标左键单击下一步。

而后选择 TCP 并设置特定本地端口3389。



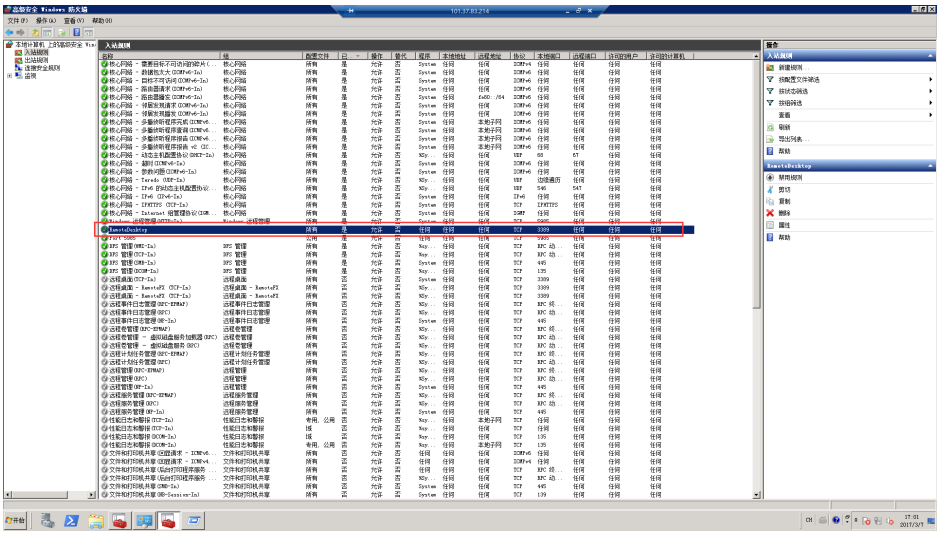
下一步选择允许链接。



下一步 默认配置即可。

下一步 填写规则名称，例如 RemoteDesktop ,最后鼠标左键单击完成。

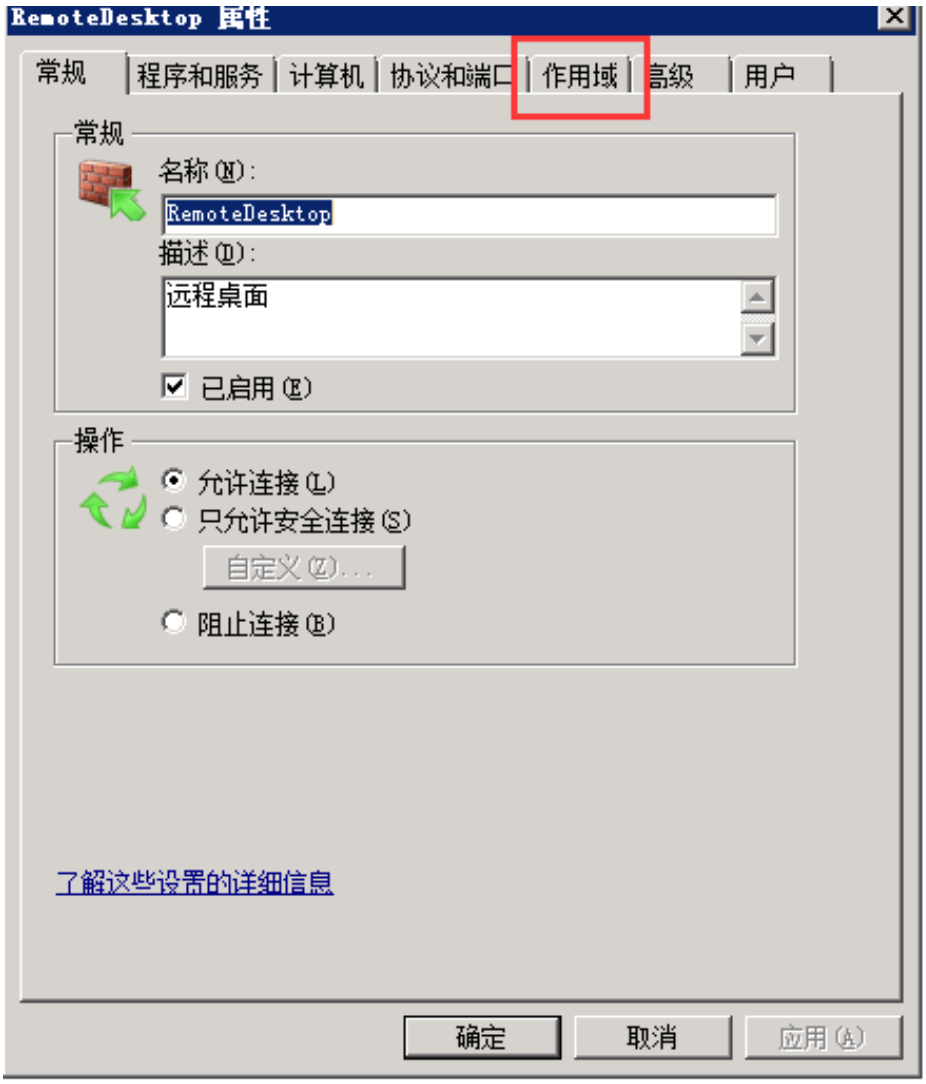
看到我们刚刚添加的规则。



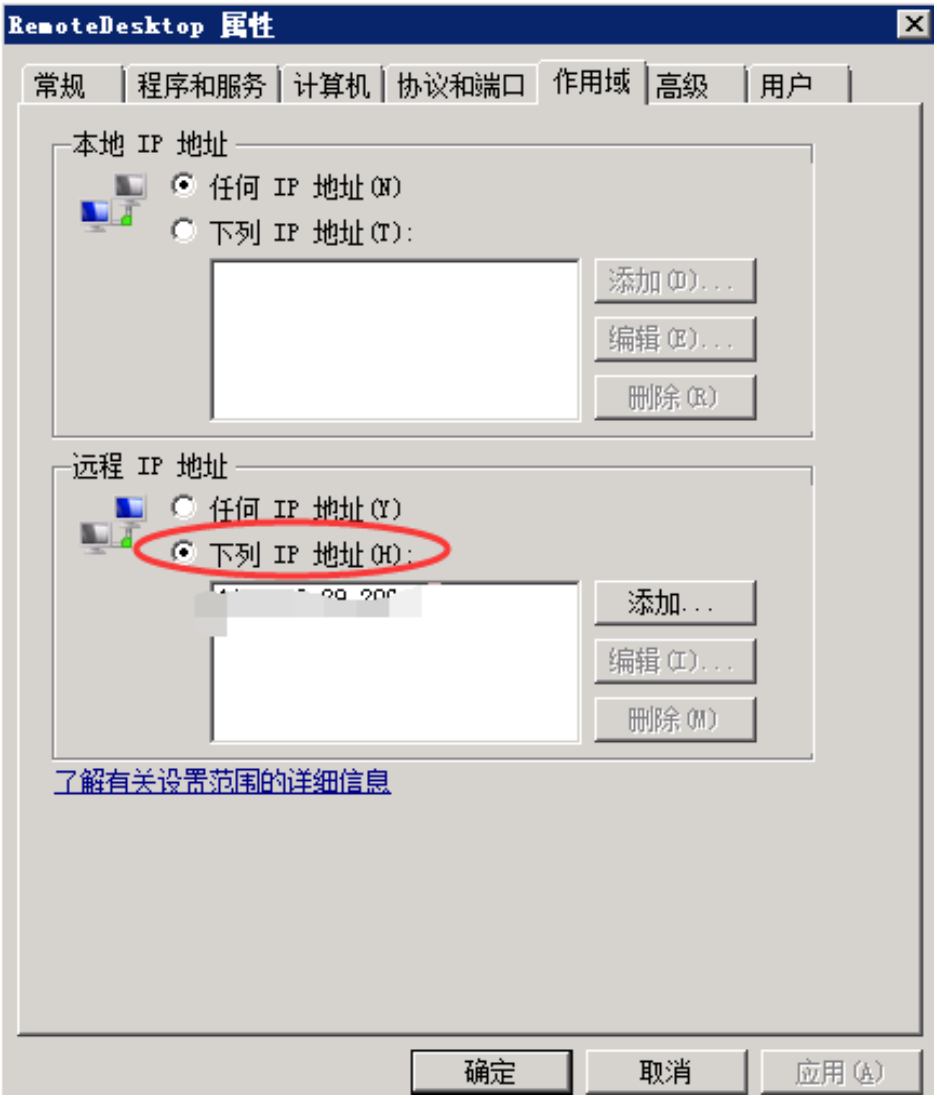
以上步骤就是把Windows远程端口加入到高级安全Windows防火墙了，但是依然没有实现我们的限制访问，接下来我们来实现访问限制

（2）配置作用域

右键选中我们刚刚创建的入站规则，然后选择属性>作用域>远程IP地址>添加（将需要远程此服务器的IP地址填写进去，注意：一旦启用作用域，除了作用域里面的IP地址，别的地址将无法远程链接此服务器）。

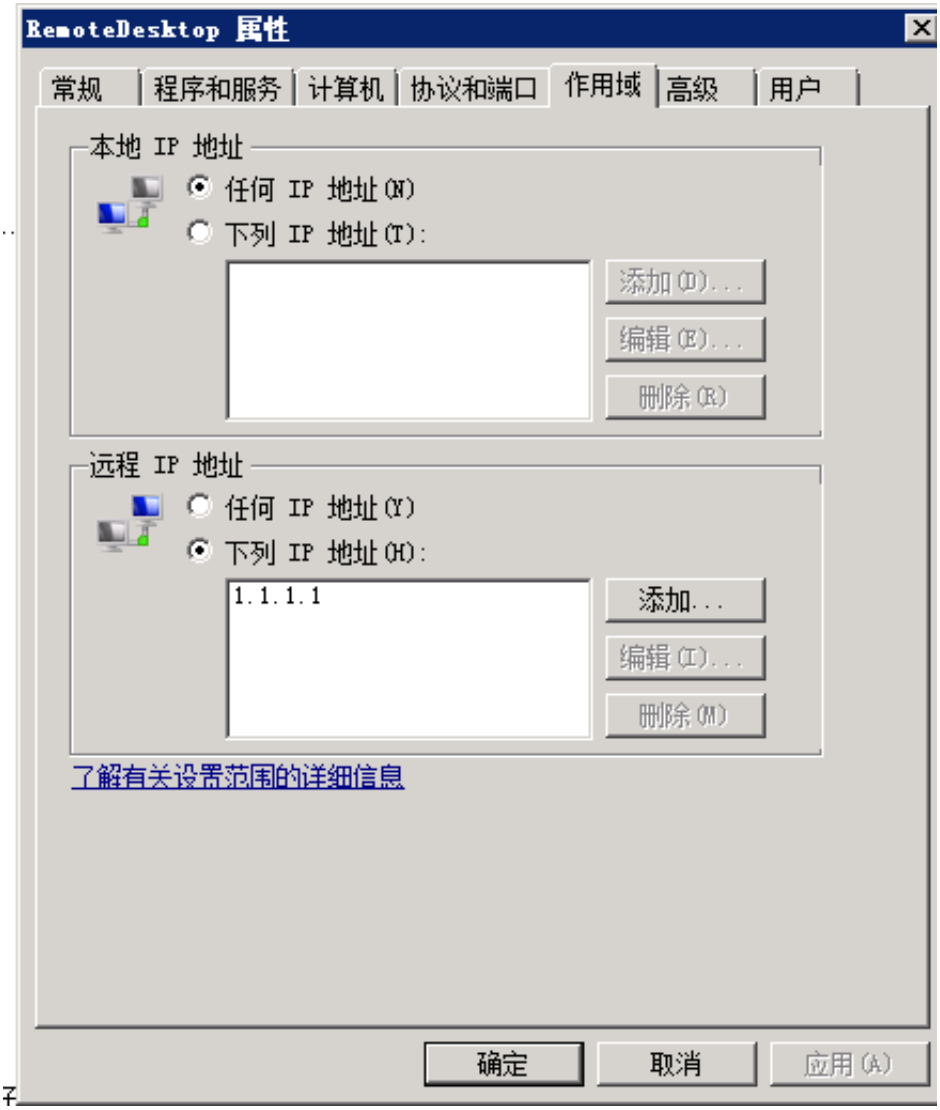

RemoteDesktop 属性
常规
程序和服务
计算机
协议和端口
作用域
高级
用户
常规
名称(N):
RemoteDesktop
描述(D):
远程桌面
已启用(E)
操作
允许连接(L)
只允许安全连接(S)
自定义(Z)...
阻止连接(B)
了解这些设置的详细信息
确定
取消
应用(A)

添加远程IP地址。

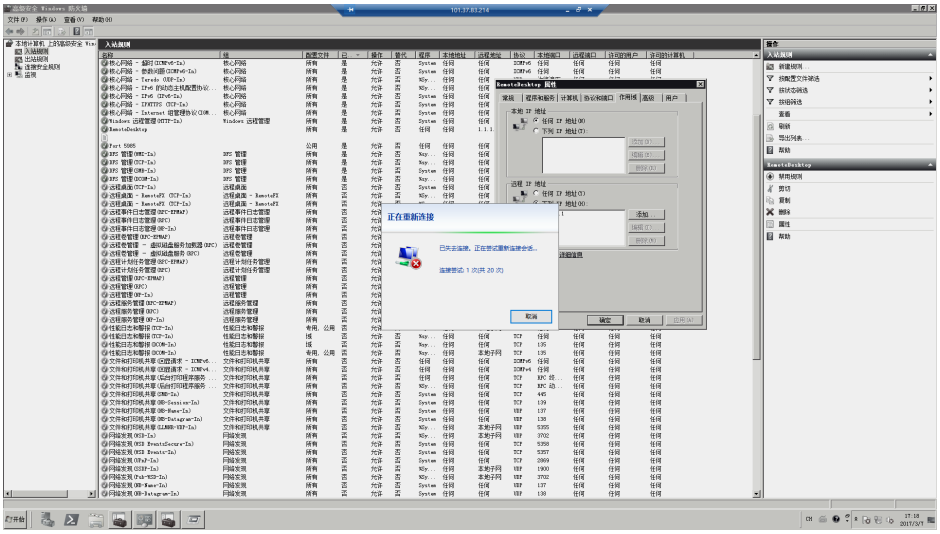


（3）验证作用域

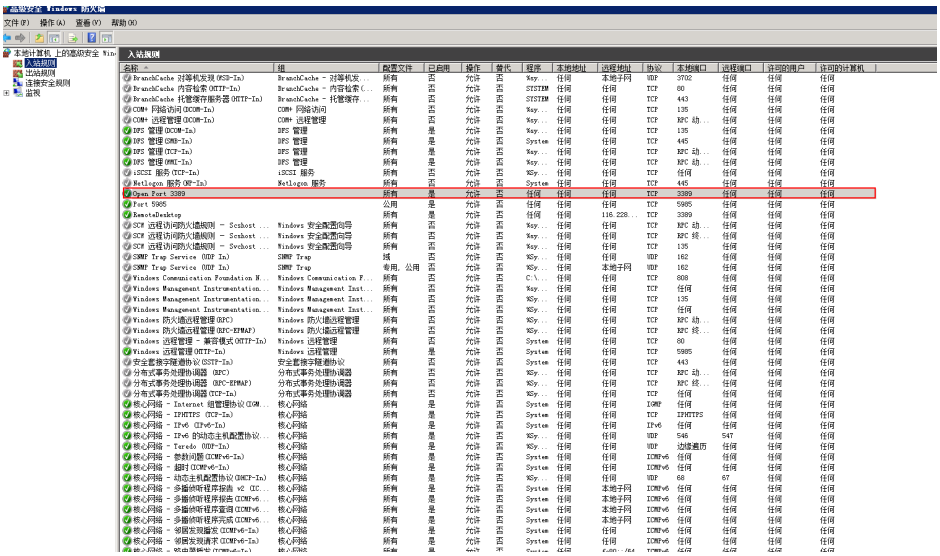
我们在作用域——远程IP地址里面随便写个地址，看看远程连接会发生什么。



远程连接断掉。



如果远程连接没有断开，让我们把下图中open port 3389这条入站规则禁用掉就可以了。

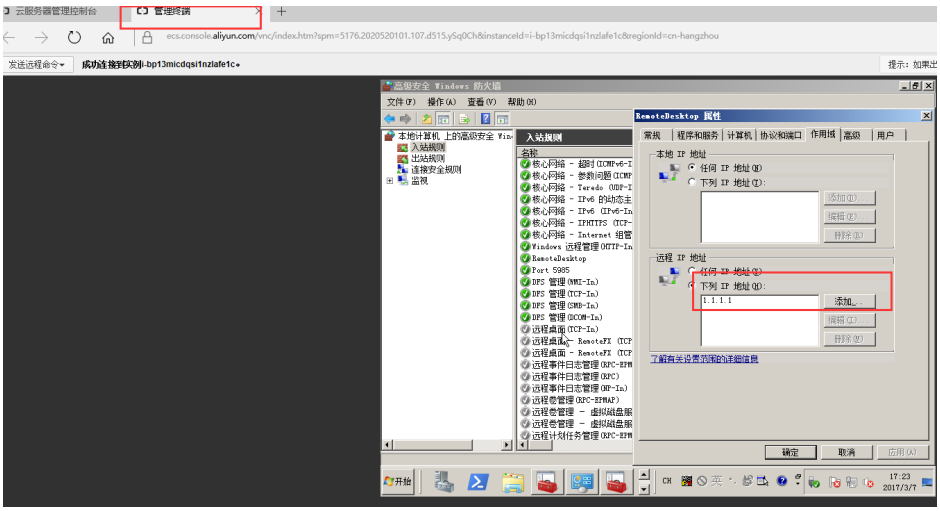


远程连接自己断开了，这就说明我们的作用域生效了，那现在自己都无法远程了，怎么办呢？别急，我们还有阿里云控制台，登录阿里云控制台，然后将上面的作用域地址换成自己的地址（这里要写办公环境的公网地址，除非您的办公环境和阿里云线上的环境打通，）就可以正常远程了。

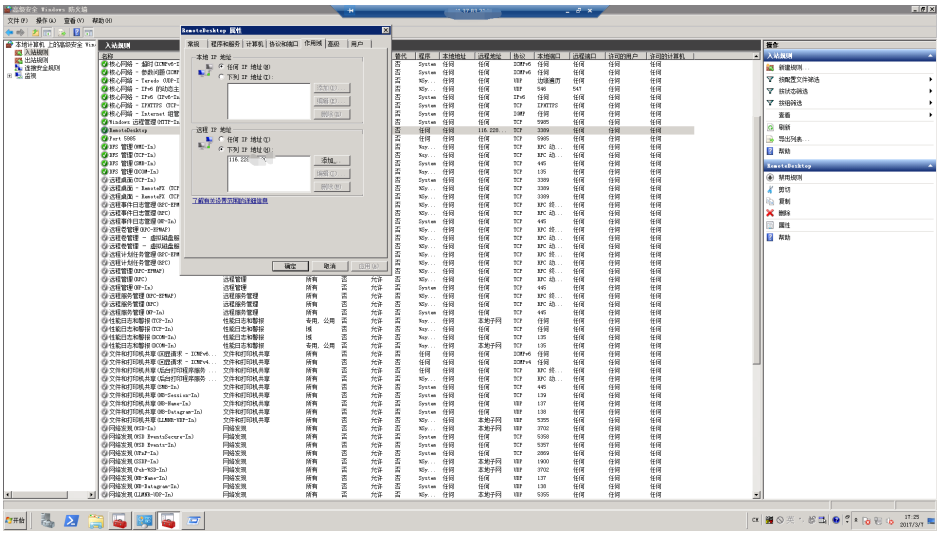
进入阿里云的控制台界面，找到相应实例打开远程连接。

登录系统。

与之前同样的方式，修改RemoteDesktop的作用域的远程IP地址，将之前测试设置的1.1.1.1换回自己的IP地址。



换回自己的IP地址后可以正常远程了，如果不知道自己的公网IP，可以点击此处查看



以上就是使用高级安全Windows防火墙来实现对服务器远程访问的限制，其他的服务和端口都可以按照上面的方法来实现，例如，关闭不常用的135 137 138 445 端口，限制FTP和相关服务的访问等等，这样才能做到最大限度地保障服务器安全的运行。

命令行的方式

1.导出防火墙配置到文件

```
netsh advfirewall export c:\adv.pol
```

2.导入防火墙配置文件到系统中

```
netsh advfirewall import c:\adv.pol
```

3.防火墙恢复默认设置

```
Netsh advfirewall reset
```

4.关闭防火墙

```
netsh advfirewall set allprofiles state off
```

5.开启防火墙

netsh advfirewall set allprofiles state on

6.在所有配置文件中设置默认阻挡入站并允许出站通信

netsh advfirewall set allprofiles firewallpolicy blockinbound,allowoutbound

7.删除名为 ftp 的规则

```
netsh advfirewall firewall delete rule name=ftp
```

8.删除本地端口 80 的所有入则

```
netsh advfirewall firewall delete rule name=all protocol=tcp localport=80
```

9.添加远程桌面入站规则允许端口3389

```
netsh advfirewall firewall add rule name=远程桌面(TCP-In-3389) protocol=TCP dir=in localport=3389
action=allow
```

相关链接

用户可通过云中沙箱平台体验上述文档中的操作，点击此处。

Windows防火墙限制端口/IP/应用访问的方法以及例外的配置

Windows 系统远程桌面端口查看和修改方法

Linux 修改默认远程端口方法

更多开源软件尽在云市场

您可以在实例上安装 Web 服务器，使实例对外提供 Web 服务。目前主流的 Web 服务器包括 nginx、Apache HTTP Server、IIS、Apache Tomcat 等。本文以 nginx 为例，说明如何在阿里云的 ECS 实例上安装 Web 服务器，并使其对外提供 Web 服务。

前提条件

您应该已经 创建了实例，并已经能正常 远程登录实例。

操作步骤

操作步骤如以下流程图所示。

根据实例的操作系统，您需要选择不同的操作：

- Linux 实例
- Windows 实例

Linux 实例

在这一部分，示例中使用的 Linux 实例上运行的镜像为 CentOS 6.8 64位。您应该按以下步骤在 Linux 实例上安装并运行 nginx 服务器：

根据 Linux 实例的网络类型，在实例所在安全组中添加如下安全组规则：

网络类型	网卡类型	规则方向	授权策略	协议类型	端口范围	授权类型	授权对象	优先级
VPC 网络	不需要配置	入方向	允许	HTTP (80)	80/80	地址段访问	0.0.0.0/0	1
经典网络	公网							

如果您需要使用其他端口，请参考 这里。

远程登录 Linux 实例。

运行命令 yum install nginx，安装 nginx。

运行命令 service nginx start，启动 nginx。

如果报错：

```
Starting nginx: nginx: [emerg] socket() [::]:80 failed (97: Address family not supported by protocol)
[FAILED]
```

表示不支持 IPv6 地址。您需要通过 vi /etc/nginx/conf.d/default.conf 将文件中的 server 监听端口部分做如下修改：

```
server {
listen 80 default_server;
#listen [::]:80 default_server;
```

如果是 CentOS 7 以上的系统，运行命令 systemctl start nginx 启动 nginx。

运行命令 netstat -an | grep 80，查看 TCP 80 是否被监听。
如果返回以下结果，说明 TCP 80 端口的 Web 服务启动。

```
tcp        0      0 0.0.0.0:80              0.0.0.0:*               LISTEN
```

在本地机器的浏览器中输入实例的公网 IP 地址，如果出现以下页面，说明您已经在 ECS 实例上正确安装了 nginx 服务器。

修改 Linux 实例的 HTTP 访问端口（本示例中改为端口 81）：

1. 在 ECS 控制台上，根据 Linux 实例的网络类型，在实例所在安全组中添加如下安全组规则：

网络类型	网卡类型	规则方向	授权策略	协议类型	端口范围	授权类型	授权对象	优先级
VPC 网络	不需要配置	入方向	允许	自定义 TCP	81/81	地址段访问	0.0.0.0/0	1
经典网络	公网							

2. 登录实例，通过 vi /etc/nginx/conf.d/default.conf 将文件中的 server 监听端口部分做如下修改：

```
server {
listen 81 default_server;
#listen [::]:80 default_server;
```

3. 保存并退出编辑。
4. 重新启动 nginx。
5. 在本地机器的浏览器中输入实例的 公网 IP 地址:81。

Windows 实例

在这一部分，示例中使用的 Windows 实例上运行的镜像为 Windows Server 2012 R2 64 位。

您应该按以下步骤在 Windows 实例上安装并运行 nginx 服务器：

根据 Windows 实例的网络类型，在实例所在安全组中添加如下安全组规则：

网络类型	网卡类型	规则方向	授权策略	协议类型	端口范围	授权类型	授权对象	优先级
VPC 网络	不需要配置	入方向	允许	HTTP (80)	80/80	地址段访问	0.0.0.0/0	1
经典网络	公网							

如果您需要使用其他端口，请参考 这里。

远程登录 Windows 实例。

从 http://nginx.org/en/download.html 上下载需要的 nginx 压缩文件。在本示例中，选择下载 nginx/Windows-1.13.4。

右击压缩文件，选择 **全部提取** 到任意路径下。本示例中路径为 C:\nginx-1.13.4。

启动命令提示符，运行以下命令：

```
cd C:\nginx-1.13.4\nginx-1.13.4 #转到 C:\nginx-1.13.4\nginx-1.13.4
start nginx #启动 nginx
```

运行命令 netstat -aon | findstr :80，查看 TCP 80 是否被监听。
如果返回以下结果，说明 TCP 80 端口的 Web 服务启动。

```
TCP    0.0.0.0:80         0.0.0.0:0          LISTENING       1172
```

在浏览器中输入实例的公网 IP 地址，如果出现以下页面，说明您已经在 ECS 实例上正确安装了 nginx 服务器。

按以下步骤修改 Windows 实例的 HTTP 访问端口（本示例中改为端口 81）：

1. 在 ECS 控制台上，根据 Windows 实例的网络类型，在实例所在安全组中添加如下安全组规则：

网络类型	网卡类型	规则方向	授权策略	协议类型	端口范围	授权类型	授权对象	优先级
VPC 网络	不需要配置	入方向	允许	自定义 TCP	81/81	地址段访问	0.0.0.0/0	1
经典网络	公网							

2. 在 C:\nginx-1.13.4\nginx-1.13.4\conf 目录下，打开 nginx.conf 文件，在以下内容里，将端口号修改为您需要的值，比如本例中将 80 改为 81。

```
 server {
listen 81;
server_name localhost;
```

3. 重新启动 nginx。
4. 在本地机器的浏览器中输入 实例的公网 IP 地址:81。

数据恢复

简介

在日常使用中有时难免会出现数据被误删除的情况，在这个时候该如何快速、有效地恢复数据呢？在阿里云上恢复数据有多种方式，例如：

通过阿里云控制台回滚备份好的快照，自定义镜像恢复等方式。

购买多台ECS，实现业务的负载均衡，高可用。

利用对象存储 OSS（Object Storage Service），存储静态网页和海量图片、视频等重要数据。

本文档主要以CentOS7操作系统为例，介绍如何使用开源工具Extundelete快速恢复被误删除掉的数据。

在Linux下，基于开源的数据恢复工具有很多，常见的有debugfs、R-Linux、ext3grep、extundelete等，比较常用的有ext3grep和extundelete，这两个工具的恢复原理基本一样，只是extundelete功能更加强大。

Extundelete是基于linux的开源数据恢复软件。在使用阿里云的云服务器时，如果您不小心误删除数据，并且Linux系统也没有与Windows系统下回收站类似的功能，您可以方便快速安装此工具。

Extundelete能够利用inode信息结合日志去查询该inode所在的block位置，以次来查找和恢复所需的数据，该工具最给力的一点就是支持ext3/ext4双格式分区恢复，基于整个磁盘的恢复功能较为强大。

注意事项

在数据被误删除后，第一时间要做的是卸载被删除数据所在的磁盘或磁盘分区。因为将文件删除后，仅仅是将文件的inode结点中的扇区指针清零，实际文件还存储在磁盘上，如果磁盘以读写模式挂载，这些已删除的文件的数据块就可能被操作系统重新分配出去，在这些数据块被新的数据覆盖后，这些数据就真的丢失了，恢复工具也回力无天。所以，以只读模式挂载磁盘可以尽量降低数据块中数据被覆盖的风险，以提高恢复数据成功的几率。

注：在实际线上恢复过程中，切勿将extundelete安装到您误删的文件所在硬盘，这样会有一定几率将需要恢复的数据彻底覆盖，切记操作前做好快照备份。

适用对象

磁盘中文件误删除的用户，且未对磁盘进行过写入等操作

网站访问量小、少量 ECS 实例的用户

使用方法

需安装的软件及版本：e2fsprogs-devel e2fsprogs gcc-c++ make（编译器等）Extundelete-0.2.4

注：extundelete需要libext2fs版本1.39或更高版本来运行，但是对于ext4支持，请确保您有e2fsprogs版本1.41或更新版本（可以通过运行命令“dumpe2fs”并记录其输出的版本）

说明：以上版本是写文档时的软件版本。您下载的版本可能与此不同。

部署extundelete工具

```
wget  http://zy-res.oss-cn-hangzhou.aliyuncs.com/server/extundelete-0.2.4.tar.bz2
yum -y install bzip2 e2fsprogs-devel e2fsprogs gcc-c++ make #安装相关依赖和库
tar -xvjf extundelete-0.2.4.tar.bz2
cd extundelete-0.2.4 #进入程序目录
./configure #如下图表示安装成功
```

```
extundelete-0.2.4/src/Makefile.am
extundelete-0.2.4/configure.ac
extundelete-0.2.4/depcomp
extundelete-0.2.4/Makefile.in
extundelete-0.2.4/Makefile.am
[root@iZy930wmhyutc2Z ~]# cd extundelete-0.2.4
[root@iZy930wmhyutc2Z extundelete-0.2.4]# ./configure
Configuring extundelete 0.2.4
Writing generated files to disk
[root@iZy930wmhyutc2Z extundelete-0.2.4]#
```

```
make && make install
```

这个时候会出现src目录，下面有个extundelete可执行文件以及相应路径，如下图，其实默认文件安装在usr/local/bin下面，下面演示就在usr/local/bin目录下。

```
[root@iZy930wmhyutc2Z extundelete-0.2.4]# ls
acinclude.m4  config.h     config.status  depcomp     Makefile     missing  stamp-h1
aclocal.m4    config.h.in  configure      install-sh  Makefile.am  README
autogen.sh    config.log   configure.ac   LICENSE     Makefile.in  src
```

使用extundelete，模拟数据误删除然后恢复的过程

1.检查ECS现有的磁盘和可用分区，并对/dev/vdb进行分区，格式化，此处不在介绍磁盘分区格式化方式，如果不会的话可以点击此文档查看操作方式“**格式化和挂载数据盘**”。

```
fdisk -l
```

```
Disk identifier: 0x0000efd2

   Device Boot      Start         End      Blocks   Id  System
/dev/vda1   *        2048    83886079    41942016   83  Linux

Disk /dev/vdb: 21.5 GB, 21474836480 bytes, 41943040 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
```

2.将分区后的磁盘挂载到/zhuyun目录下，然后在/zhuyun下面新建测试文件hello,写入test。

```
mkdir /zhuyun                          #新建zhuyun目录
mount /dev/vdb1 /zhuyun #将磁盘挂载到zhuyun目录下
echo test > hello #写入测试文件
```

3.记录文件MD5值，md5sum命令用于生成和校验删除前和恢复后俩个文件的md5值。

```
md5sum hello
```

```
[root@iZbp13micdqsi2364umm8aZ zhuyun]# md5sum hello
d8e8fca2dc0f896fd7cb4cb0031ba249  hello
```

4.模拟删除hello文件。

```
rm -rf hello
cd ~
fuser -k /zhuyun #结束使用某分区的进程树（确认没有资源占用的话，可以跳过此步）
```

5.卸载数据盘。

```
umount /dev/vdb1                #任何的文件恢复工具，在使用前，均要将要恢复的分区卸载或挂载为只读，防止数据被覆盖使用
```

6.使用Extundelete工具恢复文件。

```
extundelete --inode 2 /dev/vdb1      #为查找某i节点中的内容，使用2则说明为整个分区搜索，如果需要进入目录搜索，只须要指定目录I节点即可。这是可以看到删除的文件名和inode
```

```
Direct blocks: 127754, 4, 0, 0, 1, 9252, 0, 0, 0, 0, 0, 0
Indirect block: 0
Double indirect block: 0
Triple indirect block: 0

File name                                       | Inode number | Deleted status
.                                                 2
..                                                2
lost+found                                        11
hello                                             12             Deleted
```

```
/usr/local/bin/extundelete  --restore-inode 12  /dev/vdb1    #恢复删除的文件
```

这个时候会在执行命令的同级目录下出现RECOVERED_FILES目录，查看是否恢复。

```
[root@iZbp13micdqsi2364umm8aZ /]# ll RECOVERED_FILES/
total 4
-rw-r--r-- 1 root root 5 Mar  8 14:20 hello
```

通过md5值查看，前后俩个文件，一样说明恢复成功。

注：

```
--restore-inode 12                # --restore-inode 按指定的I节点恢复
--extundelete --restore-all # --restore-all 全部恢复
```

相关链接

用户可通过云中沙箱平台体验上述文档中的操作，**点击此处。**

磁盘空间满的问题处理(Windows /Linux)及最佳实践

本文主要介绍window、Linux系统磁盘空间不足时对应的处理方法。

适用对象

适用于使用阿里云ECS的用户。

主要内容

- 云服务器 ECS Linux磁盘空间满排查处理
- 云服务器 ECS window磁盘空间满排查处理

ECS Linux磁盘空间满排查处理

Windows磁盘空间满排查处理

解决Windows磁盘空间满的问题，有以下处理方式：

- 释放磁盘空间

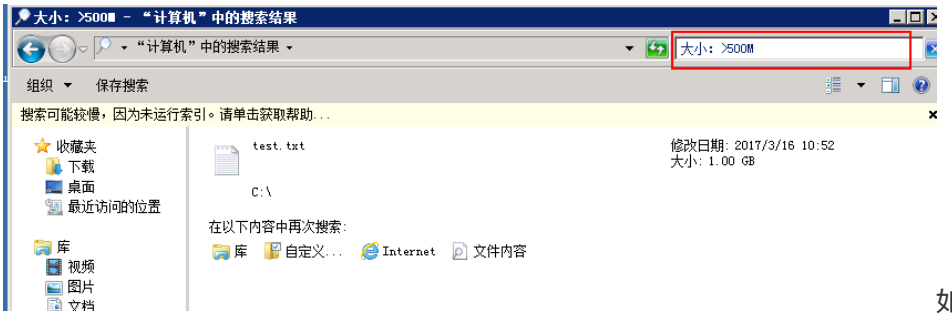
- 扩充磁盘容量
- 文件压缩保存
- 设置磁盘监控

释放磁盘空间

首先找出占用了磁盘空间过多的文件，如果文件没有用，可以及时清理。具体的操作可参考以下步骤：

下图以Windows2008R264位操作系统为例，打开“计算机”，用鼠标左键单击要清理的磁盘，按下键盘的ctrl+f键，定位到搜索框，可以根据系统定义大小筛选指定磁盘的大文件。

也可以自定义文件大小范围进行检索展示，如输入“大小：>500M”,会检索该磁盘大于500M的文件。

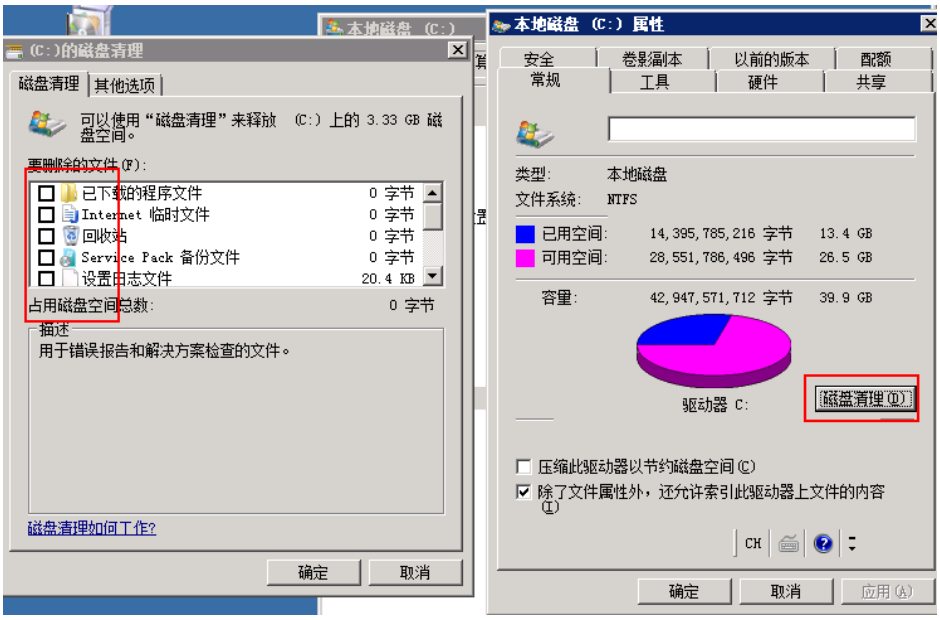


如输入“大小：>100M <500M”，会检索大于100M但小于500M的文件。

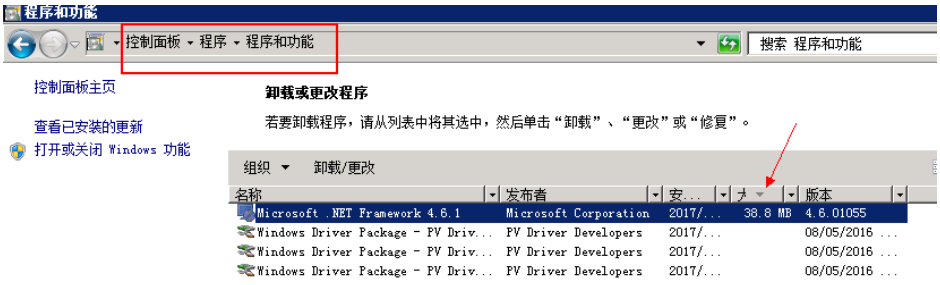
推荐使用系统自带的磁盘清理工具，删除日志文件及系统上其他不需要文件，并清空回收站。磁盘清理工具服务器默认没有安装，需要手动安装，具体安装步骤如下：

1. 打开“服务器管理器”——在“功能摘要”下，单击“添加功能”。
2. 在“选择功能”页上，选中“桌面体验”复选框，然后单击“下一步”。
3. 在“确认安装选项”页上，验证是否将安装桌面体验功能，然后单击“安装”。
4. 在“安装结果”页上，系统将提示您重新启动服务器以完成安装过程。单击“关闭”，然后单击“是”重新启动服务器。重新启动服务器之后，确认已安装了桌面体验。
5. 启动Server Manager，在“功能摘要”下，确认桌面体验列为已安装。

安装完成后单击“开始”—>“所有程序”—>“附件”—>“系统工具”—>“磁盘清理”，打开磁盘清理工具选择要清理的选项。



此外，服务器环境建议尽量保持简洁，定期清理不必要的应用程序，可以通过控制面板中的程序和功能窗口清理不再使用的程序软件。下图以Windows2008R264位操作系统示例：



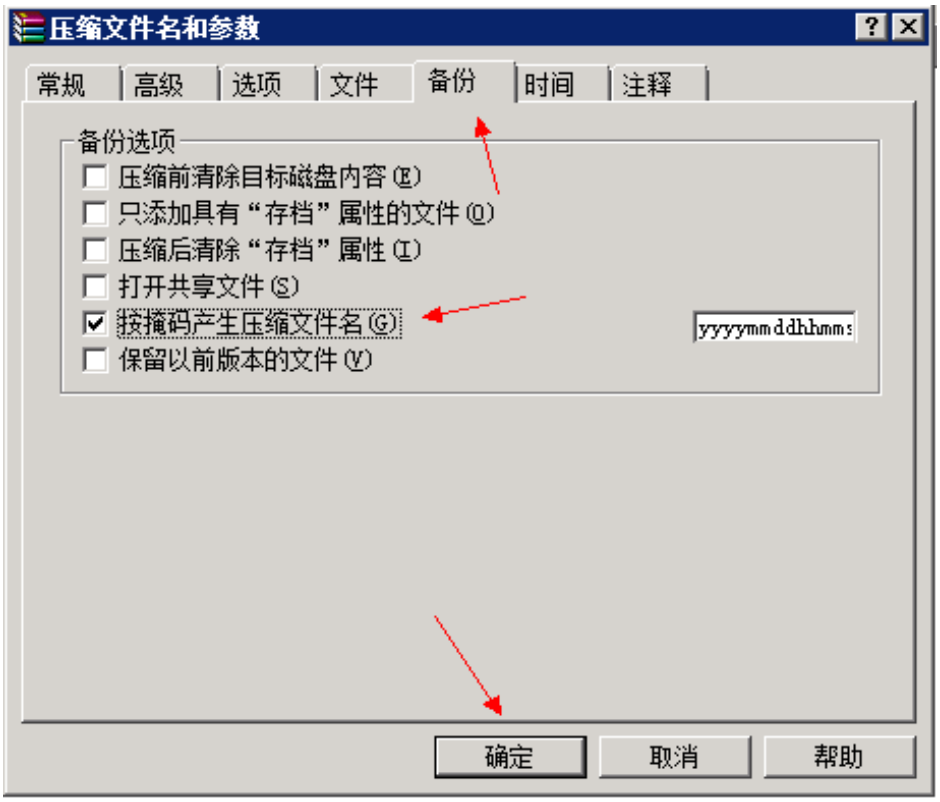
磁盘扩容

磁盘扩容有多种场景，您可以根据实际情况选择扩容windows系统盘，或者扩容windows数据盘。

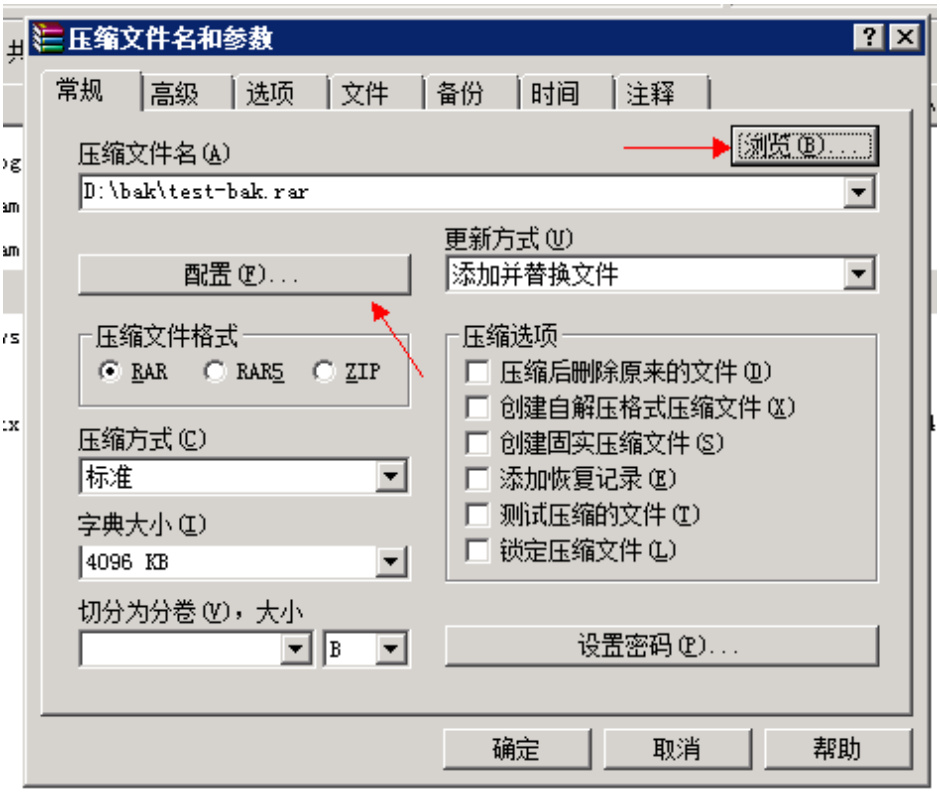
文件压缩保存

清理完不需要文件，服务器日常运维需要养成良好磁盘使用习惯。对于一些定期生成的文件可以进行归档压缩后保存，以提高磁盘使用率。

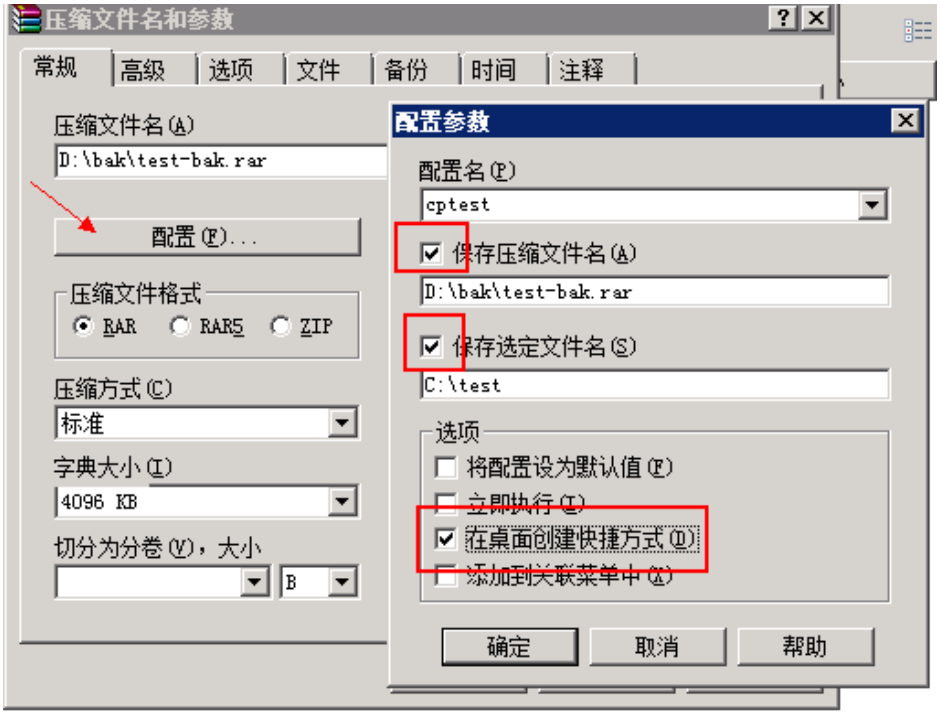
推荐使用winrar压缩工具，配置压缩策略过程如下：安装好软件后找到需要压缩备份的目录，右键选择添加到压缩文件，在设置界面单击备份选项，然后勾选按掩码产生文件名，注意此时不要单点确定。



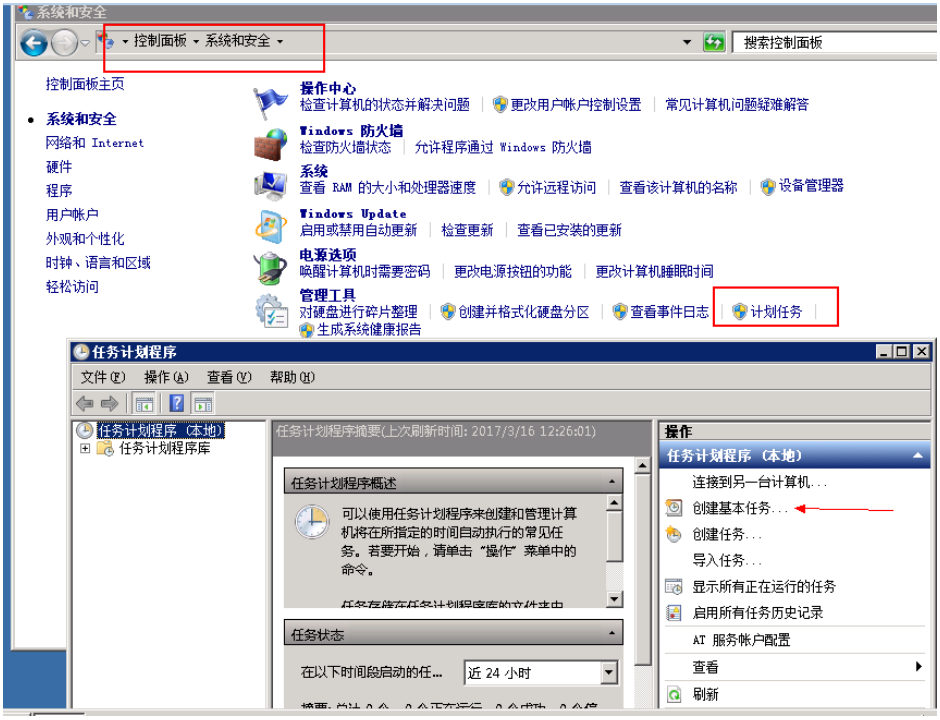
单击常规选项，单击浏览定义压缩备份的路径和修改文件名。



单击配置选项，选择保存当前配置为新配置进行设置。

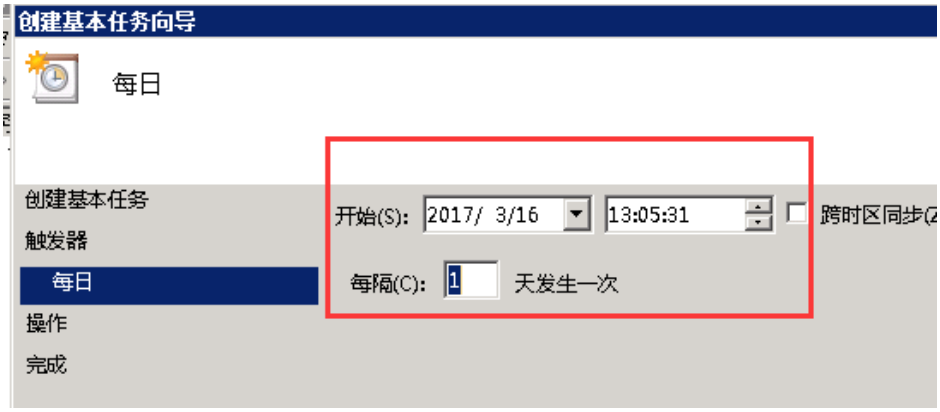


在开始菜单进入控制面板，选择系统和安全选项，单击右下角的计划任务选项，然后在计划任务栏选择创建基本任务。

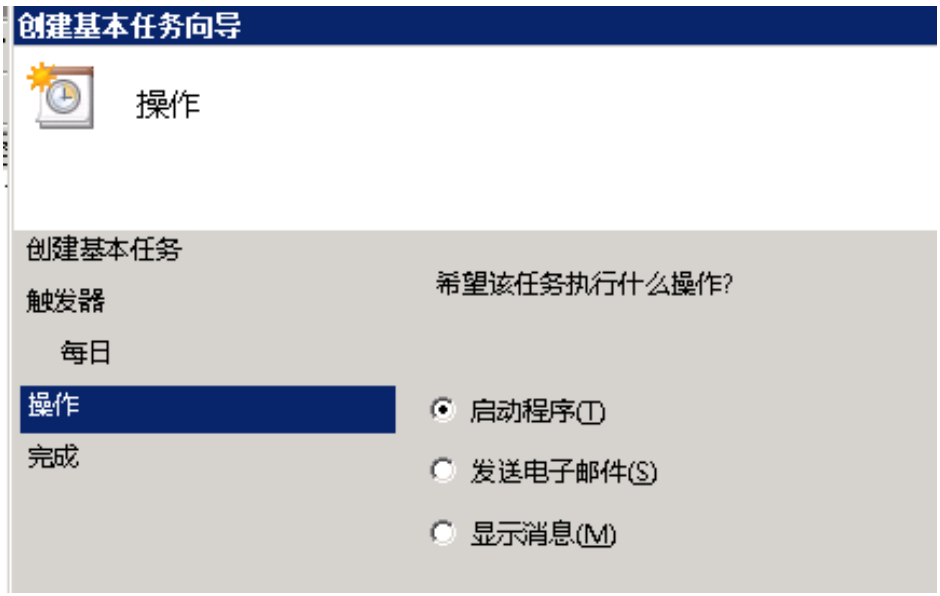


选择触发器

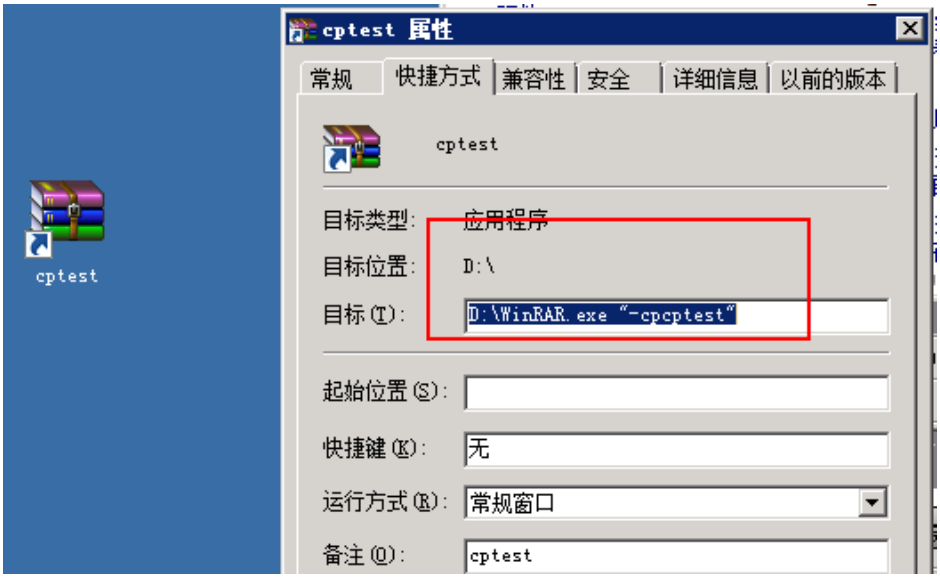
选择触发周期



选择启动程序



先找到刚才的快捷访问，右键属性，复制目标内容。

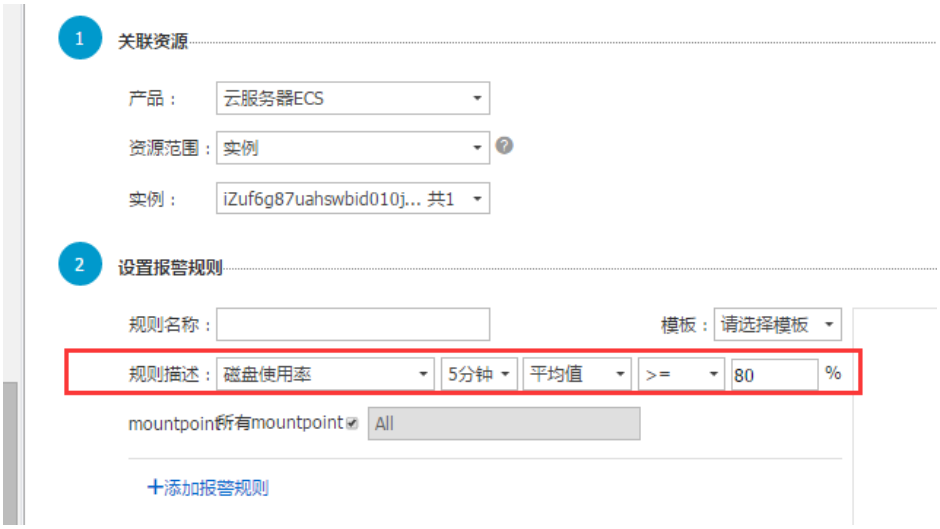


然后将复制内容粘贴到启动程序内容，点击确定完成创建

以上设置好备份策略以后，可以定期的去清理过期的备份文件，避免占用过大的空间。

设置磁盘监控

阿里云的ECS服务器有默认安装好了监控插件，如服务器无法获取磁盘监控信息，可以手动安装云监控插件，然后创建监控报警规则。可以在云监控中创建磁盘报警规则：



设置报警联系人

这样可以实时了解磁盘空间使用率是否到达一个高位值,以便及时清理。

很多客户在使用ECS，将应用部署到云端后，并不重视对数据的保护，几乎不采取任何有效的备份措施，因此我们经常遇到数据丢失无法找回的案例。

数据的丢失往往并不是云平台本身的问题，ECS提供的是底层硬件、虚拟化层面的可用性，并从物理层保证数据99.9999999%的可靠性，确保数据不会因为物理硬件的损坏而丢失，然而还有很多其他途径导致数据的丢失

，例如误删除、勒索病毒、逻辑错误等等。

数据是最重要的资产之一，一旦发生数据的丢失，造成的损失难以预估和补救。

本文档介绍如何使用快照策略和镜像备份方式对云服务器 ECS 实例进行有效的数据备份，帮助我们在出现数据丢失时能够第一时间找回数据，减少损失。

使用快照策略备份数据

快照简介

所谓快照，就是某一个时间点上某一个磁盘的数据备份。

您在使用磁盘的过程中，有可能会遇到以下需求：

当您在磁盘上进行数据的写入和存储时，希望使用某块磁盘上的数据作为其他磁盘的基础数据。

云盘（普通云盘、高效云盘和 SSD 云盘）虽然提供了安全的存储方式，确保您所存储的任何内容都不会丢失，但是如果存储在磁盘上的数据本身就是错误的数据，比如由于应用错误导致的数据错误，或者黑客利用您的应用漏洞进行恶意读写，那么就需要其他的机制来保证在您的数据出现问题时，能够恢复到您所期望的数据状态。

通过快照技术的实现，可以简单高效的满足上述需求。

快照使用增量的方式，两个快照之间只有数据变化的部分才会被拷贝，如下图所示：

快照可以分为手动快照和自动快照。

手动快照由您手动创建。您可以根据需要，手动为磁盘创建快照，作为数据备份。

自动快照是阿里云自动为您创建的快照。您需要首先创建自动快照策略，然后再把自动快照策略应用到磁盘上，阿里云就会在您设置的时间，自动为该磁盘创建快照。

快照功能已于3月28日正式商业化，将按照快照数据实际占用的存储容量来收费，具体收费模式：点此查看。

快照2.0限制每块磁盘的快照数量为64个，即最多可以为每块磁盘创建64个快照吗，不论是系统盘还是数据盘。

快照适用场景

快照是非常有价值的功能，使用快照可以在以下场景中迅速恢复数据：

- 病毒感染
- 人为误操作
- 恶意篡改
- 系统宕机造成的数据损坏
- 应用程序BUG造成的数据损坏
- 存储系统BUG造成的数据损坏

那么快照可以在以下场景下对数据起到保护作用，主要包括：

1、定期数据备份，按照设定的周期，每日、每周或每月自动执行快照策略对数据进行备份。

2、临时数据备份，例如：

a) 系统更新、应用发布等系统临时变更，为防止操作错误，在执行变更前手工创建快照对系统进行备份;

b) 系统盘扩容;

c) 磁盘数据迁移，通过对磁盘执行快照，将磁盘作为另一块磁盘的基础数据.

基于快照的机制，我们了解到快照是对磁盘状态的拷贝，在某些场景下，并不适合使用快照来备份数据，例如：

1、要求实现颗粒度恢复，例如只需要恢复某个文件，而不是整个磁盘恢复；

2、部分微软的应用，如Windows Active Directory、Exchange邮件系统等。

大部分场景下，快照都是非常有价值的备份手段，因此强烈建议开启自动快照策略，并在ECS或磁盘创建后第一时间应用快照策略。

创建自定义自动快照策略

通过创建磁盘的自动快照策略，我们可以方便的定义自动快照的创建时间、重复时间和保留时间等参数。

对于不同类型的数据，我们可以采取不同的快照策略来实现更精细化的数据备份颗粒度。以下快照策略供参考：

- 系统盘：每天凌晨0:00执行，保留30天
- 应用服务器：每天22:00执行，保留60天
- 文件服务器：每6小时执行一次，保留30天
- 数据库服务器：每天7点和19点执行，保留30天

具体的操作如下：

1、登录云服务器管理控制台。

2、单击左侧导航中的 **快照>自动快照策略**。可以看到自动快照策略列表：

3、单击右上角的 **创建自动快照策略**。

4、定义自动快照策略的参数。

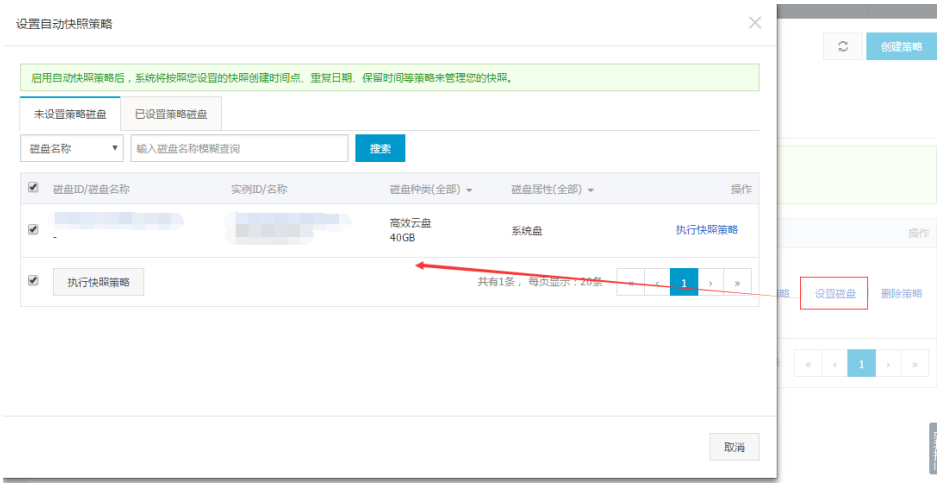
- 策略名称：自动快照策略的名称，长度为 2~128 个字符，以大小写字母或中文开头，可包含数字，”.””_”或”-”等字符。
- 创建时间：每天有 24 个时间点创建快照，从00:00 ~ 23:00可选。
- 重复日期：每周有 7 天重复日期，从周一 ~ 周日可选。
- 保留时间：快照保留的天数，1~65536 或永久保留可选，默认 30 天。

5、单击确认，自动快照策略创建好之后，需要将此策略应用到磁盘。

6、单击左侧导航中的快照>自动快照策略。

7、找到需要执行的自动快照策略，单击其右侧的设置磁盘。

8、单击 **未设置策略磁盘** 页签，找到要执行策略的磁盘，单击其右侧的执行快照策略；如果你有多块磁盘还可以选择多个磁盘，单击下面的执行快照策略。



9、设置完之后可以看到关联磁盘数变为1了：

从上面的过程我们可以知道自动快照策略与普通快照相比有以下优势：

- 自动快照策略可以对多块磁盘同时创建快照，提高了管理员的工作效率；
- 快照的保留期限我们可以自定义，这样子可以保证快照不会积累过多占用服务器的空间；
- 我们可以根据实际需求自定义快照的创建时间，重复日期，灵活调整需求，减少人工干预，节省管理员的时间，真正实现自动化运维；

通过快照回滚磁盘

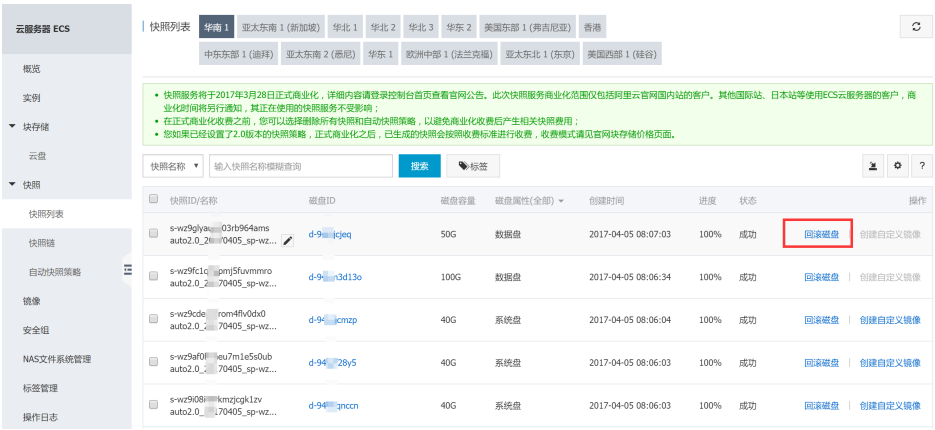
当发生意外，导致数据丢失时，我们需要从快照恢复数据，操作方法如下：

方法一

1、在控制台下选择云服务器ECS。

2、在ECS控制台下找到快照和快照列表。

3、找到对应磁盘的快照，并注意查看快照的时间，确保快照包含了我们需要还原的数据。



4、点击右方的“回滚磁盘”，即可还原。

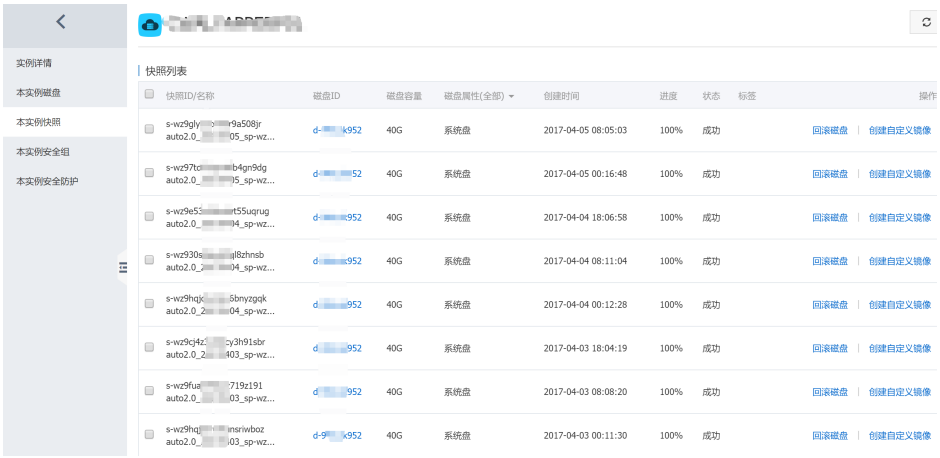
方法二

1、在控制台下选择云服务器ECS。

2、选择需要还原磁盘数据的ECS实例。

3、在实例详情下，可以看到“本实例快照”。

4、在本实例快照下，对应磁盘的快照，并注意查看快照的时间，确保快照包含了我们需要还原的数据。



5、点击右方的“回滚磁盘”，即可还原。

使用自定义镜像备份数据

快照是跟随虚拟机磁盘存储的，不能脱离虚拟机磁盘使用，而虚拟机磁盘不能跨可用区和区域恢复。如果我们需要将备份存储或恢复到其他可用区、区域时，就要用到自定义镜像。

注意自定义镜像默认是不能够跨区域使用的，如果需要跨区域使用则需要先将镜像复制到其他区域，参考：**复制镜像**。

自定义镜像包括使用实例创建自定义镜像和使用快照自定义镜像。

镜像简介

镜像是云服务器 ECS 实例运行环境的模板，一般包括操作系统和预装的软件。您可以使用镜像创建新的 ECS 实例和更换 ECS 实例的系统盘。

云服务器 ECS 提供了以下灵活多样的方式让您方便的获取镜像：

- 选择阿里云官方提供的公共镜像（支持 Linux 和 Windows 的多个发行版本）
- 去镜像市场选择第三方服务商（ISV）提供的镜像
- 根据现有的云服务器 ECS 实例创建自定义镜像
- 选择其他阿里云用户共享给您的镜像

您可以把线下环境的镜像文件导入到ECS的集群中生成一个自定义镜像。

您还可以把自定义镜像复制到其他地域，实现环境和应用的跨地域一致性部署。

镜像适用场景

镜像适用于以下场景：

1、备份短期内不会更改的系统，如已经完成发布或更新的应用系统。

2、以已经完成安装和配置的系统为模板，创建新的应用服务器，如批量部署。

3、系统及数据迁移，如将经典网络的ECS迁移到VPC下。

4、跨可用区和地域还原。

使用实例创建自定义镜像

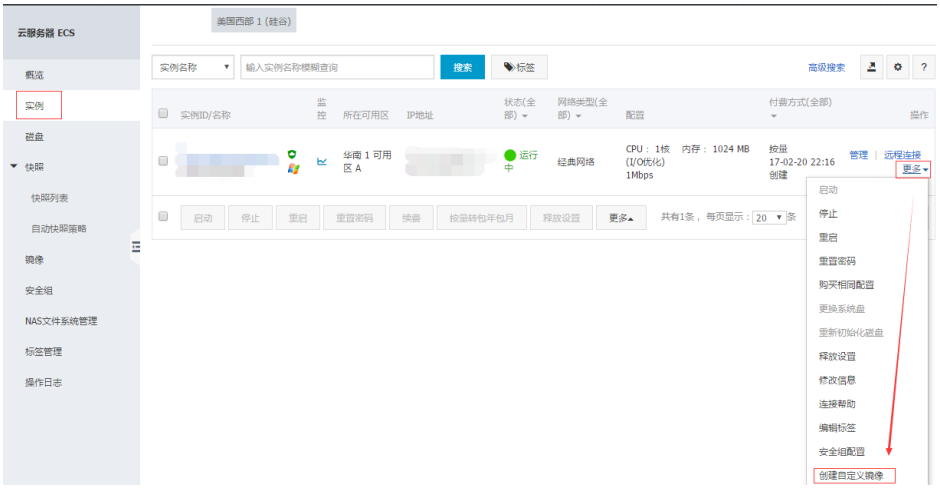
通过基于实例创建自定义镜像，我们可以把实例中的所有磁盘，包括系统盘和数据盘中的数据，全部完整的复制到自定义镜像中。

在创建自定义镜像的过程中，该实例的每块磁盘都会自动创建一个新快照，这些新快照构成了一个完整的自定义镜像。

注意：请将实例中的敏感数据删除之后再创建自定义镜像，避免数据安全隐患。

操作步骤

1、登录云服务器管理控制台，单击左侧导航栏中的实例，在实例列表页面顶部，选择目标实例所在的地域，找到需要的实例。单击列表最右侧的更多>创建自定义镜像。



2、输入镜像名称和描述信息，然后单击创建。

3、所有磁盘的快照全部创建结束后，镜像才能使用。请耐心等待。

使用快照创建自定义镜像

自定义镜像是 ECS 实例系统盘某一时刻的快照，我们可以使用快照创建自定义镜像，将快照的操作系统、数据环境信息完整的包含在镜像中。然后使用自定义镜像创建多台具有相同操作系统和数据环境信息的实例，非常方便的复制实例，而且也快速节省管理员的时间，提高了管理员的工作效率。

说明

一个帐号在一个地域最多能创建 100 个自定义镜像。

创建的自定义镜像不能跨区域使用。

通过自定义镜像开通的云服务器可以更换操作系统。更换系统后原来的自定义镜像还能够还可以继续使用。

使用自定义镜像开通的云服务器可以升级 CPU、内存、带宽、硬盘等。

自定义镜像功能不受售卖模式限制，即不区分包年包月和按量付费。包年包月云服务器的自定义镜像，可以用于开通按量付费的云服务器；反之亦然。

用于创建自定义镜像的云服务器到期或数据释放后（即用于快照的系统盘到期或释放），创建的自定义镜像不会受影响，使用自定义镜像开通的云服务器也不会受影响。但自动快照则会随着云服务器释放而被清除。

Linux 注意事项

在使用 Linux 系统创建自定义镜像时，注意不要在 /etc/fstab 文件中加载数据盘的信息，否则使用该

镜像创建的实例无法启动。

强烈建议您在制作自定义镜像前把 Linux 下的数据盘都 unmount，然后再打快照和创建自定义镜像，否则有可能造成以该自定义镜像创建的云服务器不能启动和使用。

内核和操作系统版本请不要随意进行升级。

请勿调整系统盘分区，目前只支持单个根分区。

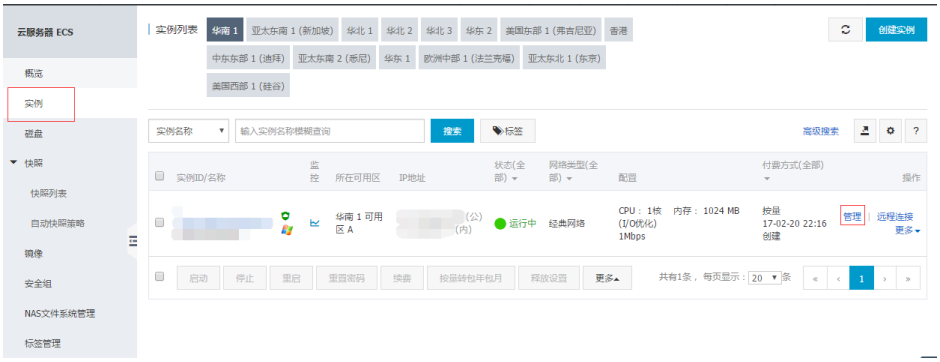
请检查系统盘使用剩余空间，确保系统盘没有被写满。

请勿修改关键系统文件如 /sbin, /bin, /lib 目录等。

请勿修改默认登录用户名root。

操作步骤

1、登录云服务器管理控制台，单击实例所在的地域，然后单击左侧导航的实例。单击实例的名称，或在实例右侧，单击管理：



2、单击左侧的本实例快照。确定快照的磁盘属性是系统盘，数据盘不能用于创建镜像。然后单击创建自定义镜像。

3、在弹出的对话框中，您可以看到快照的 ID。输入自定义镜像的名称和描述。

4、在对系统快照创建自定义镜像的过程中，我们还可以选择多块数据盘快照，包含在该镜像中（如下图）

注意：请将数据盘中的敏感数据删除之后再创建自定义镜像，避免数据安全隐患。如果快照 ID 为空，则该磁盘会作为空盘创建，默认容量为 5GB。

如果选择了快照，则磁盘容量为快照的容量。

5、单击创建。自定义镜像创建成功，我们可以单击左侧导航中的镜像，然后查看创建的镜像。

在处理客户磁盘相关问题时，您经常会遇到操作系统中数据盘分区丢失的情况。本文档介绍了 Linux 下常见的数据分区丢失问题以及对应的处理方法，同时给出客户最佳实践以避免可能的数据丢失风险。

前提条件

在对数据修复之前，首先需要对分区丢失的数据盘创建快照，快照创建完成后再进行尝试修复。如果在修复过

程中出现问题，可以通过快照回滚还原到修复之前的状态。

工具说明

Linux 下磁盘分区修复和数据恢复使用的工具：fdisk，testdisk， partprobe 。

- fdisk

Linux 系统默认有的分区工具。

- testdisk

Linux 系统默认没有安装。比如 Centos 系统可以通过 yum install -y testdisk 在线进行安装。主要用作对 Linux 系统磁盘分区恢复或者数据恢复。

- partprobe

Linux 默认工具。主要是在系统不重启的情况下，让 kernel 重新读取分区。

Linux 下数据盘分区丢失和数据恢复处理办法

Linux 数据盘分区丢失或者数据丢失一般是用户重启系统后显现出来的。首先怀疑可能是用户 /etc/fstab 下没有配置自动挂载，所以先让用户手动挂载下。

如果手动挂载出现报分区表丢失，那么您可以通过如下三种办法先尝试进行处理。

通过 fdisk 进行分区恢复

一般用户对数据盘分区的时候，分区磁盘的起止扇区一般使用默认的值，所以可以先尝试直接使用 fdisk 新建分区进行恢复。

```
[root@Aliyun ~]# fdisk /dev/xvdb
Welcome to fdisk (util-linux 2.23.2).

Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.


Command (m for help): n
Partition type:
   p   primary (0 primary, 0 extended, 4 free)
   e   extended
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-10485759, default 2048):
Using default value 2048
Last sector, +sectors or +size{K,M,G} (2048-10485759, default 10485759):
Using default value 10485759
Partition 1 of type Linux and of size 5 GiB is set

Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
[root@Aliyun ~]# mount /dev/xvd
xvda   xvda1  xvdb   xvdb1
[root@Aliyun ~]# mount /dev/xvdb
xvdb   xvdb1
[root@Aliyun ~]# mount /dev/xvdb1 /mnt/
[root@Aliyun ~]# ls /mnt/
123.sh  configclient  data  diamond  install_edsd.sh  install.sh  ip.qz  logs  lost+found  test
```


如果这个方法尝试无效，那么就使用 testdisk 工具尝试修复。

fdisk 分区操作说明：格式化和挂载数据盘。

通过 testdisk 工具恢复分区

1. 输入 testdisk /dev/xvdb (请写需要回复的磁盘名称)，然后默认“Proceed” 回车。

```
TestDisk 7.0, Data Recovery Utility, April 2015
Christophe GRENIER <grenier@cgsecurity.org>
http://www.cgsecurity.org

  TestDisk is free software, and
comes with ABSOLUTELY NO WARRANTY.

Select a media (use Arrow keys, then press Enter):
>Disk /dev/xvdb - 5368 MB / 5120 MiB


>[Proceed ]  [  Quit  ]

Note: Disk capacity must be correctly detected for a successful recovery.
If a disk listed above has incorrect size, check HD jumper settings, BIOS
detection, and install the latest OS patches and disk drivers.
```

2. 选择 默认一般选择 “Intel” ，如果您是 GPT 分区，则选择 “EFI GPT” 进行扫描：

```
TestDisk 7.0, Data Recovery Utility, April 2015
Christophe GRENIER <grenier@cgsecurity.org>
http://www.cgsecurity.org

Disk /dev/xvdb - 5368 MB / 5120 MiB

Please select the partition table type, press Enter when done.
>[Intel  ] Intel/PC partition
 [EFI GPT] EFI GPT partition map (Mac i386, some x86_64...)
 [Humax  ] Humax partition table
 [Mac    ] Apple partition map
 [None   ] Non partitioned media
 [Sun    ] Sun Solaris partition
 [XBox   ] XBox partition
 [Return ] Return to disk selection


Note: Do NOT select 'None' for media with only a single partition. It's very
rare for a disk to be 'Non-partitioned'.
```

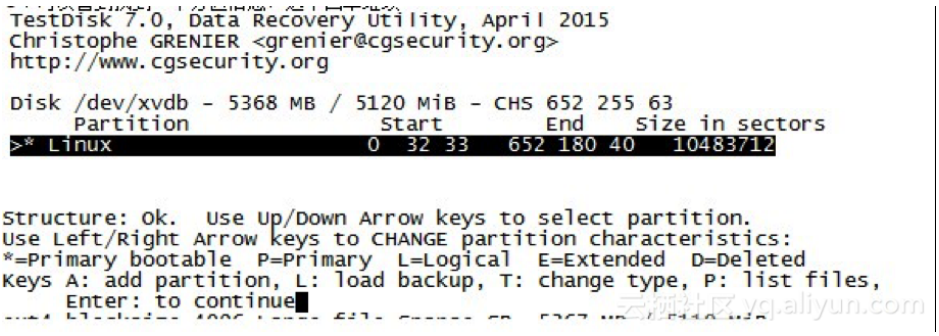
3. 选择 “Analyse” 分析回车。

```
TestDisk 7.0, Data Recovery Utility, April 2015
Christophe GRENIER <grenier@cgsecurity.org>
http://www.cgsecurity.org

Disk /dev/xvdb - 5368 MB / 5120 MiB
     CHS 652 255 63 - sector size=512

>[ Analyse  ] Analyse current partition structure and search for lost partitions
 [ Advanced ] Filesystem Utils
 [ Geometry ] Change disk geometry
 [ Options  ] Modify options
 [ MBR Code ] Write TestDisk MBR code to first sector
 [ Delete   ] Delete all data in the partition table
 [ Quit     ] Return to disk selection


Note: Correct disk geometry is required for a successful recovery. 'Analyse'
process may give some warnings if it thinks the logical geometry is mismatched.
```

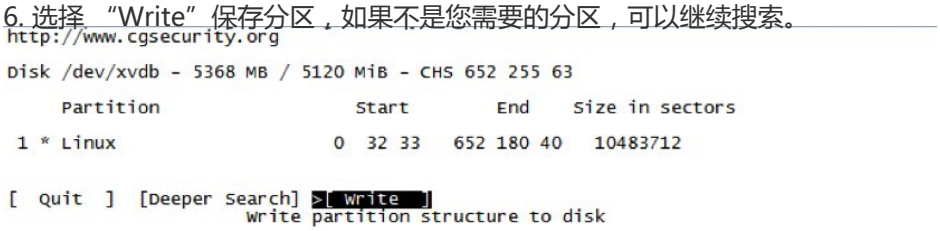
4. 可以看到没有任何信息，您继续 “Quick Search” 快速搜索回车。

```
TestDisk 7.0, Data Recovery Utility, April 2015
Christophe GRENIER <grenier@cgsecurity.org>
http://www.cgsecurity.org

Disk /dev/xvdb - 5368 MB / 5120 MiB - CHS 652 255 63
Current partition structure:
     Partition                  Start        End    Size in sectors

No partition is bootable


*=Primary bootable  P=Primary  L=Logical  E=Extended  D=Deleted
>[Quick Search]
                            Try to locate partition
```

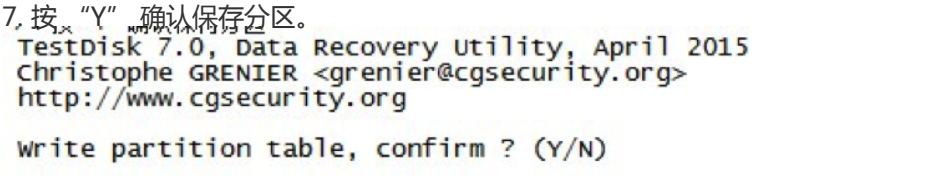
5. 可以看到找到一个分区信息，选中回车继续。



6. 选择 “Write” 保存分区，如果不是您需要的分区，可以继续搜索。

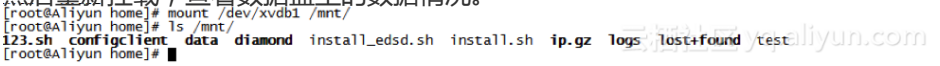


7. 按 “Y” 确认保存分区。



8. 这个时候可能的/dev 下还是看不到这个分区文件，您需要通过partprobe /dev/xvdb 命令手动刷新分区表。

然后重新挂载，查看数据盘里的数据情况。

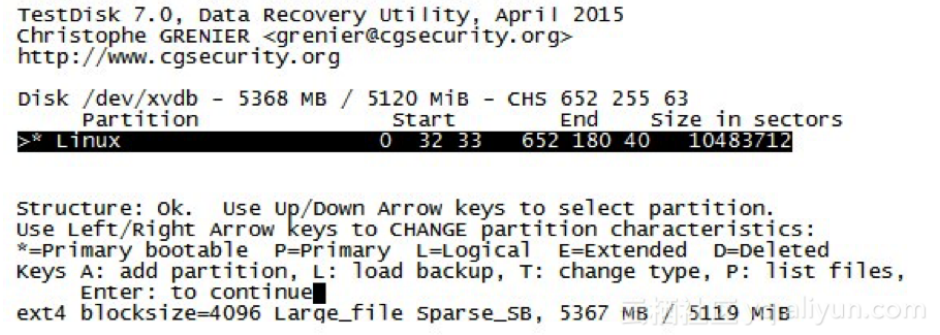


TestDisk使用说明：http://www.cgsecurity.org/wiki/TestDisk

通过 testdisk 直接恢复数据

在某些情况下，tedisk 扫描出分区，但是无法保存分区的时候，可以尝试直接把文件恢复处理，具体处理步骤如下：

1. testdisk 已经找到分区，您可以按 “P” 列出文件。



2. 可以看见存在的文件，将要恢复的文件选中，然后按 “C” 。

```
TestDisk 7.0, Data Recovery Utility, April 2015
Christophe GRENIER <grenier@cgsecurity.org>
http://www.cgsecurity.org
   * Linux                    0  32 33   652 180 40   10483712
Directory /

 drwxr-xr-x     0     0      4096 21-Feb-2017 11:57 .
 drwxr-xr-x     0     0      4096 21-Feb-2017 11:57 ..
 drwx------     0     0     16384 21-Feb-2017 11:56 lost+found
 -rw-r--r--     0     0      1701 21-Feb-2017 11:57 install_edsd.sh
 -rw-r--r--     0     0      5848 21-Feb-2017 11:57 install.sh
>-rw-r--r--     0     0     12136 21-Feb-2017 11:57 ip.gz
 -rw-r--r--     0     0         0 21-Feb-2017 11:57 test
 drwxr-xr-x     0     0      4096 21-Feb-2017 11:57 123.sh
 drwxr-xr-x     0     0      4096 21-Feb-2017 11:57 configclient
 drwxr-xr-x     0     0      4096 21-Feb-2017 11:57 data
 drwxr-xr-x     0     0      4096 21-Feb-2017 11:57 diamond
 drwxr-xr-x     0     0      4096 21-Feb-2017 11:57 logs


                                                    Next
 Use Right to change directory, h to hide deleted files
     q to quit, : to select the current file, a to select all files
     C to copy the selected files, c to copy the current file
```

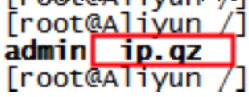

3. 然后选择需要复制的目标目录，您以恢复到home为例。

```
TestDisk 7.0, Data Recovery Utility, April 2015

Please select a destination where /ip.gz will be copied.
Keys: Arrow keys to select another directory
      C when the destination is correct
      Q to quit
Directory /
 drwxr-xr-x     0     0      4096 11-Jan-2017 09:32 .
 drwxr-xr-x     0     0      4096 11-Jan-2017 09:32 ..
 dr-xr-xr-x     0     0      4096 25-Jul-2016 16:23 boot
 drwxr-xr-x     0     0      2940 21-Feb-2017 12:30 dev
 drwxr-xr-x     0     0      4096 21-Feb-2017 12:12 etc
>drwxr-xr-x     0     0      4096 16-Feb-2017 11:48 home
 drwx------     0     0     16384 12-May-2016 19:58 lost+found
 drwxr-xr-x     0     0      4096 12-Aug-2015 22:22 media
 drwxr-xr-x     0     0      4096 21-Feb-2017 11:57 mnt
 drwxr-xr-x     0     0      4096 12-Aug-2015 22:22 opt
 dr-xr-xr-x     0     0         0 16-Feb-2017 21:35 proc
 dr-xr-x---     0     0      4096 21-Feb-2017 11:57 root
 drwxr-xr-x     0     0       560 21-Feb-2017 12:12 run
 drwxr-xr-x     0     0      4096 12-Aug-2015 22:22 srv
 dr-xr-xr-x     0     0         0 16-Feb-2017 21:35 sys
 drwxrwxrwt     0     0      4096 21-Feb-2017 12:34 tmp
 drwxr-xr-x     0     0      4096 16-Feb-2017 11:48 usr
 drwxr-xr-x     0     0      4096 16-Feb-2017 21:35 var
 lrwxrwxrwx     0     0         7  3-May-2016 13:48 bin
 lrwxrwxrwx     0     0         7  3-May-2016 13:48 lib
 lrwxrwxrwx     0     0         9  3-May-2016 13:48 lib64
 lrwxrwxrwx     0     0         8  3-May-2016 13:48 sbin
```


4. 可以看到提示复制成功。

```
TestDisk 7.0, Data Recovery Utility, April 2015
Christophe GRENIER <grenier@cgsecurity.org>
http://www.cgsecurity.org
   * Linux                    0  32 33   652 180 40   10483712
Directory /
Copy done! 1 ok, 0 failed
 drwxr-xr-x     0     0      4096 21-Feb-2017 11:57 .
 drwxr-xr-x     0     0      4096 21-Feb-2017 11:57 ..
 drwx------     0     0     16384 21-Feb-2017 11:56 lost+found
 -rw-r--r--     0     0      1701 21-Feb-2017 11:57 install_edsd.sh
 -rw-r--r--     0     0      5848 21-Feb-2017 11:57 install.sh
>-rw-r--r--     0     0     12136 21-Feb-2017 11:57 ip.gz
 -rw-r--r--     0     0         0 21-Feb-2017 11:57 test
 drwxr-xr-x     0     0      4096 21-Feb-2017 11:57 123.sh
 drwxr-xr-x     0     0      4096 21-Feb-2017 11:57 configclient
 drwxr-xr-x     0     0      4096 21-Feb-2017 11:57 data
 drwxr-xr-x     0     0      4096 21-Feb-2017 11:57 diamond
 drwxr-xr-x     0     0      4096 21-Feb-2017 11:57 logs
```

5. 切换到 home 目录查看，可以看见文件已经恢复了。



常见误区与最佳实践

数据是用户的核心资产，很多用户在ECS上构建网站、自建数据库(MYSQL/MongoDB/REDIS)。如果出现数据丢失情况，会给用户的业务带来巨大的风险。如下是您在数据安全方面总结常见误区和最佳实践。

常见误区

有些用户认为阿里云的底层存储基于三副本，因此认为操作系统内数据没有任何丢失风险。实际上这是误解，底层存储的三副本提供对数据磁盘的物理层保护，但如果系统内部使用云盘逻辑上出现问题，比如中毒，误删数据，文件系统损坏等情况，还是可能出现数据丢失。此时，您需要通过快照、异地备份等相关技术最大保证数据的安全性。

云盘的三副本说明

ECS 用户对虚拟磁盘的读写最终都会被映射为对阿里云数据存储平台上的文件的读写。阿里云提供一个扁平的线性存储空间，在内部会对线性地址进行切片，一个分片称为一个 Chunk；对于每一个 Chunk，阿里云会复制出三个副本，并将这些副本按照一定的策略存放在集群中的不同节点上，保证用户数据的可靠。至于 ECS 实例内由于病毒感染、人为误删除或黑客入侵等软故障原因造成的数据丢失，需要采用备份、快照等技术手段来解决。任何一种技术都不可能解决全部的问题，因地制宜的选择合适的数据保护措施，才能为宝贵的业务数据筑起一道坚实的防线。具体请参考：云盘三副本技术介绍。

最佳实践

数据盘分区恢复以及数据恢复是处理数据丢失问题最后的一道防线，但未必一定能够恢复数据。您强烈建议用户参考如下最佳实践，通过数据进行自动快照、手动快照快照和各类备份方案，最大程度保证数据的安全性。

启用自动快照

根据实际业务情况，对系统盘、数据盘启动自动快照。需要注意的是，自动快照在更换系统盘、服务器到期后或手动释放磁盘时，自动快照可能会被释放。

关于自动快照释放行为，可以在 ECS控制台>全部磁盘 中找到对应磁盘，选择 修改磁盘属性 进行设置，默认选择 自动快照随磁盘释放，选择后，当磁盘手动释放、磁盘随实例释放或更换系统盘时，该磁盘的自动快照会被自动删除。如果想保留快照，您可以手动去掉该选项。详情请参考：ECS云服务器自动快照FAQ。

手动快照

请在任何重要或有风险的操作前，请手动执行快照。例如：

- 系统升级内核
- 应用升级变更
- 磁盘数据恢复

在对用户磁盘做恢复的时候，一定要先对创建该磁盘的快照，快照完成后做相应的操作。

OSS、线下、异地备份

用户可酌情使用OSS、线下、异地的方式进行重要数据的备份。

在处理客户磁盘相关问题时，您经常会遇到操作系统中数据盘分区丢失的情况。本文档介绍了 Windows 下常见的数据分区丢失问题以及对应的处理方法，同时给出客户最佳实践以避免可能的数据丢失风险。

前提条件

在对数据修复之前，首先需要对分区丢失的数据盘创建快照，快照创建完成后再进行尝试修复。如果在修复过程中出现问题，可以通过快照回滚还原到修复之前的状态。

工具说明

Windows 下磁盘管理，数据恢复软件：

- 磁盘管理

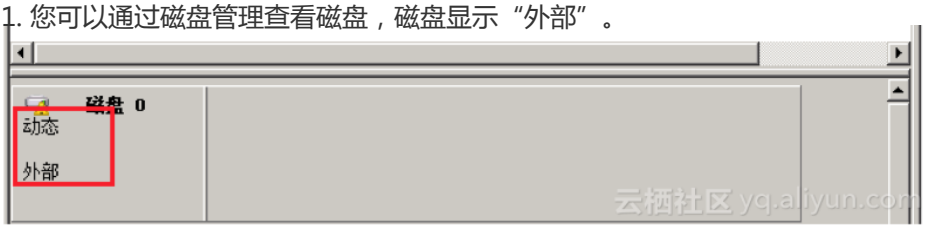
系统自带工具，可以对磁盘进行分区格式化等操作。

- 数据恢复软件

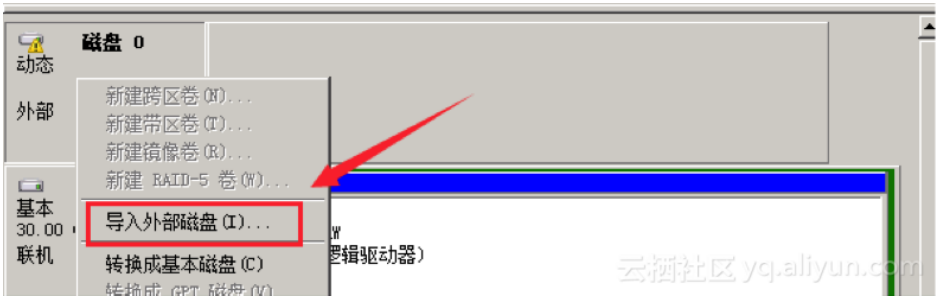
一般是商业软件，可以去相应的官网进行下载使用。主要作用是文件系统异常恢复数据。

磁盘显示为 “外部” 磁盘导致没有显示分区

1. 您可以通过磁盘管理查看磁盘，磁盘显示“外部”。

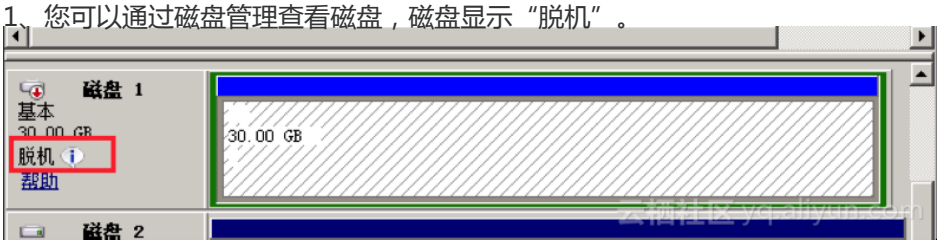


2. 针对显示为“外部”的磁盘，可以在磁盘区块上右击，选择导入外部磁盘，单击确定即可。

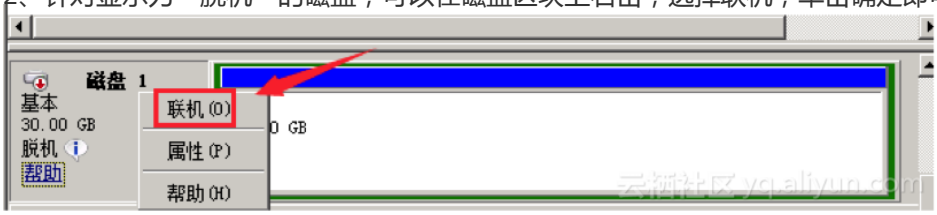


磁盘显示为 “脱机” 状态导致没有显示分区

1、您可以通过磁盘管理查看磁盘，磁盘显示“脱机”。

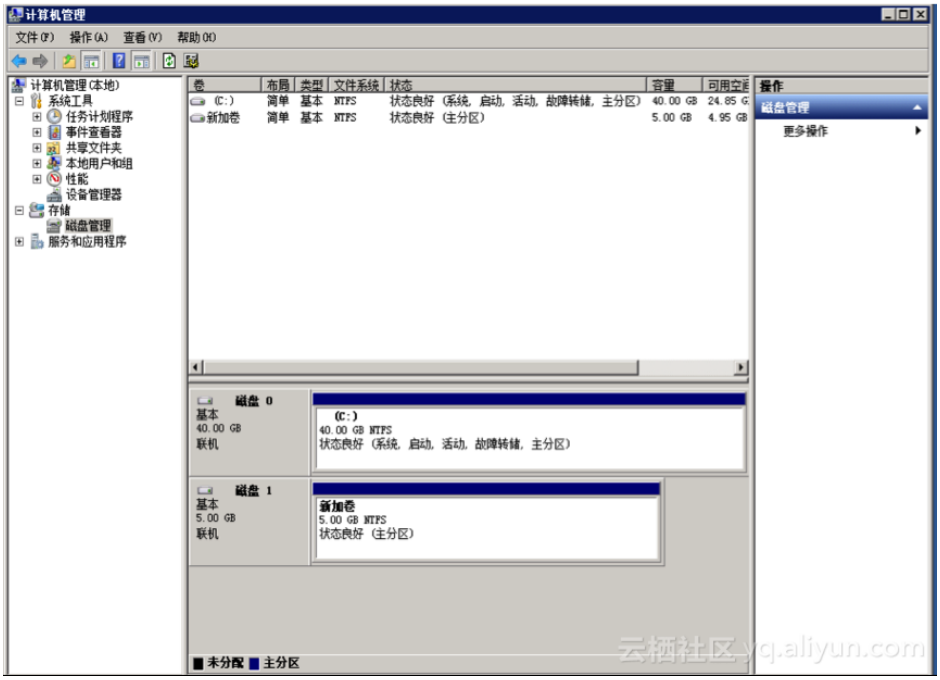


2、针对显示为“脱机”的磁盘，可以在磁盘区块上右击，选择联机，单击确定即可。

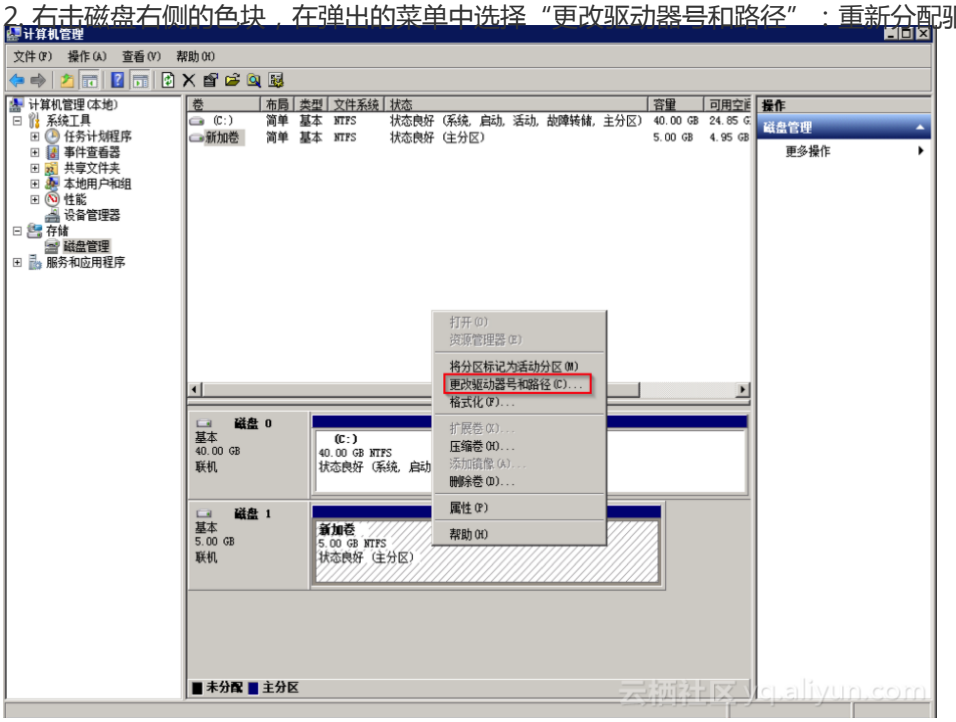


未分配盘符导致无法显示分区

1. 在磁盘管理，可以看到数据盘被系统正确识别，但是未分配盘符给这块磁盘。

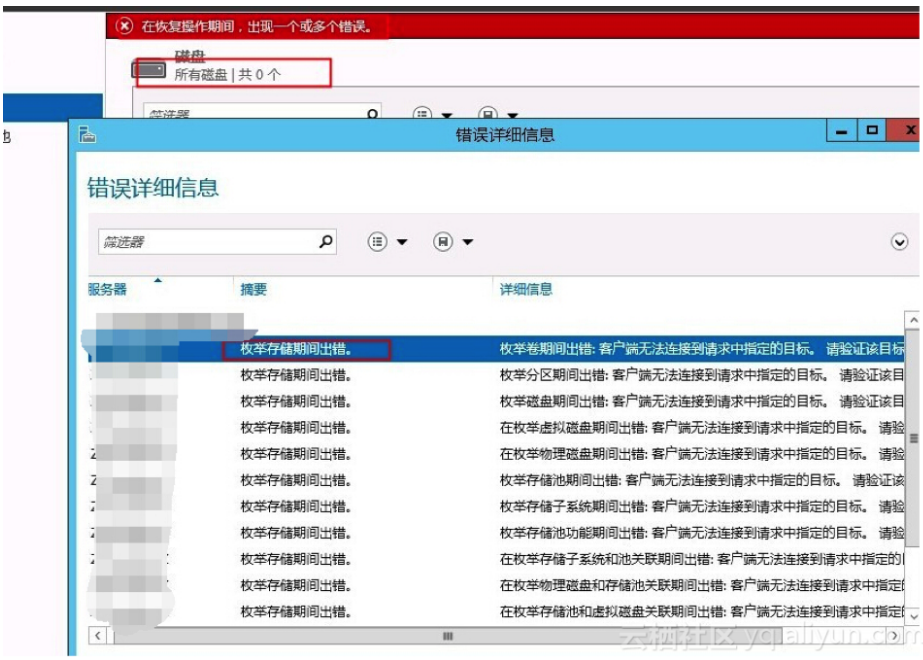


2. 右击磁盘右侧的色块，在弹出的菜单中选择“更改驱动器号和路径”：重新分配驱动号即可。

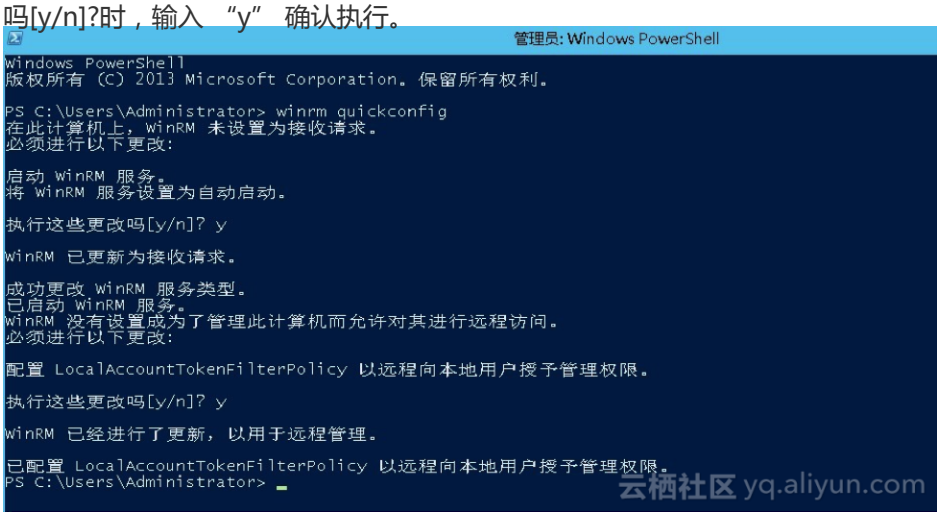


在磁盘管理无法查看数据盘，出现“枚举卷期间出错”的报错

1. 在磁盘管理里面无法查看到数据盘，在系统日志里面报“枚举卷期间出错”错误：



2. 打开Windows PowerShell 命令窗口，执行winrm quickconfig命令进行修复，在弹出询问： 执行这些更改吗[y/n]?时，输入 “y” 确认执行。



3. 修复完毕后重新打开磁盘管理，数据盘已可以正常显示。

数据盘变成RAW

在某些特殊情况下，您发现Windows下Disk变为RAW格式。Disk 显示 Raw disk 是因为 Windows 无法识别其上的文件系统。这通常是记录文件系统类型或者位置的信息丢失或者损坏了，如 partition table 或者 boot sector。比较可能的原因列举如下：

- 外接硬盘发生这种问题通常是因为断开时没有使用” safely remove hardware” 的选项。
- 意外断电导致的磁盘问题也比较常见。
- 硬件层故障也可能导致磁盘分区信息丢失。
- 底层与磁盘相关的driver或应用，例如您使用的diskprobe工具就可以直接修改磁盘的表结构。
- 计算机病毒。

微软官方给出的修复磁盘RAM是使用Dskprobe工具进行修复，详情请参考微软官方文档 Dskprobe Overview：https://technet.microsoft.com/en-us/library/cc736327(v=ws.10).aspx。
除了上述此外，Windows下有大量的免费或商业的数据恢复软件来进行丢失数据的找回。例如，您可以尝试使用Disk Genius工具扫描，来尝试恢复相应的文件。

常见误区与最佳实践

数据是用户的核心资产，很多用户在ECS上构建网站、自建数据库(MYSQL/MongoDB/REDIS)。如果出现数据丢失情况，会给用户的业务带来巨大的风险。如下是您在数据安全方面总结常见误区和最佳实践。

常见误区

有些用户认为阿里云的底层存储基于三副本，因此认为操作系统内数据没有任何丢失风险。实际上这是误解，底层存储的三副本提供对数据磁盘的物理层保护，但如果系统内部使用云盘逻辑上出现问题，比如中毒，误删数据，文件系统损坏等情况，还是可能出现数据丢失。此时，您需要通过快照、异地备份等相关技术最大保证数据的安全性。

云盘的三副本说明

ECS 用户对虚拟磁盘的读写最终都会被映射为对阿里云数据存储平台上的文件的读写。阿里云提供一个扁平的线性存储空间，在内部会对线性地址进行切片，一个分片称为一个 Chunk；对于每一个 Chunk，阿里云会复制出三个副本，并将这些副本按照一定的策略存放在集群中的不同节点上，保证用户数据的可靠。至于 ECS 实例内由于病毒感染、人为误删除或黑客入侵等软故障原因造成的数据丢失，需要采用备份、快照等技术手段来解决。任何一种技术都不可能解决全部的问题，因地制宜的选择合适的数据保护措施，才能为宝贵的业务数据筑起一道坚实的防线。具体请参考：云盘三副本技术介绍。

最佳实践

数据盘分区恢复以及数据恢复是处理数据丢失问题最后的一道防线，但未必一定能够恢复数据。您强烈建议用户参考如下最佳实践，通过数据进行自动快照、手动快照快照和各类备份方案，最大程度保证数据的安全性。

启用自动快照

根据实际业务情况，对系统盘、数据盘启动自动快照。需要注意的是，自动快照在更换系统盘、服务器到期后或手动释放磁盘时，自动快照可能会被释放。
关于自动快照释放行为，可以在 ECS控制台>全部磁盘 中找到对应磁盘，选择 修改磁盘属性 进行设置，默认选择 自动快照随磁盘释放，选择后，当磁盘手动释放、磁盘随实例释放或更换系统盘时，该磁盘的自动快照会被自动删除。如果想保留快照，您可以手动去掉该选项。详情请参考：ECS云服务器自动快照FAQ。

手动快照

请在任何重要或有风险的操作前，请手动执行快照。例如：

- 系统升级内核
- 应用升级变更
- 磁盘数据恢复

在对用户磁盘做恢复的时候，一定要先对创建该磁盘的快照，快照完成后做相应的操作。

OSS、线下、异地备份

您可酌情使用OSS、线下、异地的方式进行重要数据的备份。

配置

简介

NTP是网络时间协议(Network Time Protocol)，它是用来同步网络中各个计算机的时间的协议，对于一些对时间极度敏感的应用（例如，通信行业），如果不同机器时间不一致，就有可能导致读取到值不同。

操作步骤

修改默认NTP服务器地址

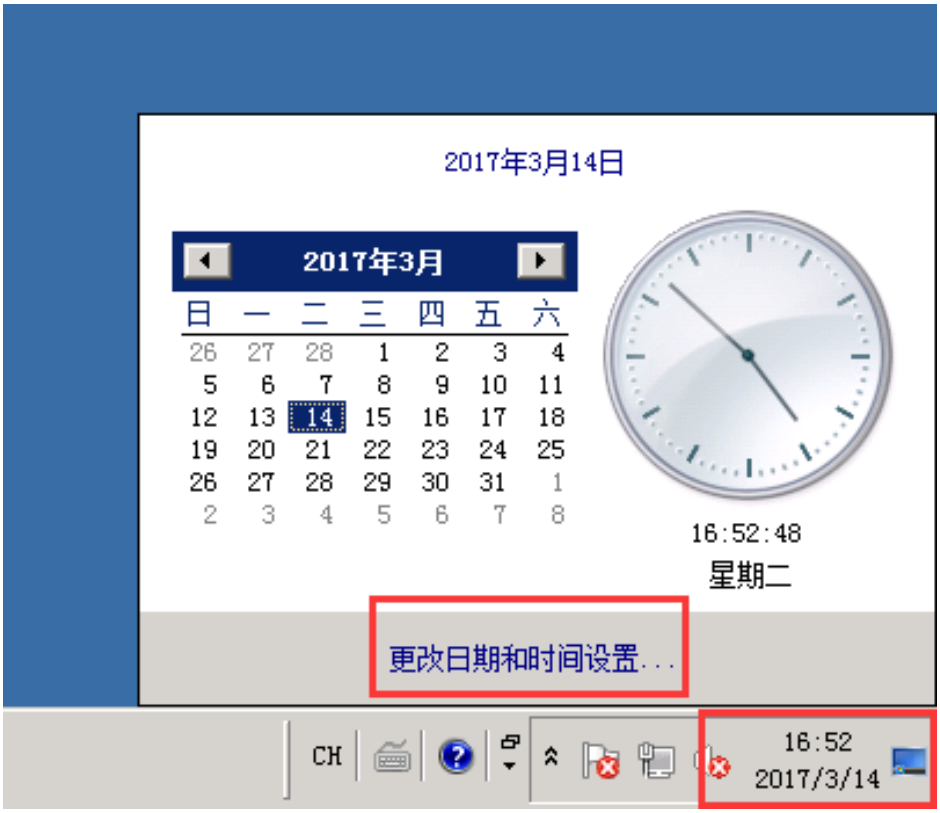
Windows Server操作系统默认都配置了微软默认的NTP服务器（time.windows.com），但可能会因为网络的原因经常出现同步出错。这时我们可以将默认的NTP服务器更换成阿里的NTP服务器。以下分别是阿里云内网和外网的NTP服务器地址。

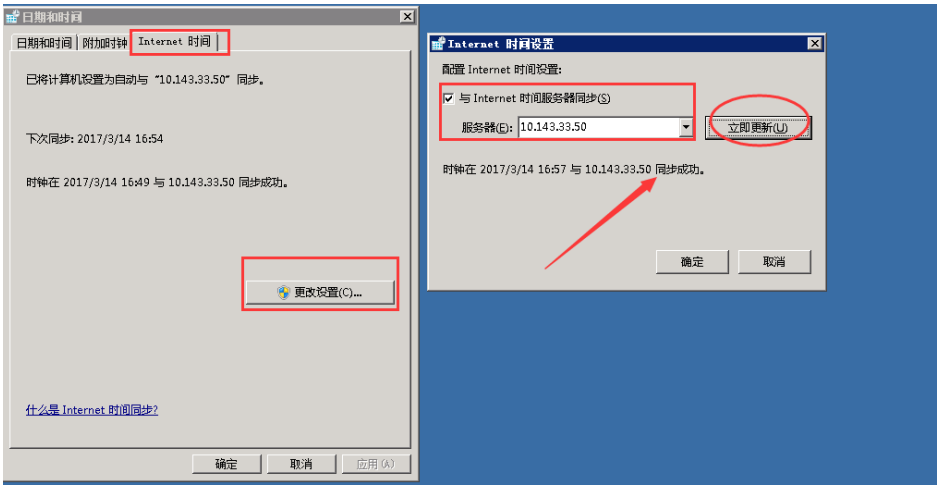
内网NTP服务器	公共NTP服务器
10.143.33.50	Unix类系统：time1-7.aliyun.com
10.143.33.51	Windows：time.pool.aliyun.com

10.143.33.49	
10.143.0.44	
10.143.0.45	
10.143.0.46	

本文以 Windows Server 2008 R2 为例。

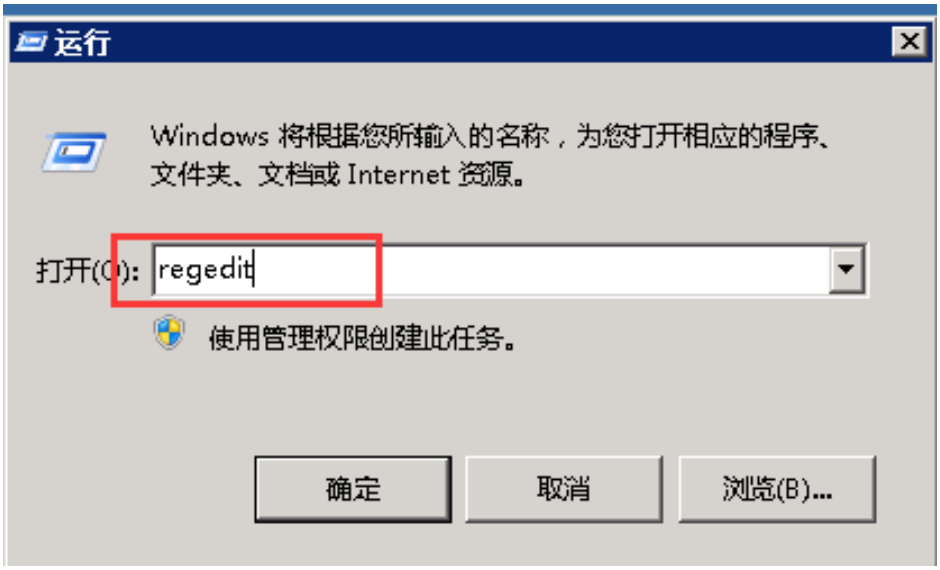
登录系统后，双击屏幕右下角的时间>更改日期和时间设置>Internet时间>更改设置>勾选与Internet时间服务器同步，服务器填写阿里云内网NTP服务器地址，然后选择立即更新，稍等一会后会提示同步成功。

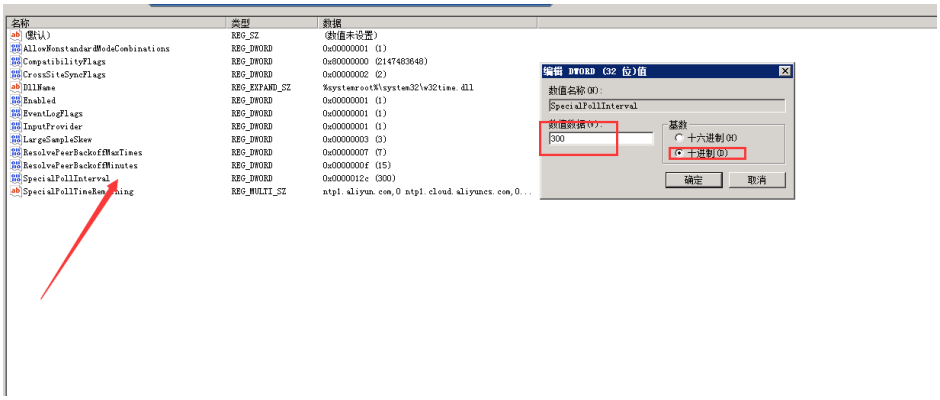




修改NTP同步的间隔

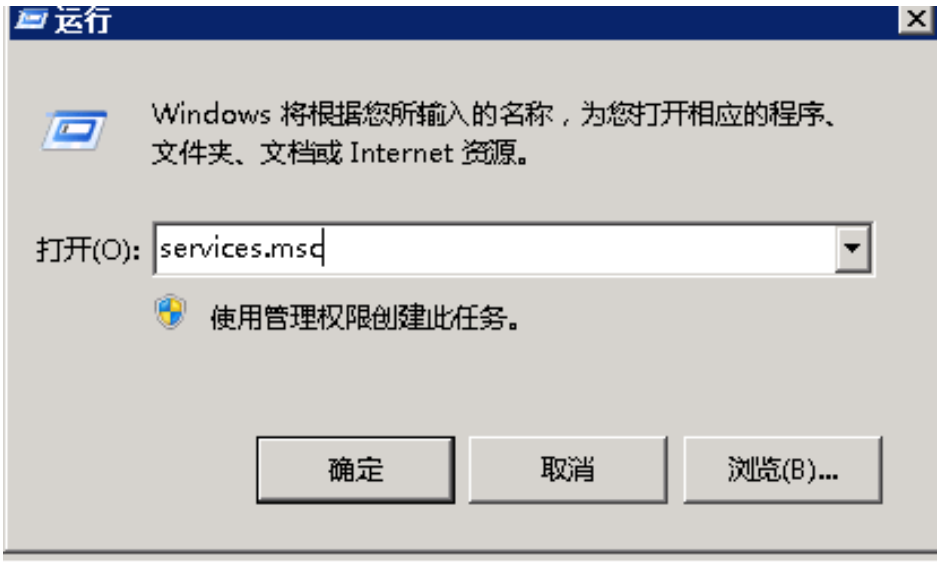
NTP同步的间隔默认是5分钟，如果想更短间隔同步一次的话可以通过修改注册表来实现Win+R键输入”regdeit”打开注册表编辑器，然后依次展开:HKEY_LOCAL_MACHINE->SYSTEM->CurrentControlSet->Services->W32Time->TimeProviders->NtpClient分支，并双击SpecialPollInterval键值，将对话框中的“基数栏”选择到“十进制”上，输入框中显示的数字正是自动对时的间隔(以秒为单位)。

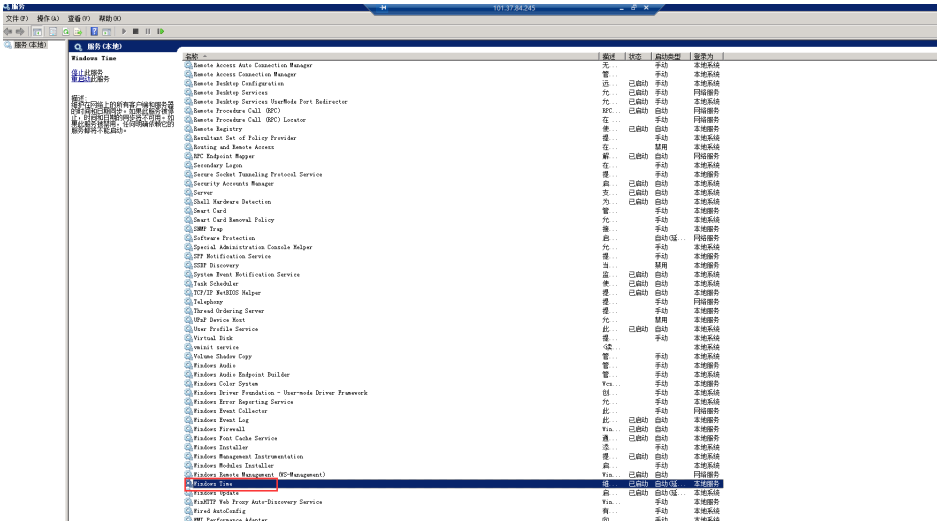


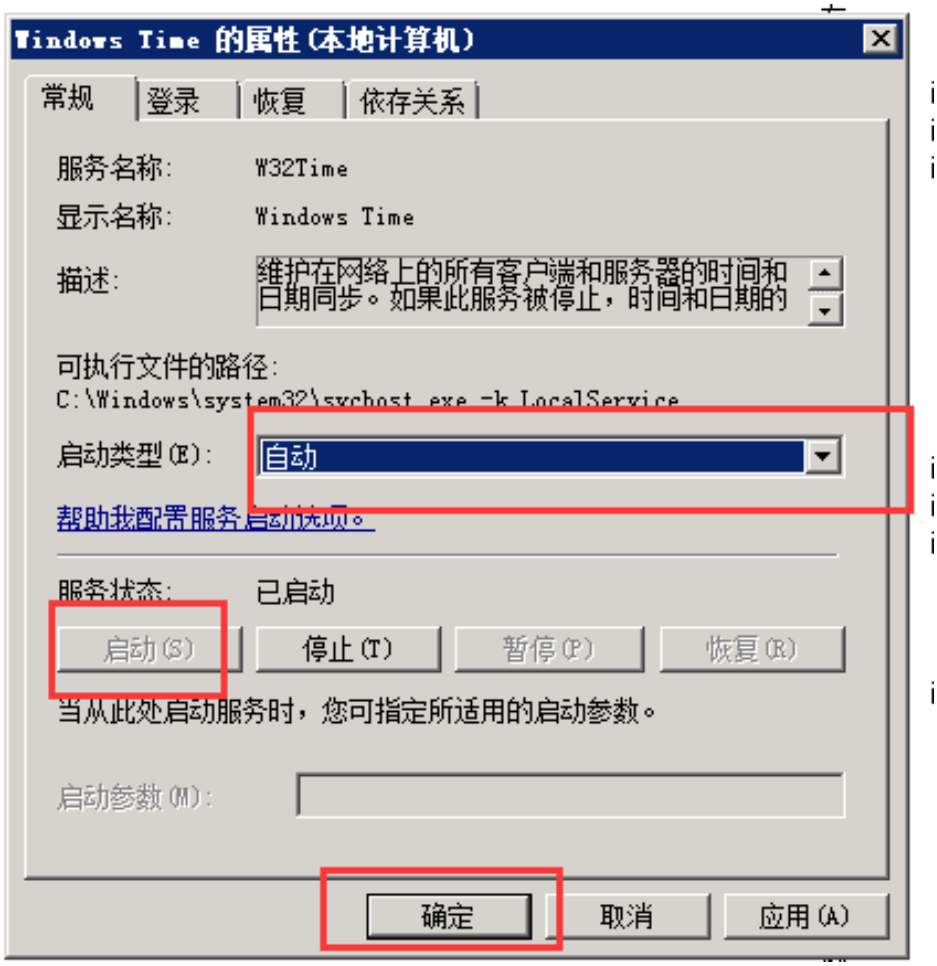


以上就是ECS之windows服务器时钟同步设置的方法，如果配置好后还是无法同步，请检查Windows time服务是否开启（默认是开启的），如果没有开启，请设置自动开启，开启方法如下：

Win+R键输入“service.msc”打开服务控制台，然后找到“Windows Time”服务>属性>启动类型>自动


运行
Windows 将根据您所输入的名称，为您打开相应的程序、文件夹、文档或 Internet 资源。
打开(O):
services.msc
使用管理权限创建此任务。
确定
取消
浏览(B)...





命令的操作方式

```
sc config W32Time start= delayed-auto                    #修改NTP配置为delayed-auto
net start w32time #启动windows时间服务
reg add HKLM\SYSTEM\CurrentControlSet\Services\W32Time\TimeProviders\NtpClient /v SpecialPollInterval /t REG_DWORD /d 0x12c /f #注册表中修改NTP的配置
w32tm /config /manualpeerlist:"ntp1.aliyun.com,0x1 ntp2.aliyun.com,0x1 ntp3.aliyun.com,0x1 ntp4.aliyun.com,0x1 ntp5.aliyun.com,0x1 ntp6.aliyun.com,0x1 ntp1.cloud.aliyuncs.com,0x1 ntp2.cloud.aliyuncs.com,0x1 ntp3.cloud.aliyuncs.com,0x1 ntp4.cloud.aliyuncs.com,0x1 ntp5.cloud.aliyuncs.com,0x1 ntp6.cloud.aliyuncs.com,0x1 ntp7.cloud.aliyuncs.com,0x1 ntp8.cloud.aliyuncs.com,0x1 ntp9.cloud.aliyuncs.com,0x1 ntp10.cloud.aliyuncs.com,0x1 ntp11.cloud.aliyuncs.com,0x1 ntp12.cloud.aliyuncs.com,0x1" /syncfromflags:manual /reliable:yes /update #更新NTP服务的地址
```

相关链接

ECS Windows默认NTP服务器设置说明

简介

在信息化高速发展的今天，服务器每天都会与其它单机交换大量文件数据，文件传输对大家来说是家常便饭。因此，其重要性就不言而喻了。文件传输方式各有不同，选择一款合适自己的文件传输工具，在工作中能起到事半功倍的效果。节省资源、方便传输、提升工作效率、加密保护等等。因此，很多文件传输工具应运而生，例如：NC、FTP、SCP、NFS、SAMBA、RSYNC/SERVERSYNC等等，每种方式都有自己的特点。本文将首先简单介绍一下文件传输的基本原理，然后，详细介绍类unix/linux、windows平台上常用文件传输方式，并针对它们各自的特点进行比较，让读者对文件传输方式有比较详尽地了解，从而能够根据不同的需要选择合适的文件传输方式。

文件传输原理

文件传输是信息传输的一种形式，它是在数据源和数据宿之间传送文件数据的过程，也称文件数据通信。操作系统把文件数据提取到内存中做暂存，再复制到目的地，加密就是在文件外加了一个壳，文件本身还是一个整体，复制只是把这个整体转移到其它地方，不需要解密，只有打开压缩包时才需解密。一个大文件作为一个数据整体，是不可能瞬间从一台主机转移到其它的主机，传输是一个持续的过程，但不是把文件分割了，因此，如果在传输的过程中意外中断，目标路径中是不会有传输的文件，另外，如果传输的是多个文件，那么，这些文件是按顺序分别传输，如果中间中断，则正在传输的文件会传输失败，但是，之前已经传完的文件传输成功（如果传输的是文件压缩包，那么，不管里面有几个文件，它本身被视为一个文件）。

通常我们看到的 NC、FTP、SCP、NFS 等等，都是可以用来传输文件数据的工具，下面我们将详细介绍主要文件传输工具的特点以及用法。

NETCAT

在网络工具中有“瑞士军刀”的美誉，它功能强大，作为网络工具的同时，它传输文件的能力也不容小觑。

常用参数：

参数	说明
-g <网关>	设置路由器跃程通信网关，最多可设置8个
-G <指向器数目>	设置来源路由指向器，其数值为4的倍数
-i <延迟秒数>	设置时间间隔，以便传送信息及扫描通信端口
-l	使用监听模式，管控传入的资料
-o <输出文件>	指定文件名称，把往来传输的数据以16进制字码倾倒成该文件保存
-p <通信端口>	设置本地主机使用的通信端口
-r	指定本地与远端主机的通信端口
-u	使用UDP传输协议
-v	显示指令执行过程

-w <超时秒数>	设置等待连线的时间
-z	使用0输入/输出模式，只在扫描通信端口时使用
-n	直接使用IP地址，而不通过域名服务器

简单用法举例

1.端口扫描21-24(以IP192.168.2.34为例)。

```
nc -v -w 2 192.168.2.34 -z 21-24
```

nc: connect to 192.168.2.34 port 21 (tcp) failed: Connection refused

Connection to 192.168.2.34 22 port [tcp/ssh] succeeded!

nc: connect to 192.168.2.34 port 23 (tcp) failed: Connection refused

nc: connect to 192.168.2.34 port 24 (tcp) failed: Connection refused

2.从192.168.2.33拷贝文件到192.168.2.34。

在192.168.2.34上：

```
nc -l 1234 > test.txt
```

在192.168.2.33上：

```
nc 192.168.2.34 < test.txt
```

3.用nc命令操作memcached。

存储数据：

```
printf “set key 0 10 6rnresultrn” |nc 192.168.2.34 11211
```

获取数据：

```
printf “get keyrn” |nc 192.168.2.34 11211
```

删除数据：

```
printf “delete keyrn” |nc 192.168.2.34 11211
```

查看状态：

```
```

```
printf “statsrn” |nc 192.168.2.34 11211
```

模拟top命令查看状态：

```
watch “echo stats” |nc 192.168.2.34 11211
```

清空缓存：

```
printf “flush_allrn” |nc 192.168.2.34 11211      #谨慎操作，清空了缓存就没了
```

SCP（安全拷贝 secure copy）

介绍

SCP 命令的用法和 RCP 命令格式非常类似，区别就是 SCP 提供更安全保障，SCP 在需要进行验证时会要求你输入密码或口令，一般推荐使用 SCP 命令，因为它比 RCP 更安全。SCP 命令使用 SSH 来传输数据，并使用与 SSH 相同的认证模式，提供同样的安全保障，SSH 是目前较可靠得，为远程登录会话和其他网络服务提供安全性的协议，利用 SSH 协议可以有效防止远程管理过程中的信息泄露问题。SCP 是基于 SSH 的应用，所以进行数据传输的机器上必须支持 SSH 服务。

特点

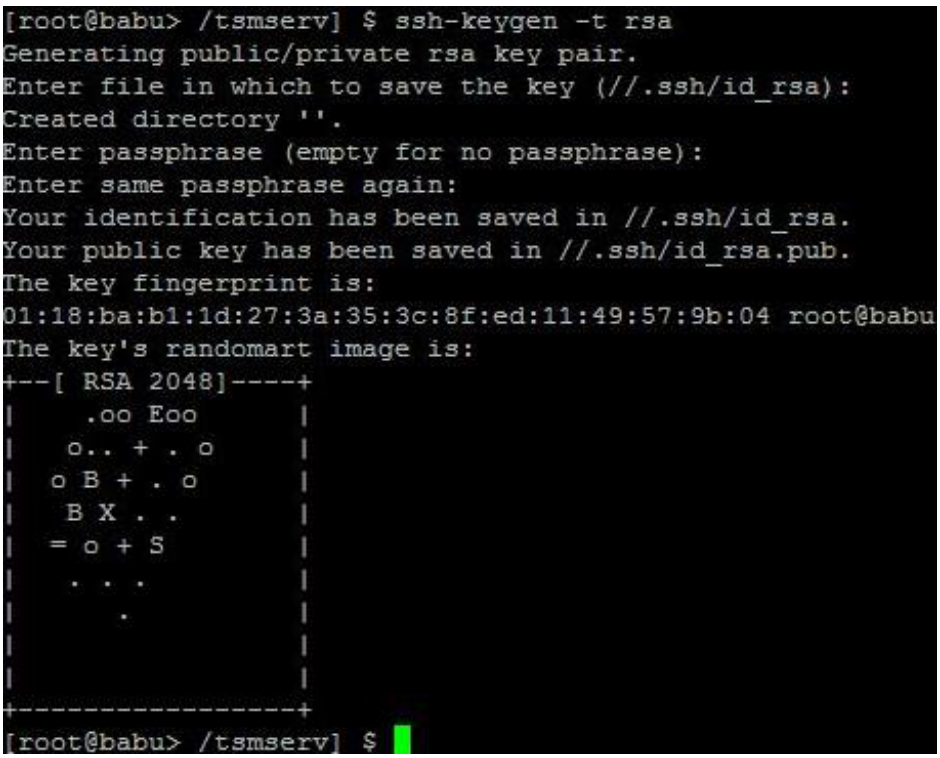
SCP 类似于RCP, 它能够保留一个特定文件系统上的文件属性，能够保留文件属性或者需要递归的拷贝子目录。

SCP它具备更好文件传输保密性。与此同时，付出的代价就是文件传输时需要输入密码而且涉及到 SSH 的一些配置问题，这些都影响其使用的方便性，对于有特定需求的用户，是比较合适的传输工具。

常用示例

使用 SCP 命令，需要输入密码，如果不想每次都输入，可以通过配置 SSH，这样在两台机器间拷贝文件时不需要每次都输入用户名和密码：

生成 RSA 类型的密钥：



上述命令生成 RSA 类型的密钥。在提示密钥的保存路径和密码时，可以直接回车使用默认路径和空密码。这样，生成的公共密钥保存/.ssh/id_rsa.pub，私有密钥保存在 /.ssh/id_rsa 。然后把这个密钥对中的公共密钥的内容复制到要访问的机器上的 /.ssh/authorized_keys 文件中。这样，下次再访问那台机器时，就不用输入密码了。

scp可以在 2个 linux 主机间复制文件

命令基本格式：

```
scp [可选参数] file_source file_target
```

从本地复制到远程（如下四种方式）：

```
scp local_file remote_username@remote_ip:remote_folder
scp local_file remote_username@remote_ip:remote_file
scp local_file remote_ip:remote_folder
scp local_file remote_ip:remote_file
```

注：第1,2个指定了用户名，命令执行后需要再输入密码，第1个仅指定了远程的目录，文件名字不变，第2个指定了文件名。

第3,4个没有指定用户名，命令执行后需要输入用户名和密码，第3个仅指定了远程的目录，文件名字不变，第4个指定了文件名。

从远程复制到本地：

注：从远程复制到本地，只要将从本地复制到远程的命令的后2个参数 调换顺序 即可

```
scp root@www.cumt.edu.cn:/home/root/others/music /home/space/music/i.mp3
scp -r www.cumt.edu.cn:/home/root/others/ /home/space/music/
```

Rsync

Rsync是linux/Unix文件同步和传送工具。用于替代rcp的一个工具，rsync可以通过rsh或ssh使用，也能以daemon模式去运行，在以daemon方式运行时rsync server会开一个873端口，等待客户端去连接。连接时rsync server会检查口令是否相符，若通过口令查核，则可以通过进行文件传输，第一次连通完成时，会把整份文件传输一次，以后则就只需进行增量备份。

安装方式：

注：可以使用每个发行版本自带的安装包管理器安装。

```
sudo apt-get  install  rsync      #在debian、ubuntu 等在线安装方法；
slackpkg install rsync #Slackware 软件包在线安装；
yum install rsync #Fedora、Redhat 等系统安装方法；
```

源码编译安装：

```
wget http://rsync.samba.org/ftp/rsync/src/rsync-3.0.9.tar.gz
tar xf rsync-3.0.9.tar.gz
cd rsync-3.0.9
./configure && make && make install
```

参数介绍：

参数	说明
-v	详细模式输出
-a	归档模式，表示以递归的方式传输文件，并保持所有文件属性不变，相当于使用了组合参数-rlptgoD
-r	对子目录以递归模式处理
-l	保留软链接
-p	保持文件权限
-t	保持文件时间信息
-g	保持文件属组信息
-o	保持文件属主信息
-D	保持设备文件信息
-H	保留硬链结

-S	对稀疏文件进行特殊处理以节省DST的空间
-z	对备份的文件在传输时进行压缩处理

rsync六种不同的工作模式：

1.拷贝本地文件，将/home/coremail目录下的文件拷贝到/cmbak目录下。

```
rsync -avSH /home/coremail/ /cmbak/
```

2.拷贝本地机器的内容到远程机器。

```
rsync -av /home/coremail/ 192.168.11.12:/home/coremail/
```

3.拷贝远程机器的内容到本地机器。

```
rsync -av 192.168.11.11:/home/coremail/ /home/coremail/
```

4.拷贝远程rsync服务器(daemon形式运行rsync)的文件到本地机。

```
rsync -av root@172.16.78.192::www /databack
```

5.拷贝本地机器文件到远程rsync服务器(daemon形式运行rsync)中。当DST路径信息包含”::”分隔符时启动该模式。

```
rsync -av /databack root@172.16.78.192::www
```

6.显示远程机的文件列表。这类似于rsync传输，不过只要在命令中省略掉本地机信息即可。

```
rsync -v rsync://192.168.11.11/data
```

rsync配置文件说明：

```
cat/etc/rsyncd.conf             #内容如下
port = 873 #端口号
uid = nobody #指定当模块传输文件的守护进程UID
gid = nobody #指定当模块传输文件的守护进程GID
use chroot = no #使用chroot到文件系统中的目录中
max connections = 10 #最大并发连接数
strict modes = yes #指定是否检查口令文件的权限
pid file = /usr/local/rsyncd/rsyncd.pid #指定PID文件
lock file = /usr/local/rsyncd/rsyncd.lock #指定支持max connection的锁文件，默认为/var/run/rsyncd.lock
motd file = /usr/local/rsyncd/rsyncd.motd #定义服务器信息的，自己写 rsyncd.motd 文件内容
log file = /usr/local/rsyncd/rsync.log #rsync 服务器的日志
```

```
log format = %t %a %m %f %b
syslog facility = local3
timeout = 300

[conf] #自定义模块
path = /usr/local/nginx/conf #用来指定要备份的目录
comment = Nginx conf
ignore errors #可以忽略一些IO错误
read only = no #设置no，客户端可以上传文件，yes是只读
write only = no #no为客户端可以下载，yes不能下载
hosts allow = 192.168.2.0/24 #可以连接的IP
hosts deny = * #禁止连接的IP
list = false #客户请求时，使用模块列表
uid = root
gid = root
auth users = backup #连接用户名，和linux系统用户名无关系
secrets file = /etc/rsyncd.pass #验证密码文件
```

背景

一般情况下，对数据库的读和写都在同一个数据库服务器中操作时，业务系统性能会降低。为了提升业务系统性能，优化用户体验，可以通过读写分离来减轻主数据库的负载。本篇文章分别从应用层和系统层来介绍读写分离的实现方法。

应用层实现方法：

应用层中直接使用代码实现，在进入Service之前，使用AOP来做出判断，是使用写库还是读库，判断依据可以根据方法名判断，比如说以query、find、get等开头的就走读库，其他的走写库。

优点：

1、多数据源切换方便，由程序自动完成。

2、不需要引入中间件。

3、理论上支持任何数据库。

缺点：

1、由程序员完成，运维参与不到。

2、不能做到动态增加数据源。

系统层实现方法：

方式一：使用DRDS实现

https://help.aliyun.com/document_detail/29681.html

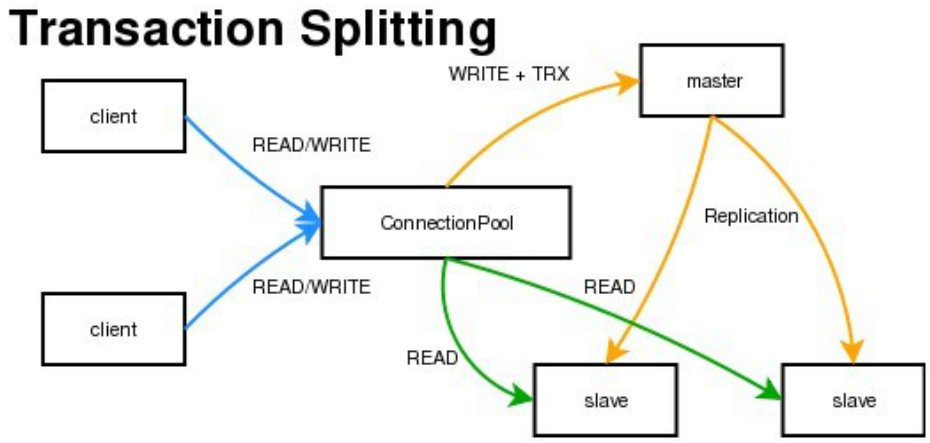
方式二：使用中间件MySQL-proxy实现

本教程使用MySQL-proxy实现读写分离。

MySQL-proxy介绍：

MySQL Proxy是一个处于Client端和MySQL server端之间的简单程序，它可以监测、分析或改变它们的通信。它使用灵活，没有限制，常见的用途包括：负载平衡，故障、查询分析，查询过滤和修改等等。

MySQL-proxy原理：



MySQL Proxy是一个中间层代理，简单的说，MySQL Proxy就是一个连接池，负责将前台应用的连接请求转发给后台的数据库，并且通过使用lua脚本，可以实现复杂的连接控制和过滤，从而实现读写分离和负载平衡。对于应用来说，MySQL Proxy是完全透明的，应用则只需要连接到MySQL Proxy的监听端口即可。当然，这样proxy机器可能成为单点失效，但完全可以使用多个proxy机器做为冗余，在应用服务器的连接池配置中配置到多个proxy的连接参数即可。

优点：

1、源程序不需要做任何改动就可以实现读写分离。

2、动态添加数据源不需要重启程序。

缺点：

1、程序依赖于中间件，会导致切换数据库变得困难。

2、由中间件做了中转代理，性能有所下降。

操作步骤

环境说明：

主库IP：121.40.18.26

从库IP：101.37.36.20

MySQL-proxy代理IP：116.62.101.76

前期准备：

1、新建3台ECS，并安装mysql。

2、搭建主从，必须保证主从数据库数据一致。

主环境

1.修改mysql配置文件。

```
vim /etc/my.cnf

[mysqld]
server-id=202 #设置服务器唯一的id，默认是1
log-bin=mysql-bin # 启用二进制日志
```

从环境

```
[mysqld]
server-id=203
```

2.重启主从服务器中的MySQL服务。

```
/etc/init.d/mysqld restart
```

3.在主服务器上建立帐户并授权slave。

```
mysql -uroot -p95c7586783
grant replication slave on *.* to 'syncms'@'填写slave-IP' identified by '123456';
flush privileges;
```

4.查看主数据库状态。

```
mysql> show master status;
```

```
mysql> show master status;
+------------------+----------+--------------+------------------+-------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB | Executed_Gtid_Set |
+------------------+----------+--------------+------------------+-------------------+
| mysql-bin.000005 |      602 |              |                  |                   |
+------------------+----------+--------------+------------------+-------------------+
1 row in set (0.00 sec)
```

5.配置从数据库。

```
change master to master_host='填写master-IP', master_user='syncms',      master_password='123456',
master_log_file='mysql-bin.000005', master_log_pos=602;
```

6.启动slave同步进程并查看状态。

```
start slave;
show slave status\G
```

```
mysql> show slave status\G
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 116.62.101.35
                  Master_User: syncms
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000007
          Read_Master_Log_Pos: 154
               Relay_Log_File: iZbp17p8l1ul3oj2nztb4kZ-relay-bin.000003
                Relay_Log_Pos: 367
        Relay_Master_Log_File: mysql-bin.000007
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
              Replicate_Do_DB:
          Replicate_Ignore_DB:
           Replicate_Do_Table:
       Replicate_Ignore_Table:
      Replicate_Wild_Do_Table:
  Replicate_Wild_Ignore_Table:
                   Last_Errno: 0
                   Last_Error:
```

7.验证主从同步。

主库上操作：

```
mysql> create database testproxy;
mysql> create table testproxy.test1(ID int primary key,name char(10) not null);
mysql> insert into testproxy.test1 values(1,'one');
mysql> insert into testproxy.test1 values(2,'two');
mysql> select * from testproxy.test1;
```

```
mysql> create database testproxy;
Query OK, 1 row affected (0.01 sec)

mysql> create table testproxy.test1(ID int primary key,name char(10) not null);
Query OK, 0 rows affected (0.07 sec)

mysql> insert into testproxy.test1 values(1,'one');
Query OK, 1 row affected (0.02 sec)

mysql> insert into testproxy.test1 values(2,'two');
Query OK, 1 row affected (0.03 sec)

mysql> select * from testproxy.test1;
+----+------+
| ID | name |
+----+------+
|  1 | one  |
|  2 | two  |
+----+------+
2 rows in set (0.01 sec)
```

从库操作：

从库中查找testproxy.test1表的数据，与主库一致，主从同步成功

```
select * from testproxy.test1;
```

```
mysql> select * from testproxy.test1;
+----+------+
| ID | name |
+----+------+
|  1 | one  |
|  2 | two  |
+----+------+
2 rows in set (0.00 sec)
```

读写分离配置

1.安装MySQL-Proxy。

```
wget https://cdn.mysql.com/archives/mysql-proxy/mysql-proxy-0.8.5-linux-glibc2.3-x86-64bit.tar.gz
mkdir /alidata
tar xvf mysql-proxy-0.8.5-linux-glibc2.3-x86-64bit.tar.gz
mv mysql-proxy-0.8.5-linux-glibc2.3-x86-64bit/ /alidata/mysql-proxy-0.8.5
```

2.环境变量设置。

```
vim /etc/profile                    #加入以下内容
PATH=$PATH:/alidata/mysql-proxy-0.8.5/bin
export $PATH
source /etc/profile #使变量立即生效
mysql-proxy -V
```

```
[root@iZbp1ajyjlht1reyxsfu4xZ ~]# mysql-proxy -V
mysql-proxy 0.8.5
  chassis: 0.8.5
  glib2: 2.16.6
  libevent: 2.0.21-stable
  LUA: Lua 5.1.4
    package.path: /alidata/mysql-proxy-0.8.5/lib/mysql-proxy/lua/?.lua;
    package.cpath: /alidata/mysql-proxy-0.8.5/lib/mysql-proxy/lua/?.so;
-- modules
  proxy: 0.8.5
```

3.读写分离设置。

```
cd /alidata/mysql-proxy-0.8.5/share/doc/mysql-proxy/
vim rw-splitting.lua
```

MySQL Proxy会检测客户端连接, 当连接没有超过min_idle_connections预设值时, 不会进行读写分离默认最小4个(最大8个)以上的客户端连接才会实现读写分离, 现改为最小1个最大2个，便于读写分离的测试，生产环境中，可以根据实际情况进行调整。

调整前：

```
-- connection pool
if not proxy.global.config.rwsplit then
        proxy.global.config.rwsplit = {
                min_idle_connections = 4,
                max_idle_connections = 8,

                is_debug = false
        }
end
```

调整后：

```
-- connection pool
if not proxy.global.config.rwsplit then
        proxy.global.config.rwsplit = {
                min_idle_connections = 1,
                max_idle_connections = 2,

                is_debug = true
        }
end
```

4.将lua管理脚本（admin.lua）复制到读写分离脚本(rw-splitting.lua)所在目录。

```
cp /alidata/mysql-proxy-0.8.5/lib/mysql-proxy/lua/admin.lua /alidata/mysql-proxy-0.8.5/share/doc/mysql-proxy/
```

授权

1.主库中操作授权，因主从同步的原因，从库也会执行。

```
mysql -uroot -p95c7586783
grant all on *.* to 'mysql-proxy'@'填写MySQL Proxy IP' identified by '123456';
flush privileges;
```

2.开启MySQL-Proxy。

```
mysql-proxy --daemon --log-level=debug --log-file=/var/log/mysql-proxy.log --plugins=proxy -b 填写master-IP:3306 -r 填写slave-IP:3306 --proxy-lua-script="/alidata/mysql-proxy-0.8.5/share/doc/mysql-proxy/rw-splitting.lua" --plugins=admin --admin-username="admin" --admin-password="admin" --admin-lua-script="/alidata/mysql-proxy-0.8.5/share/doc/mysql-proxy/admin.lua"
```

3.启动MySQL-Proxy之后，查看端口和相关进程。

```
netstat -tpln
```

```
[root@iZbp1ajyjlht1reyxsfu4xZ ~]# netstat -tpln
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State       PID/Program name
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN      826/sshd
tcp        0      0 0.0.0.0:4040            0.0.0.0:*               LISTEN      22767/mysql-proxy
tcp        0      0 0.0.0.0:4041            0.0.0.0:*               LISTEN      22767/mysql-proxy
```

```
ps -ef | grep mysql
```

```
[root@iZbp1ajyjlht1reyxsfu4xZ ~]# ps -ef | grep mysql
root     22767     1  0 10:59 ?        00:00:00 /alidata/mysql-proxy-0.8.5/libexec/mysql-proxy --daemon --log-level=debug --log-file=/var/log/mysql-proxy.log --plugins=proxy -b 121.40.18.26:3306 -r 101.37.36.20:3306 --proxy-lua-script=/alidata/mysql-proxy-0.8.5/share/doc/mysql-proxy/rw-splitting.lua --plugins=admin --admin-username=admin --admin-password=admin --admin-lua-script=/alidata/mysql-proxy-0.8.5/share/doc/mysql-proxy/admin.lua
root     22794 22602  0 11:02 pts/0    00:00:00 grep --color=auto mysql
```

测试读写分离

1.关闭从复制

```
stop slave;
```

2.MySQL-Proxy上操作，登录mysql-proxy后台管理。

```
mysql  -u  admin -padmin -P 4041 -h MySQL-Proxy-IP
select * from backends; #查看状态
```

```
MySQL [(none)]> select * from backends;
+-------------+-------------------+---------+------+------+-------------------+
| backend_ndx | address           | state   | type | uuid | connected_clients |
+-------------+-------------------+---------+------+------+-------------------+
|           1 | 121.40.18.26:3306 | unknown | rw   | NULL |                 0 |
|           2 | 101.37.36.20:3306 | unknown | ro   | NULL |                 0 |
+-------------+-------------------+---------+------+------+-------------------+
2 rows in set (0.00 sec)
```

第一次连接，会连接到主库上。

```
mysql -umysql-proxy -p123456 -h 116.62.101.76 -P 4040
insert into testproxy.test1 values(3,'three'); #新增一条数据，由于测试需要，关闭了从复制，因此该数据在主库中存在，在从库中不存在
```

```
[root@iZbp1ajyjlht1reyxsfu4xZ ~]# mysql -umysql-proxy -p123456 -h 116.62.101.76 -P 4040
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 6
Server version: 5.7.17-log MySQL Community Server (GPL)

Copyright (c) 2000, 2016, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]> insert into testproxy.test1 values(3,'three');
Query OK, 1 row affected (0.03 sec)

MySQL [(none)]>
```

多开几个连接进行测试，当查询testproxy.test1表的数据显示是从库的数据时，读写分离成功。

```
mysql -umysql-proxy -p123456 -h 116.62.101.76 -P 4040
select * from testproxy.test1;
```

```
MySQL [(none)]> select * from testproxy.test1
    -> ;
+----+------+
| ID | name |
+----+------+
|  1 | one  |
|  2 | two  |
+----+------+
2 rows in set (0.00 sec)

MySQL [(none)]> insert into testproxy.test1 values(9,'nine')
    -> ;
Query OK, 1 row affected (0.02 sec)

MySQL [(none)]> select * from testproxy.test1
    -> ;
+----+------+
| ID | name |
+----+------+
|  1 | one  |
|  2 | two  |
+----+------+
2 rows in set (0.00 sec)
```

简介

FTP 是File Transfer Protocol（文件传输协议）的英文简称，而中文简称为**文传协议**。用于Internet上的控制文件的双向传输。同时，它也是一个应用程序（Application）。基于不同的操作系统有不同的FTP应用程序，而所有这些应用程序都遵守同一种协议以传输文件。

互联网上提供文件存储和访问服务的计算机，他们依照的是FTP协议提供服务！支持FTP协议的服务器就是FTP服务器！FTP协议提供存储和传输服务的一套协议。

下载（Download）和**上传**（Upload）。**下载**文件就是从远程主机拷贝文件至自己的计算机上；**上传**文件就是将文件从自己的计算机中拷贝至远程主机上。用Internet语言来说，用户可通过客户机程序向（从）远程主机上传（下载）文件。

工作原理

FTP采用客户端/服务端的工作模式（C/S结构），通过TCP协议建立客户端和服务器之间的连接，但与其他大多数应用协议不同，FTP协议在客户端和服务端之间建立了两条通信链路，分别是控制链路和数据链路，其中，控制链路负责FTP会话过程中FTP命令的发送和接收，数据链路则负责数据的传输。

FTP会话包含了 2 个通道，控制通道和数据通道，FTP的工作有 2 种方式，一种是主动模式，一种是被动模式，以FTPServer为参照物，主动模式，服务器主动连接客户端传输，被动模式，等待客户端的的连接 。（无论是主动模式还是被动模式，首先的控制通道都是先建立起来的，只是在数据传输模式上的区别）。

本教程主要介绍在 Windows server 2008 R2 和 CentOS 7.2 的系统环境上手动部署。

注意：您可以参考**安全加固方案**对您的FTP服务进行安全加固。

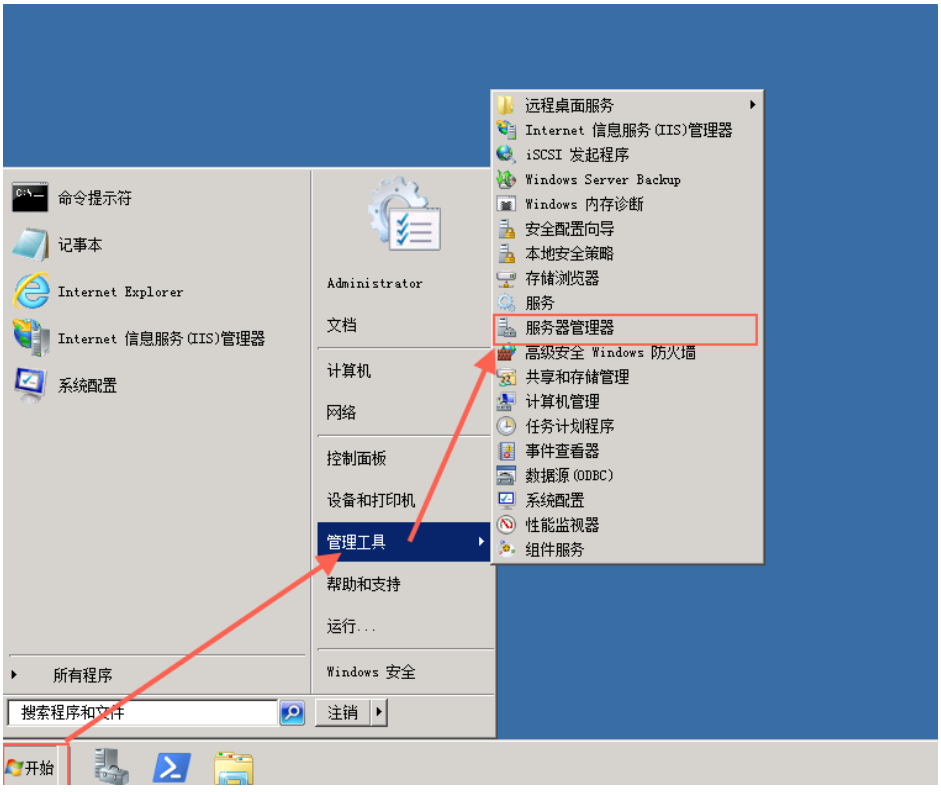
Windows server 2008 R2

安装前准备

选用windows server 2008 R2 企业版 64位中文版的系统，阿里云在公共镜像中提供了该系统镜像，用户可直接在控制台中更换此系统。并通过远程链接进入到系统中。

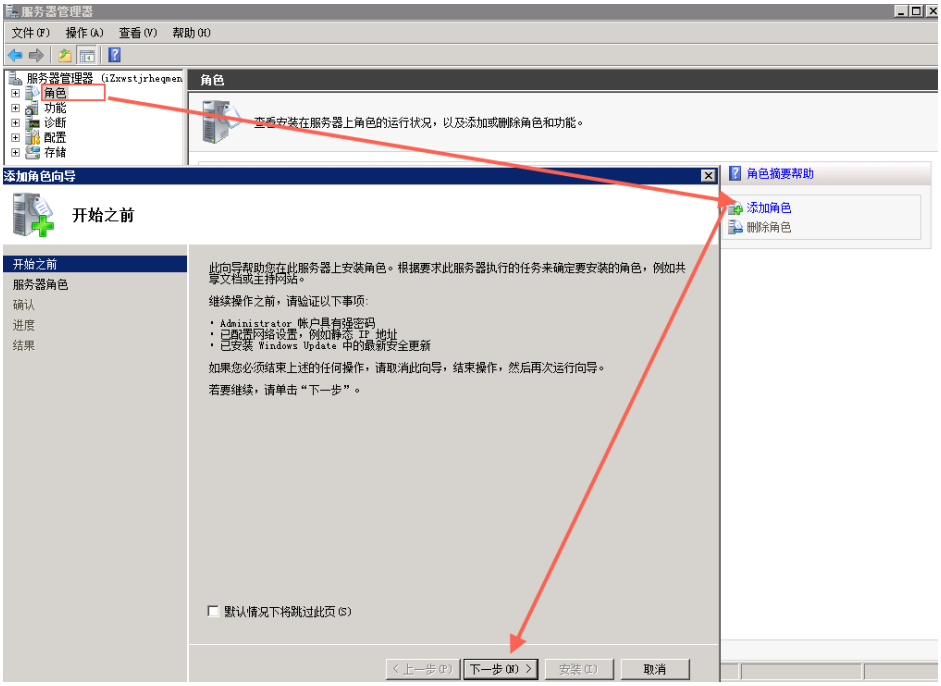
安装FTP服务

开始 > 管理工具 > 服务管理器。



安装 IIS/FTP 角色。

打开服务器管理器，找到添加角色，然后点击，弹出添加角色对话框，选择下一步。



选择Web服务器（IIS），然后选择FTP服务，直到安装完成。

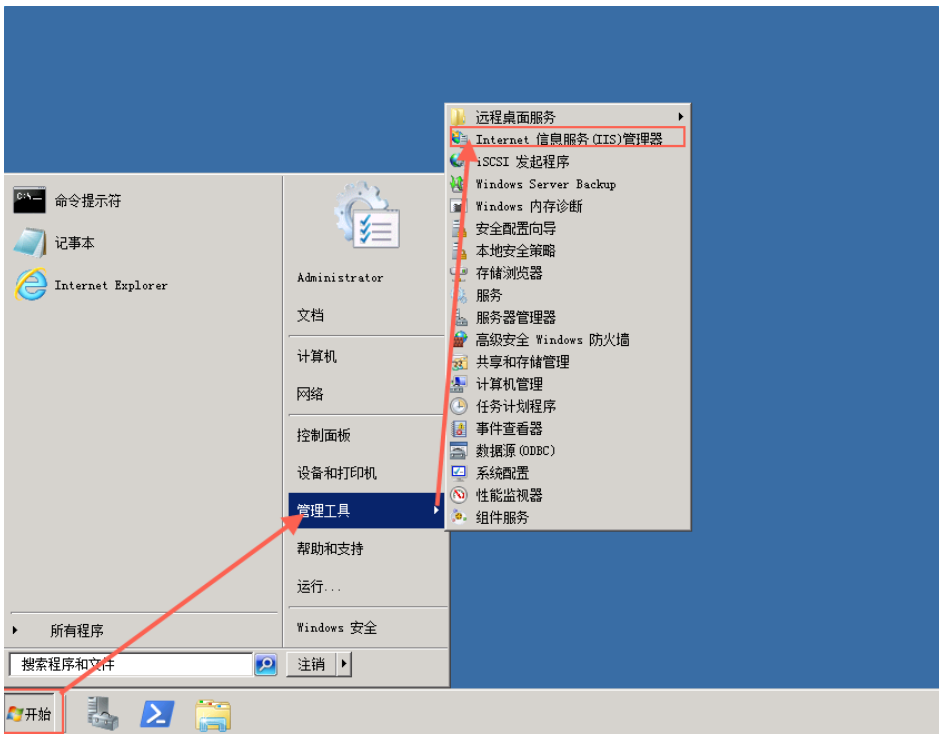

服务器管理器
文件(F) 操作(A) 查看(V) 帮助(H)
服务器管理器 (iZxwstjrheqmen
角色
功能
诊断
配置
存储
角色
查看安装在服务器上角色的运行状况，以及添加或删除角色和功能。
添加角色向导
Web 服务器(IIS)
开始之前
服务器角色
Web 服务器(IIS)
角色服务
确认
进度
结果
Web 服务器 (IIS) 简介
Web 服务器是具有特定软件的计算机，这些软件使服务器可以从客户端计算机接受请求并对这些请求返回响应。Web 服务器支持在 Internet、Intranet 或 Extranet 上共享信息。Web 服务器角色包括 Internet 信息服务 (IIS) 7.0，是一个集成了 IIS 7.0、ASP.NET 和 Windows Communication Foundation 的统一 Web 平台。IIS 7.0 还具有安全性增强、诊断简单和委派管理的特点。
注意事项
使用 Windows 系统资源管理器 (WSRM) 有助于确保等同对待 Web 服务器通信，特别是此计算机上存在多个角色时。
Web 服务器 (IIS) 角色的默认安装包括角色服务的安装，该安装使您能够使用静态内容、进行最小自定义 (如默认的文档和 HTTP 错误)、监视和记录服务器活动以及配置静态内容压缩。
其他信息
Web 服务器 (IIS)概述
IIS 7.0 中的可用角色服务概述
IIS 清单
IIS 中的常见管理任务
WSRM 概述
角色摘要帮助
添加角色
删除角色
< 上一步(P)
下一步(N) >
安装(I)
取消


服务器管理器
文件(F) 操作(A) 查看(V) 帮助(H)
服务器管理器 (iZxwstjrheqmen
角色
功能
诊断
配置
存储
角色
查看安装在服务器上角色的运行状况，以及添加或删除角色和功能。
添加角色向导
选择角色服务
开始之前
服务器角色
Web 服务器(IIS)
角色服务
确认
进度
结果
选择为Web 服务器(IIS)安装的角色服务:
角色服务(R):
IIS 客户端证书映射身份验证
URL 授权
请求筛选
IP 和域限制
性能
静态内容压缩
动态内容压缩
管理工具
IIS 管理控制台
IIS 管理脚本和工具
管理服务
IIS 6 管理兼容性
IIS 6 元数据库兼容性
IIS 6 WMI 兼容性
IIS 6 脚本工具
IIS 6 管理控制台
FTP 服务器
FTP Service
FTP 扩展
IIS 可承载 Web 核心
描述:
FTP 服务器能够使用 FTP 协议在客户端和服务器之间传输文件。用户可以使用 FTP 客户端或支持 FTP 的 Web 浏览器建立 FTP 连接并传输文件。
有关角色服务的详细信息
角色摘要帮助
添加角色
删除角色
< 上一步(P)
下一步(N) >
安装(I)
取消

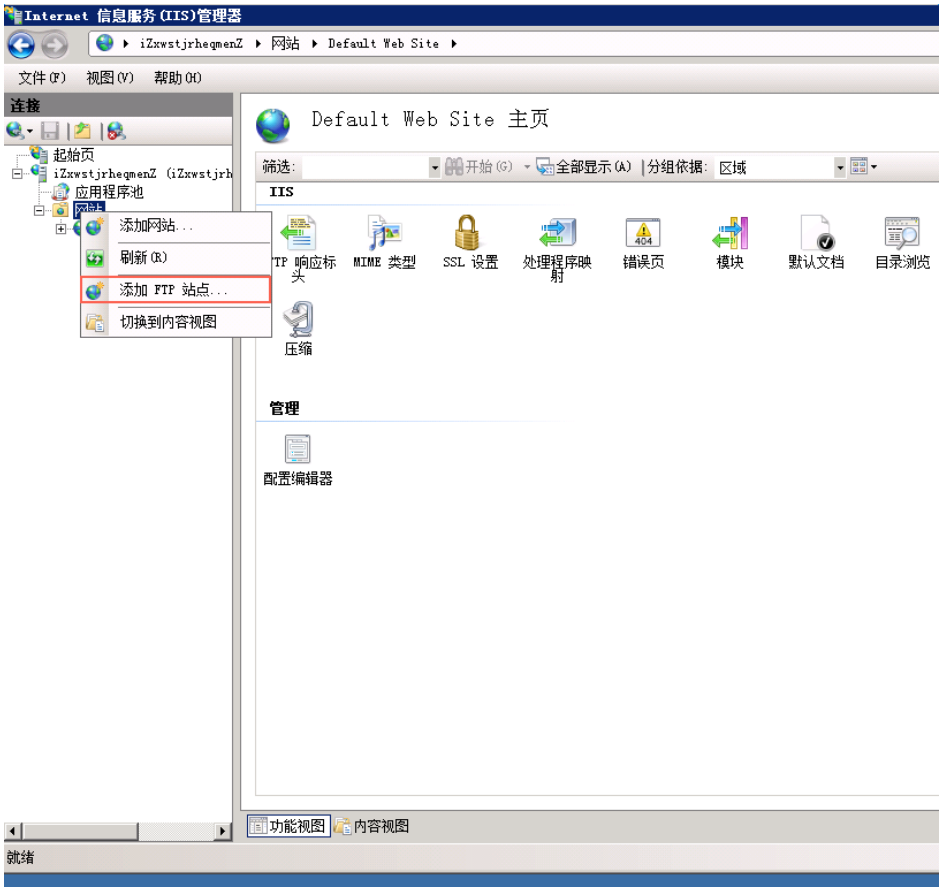

服务器管理器
文件(F) 操作(A) 查看(V) 帮助(H)
服务器管理器 (iZxwstjrheqmen
角色
功能
诊断
配置
存储
角色
查看安装在服务器上角色的运行状况，以及添加或删除角色和功能。
添加角色向导
确认安装选择
开始之前
服务器角色
Web 服务器(IIS)
角色服务
确认
进度
结果
若要安装以下角色、角色服务或功能，请单击“安装”。
2 条提示性消息显示如下
Web 服务器(IIS)
查找有关 Windows 系统资源管理器(WSRM)以及它如何帮助优化 CPU 使用情况的详细信息
Web 服务器
常见 HTTP 功能
静态内容
默认文档
目录浏览
HTTP 错误
健康和诊断
HTTP 日志记录
请求监视
安全性
请求筛选
性能
静态内容压缩
管理工具
IIS 管理控制台
FTP 服务器
FTP Service
FTP 扩展
打印、保存或通过电子邮件发送此信息
< 上一步(P)
下一步(N) >
安装(I)
取消
角色摘要帮助
添加角色
删除角色


服务器管理器
文件(F) 操作(A) 查看(V) 帮助(H)
服务器管理器 (iZxwstjrheqmen
角色
功能
诊断
配置
存储
角色
查看安装在服务器上角色的运行状况，以及添加或删除角色和功能。
添加角色向导
安装结果
开始之前
服务器角色
Web 服务器(IIS)
角色服务
确认
进度
结果
已成功安装以下角色、角色服务或功能:
Web 服务器(IIS)
安装成功
已安装以下角色服务:
Web 服务器
常见 HTTP 功能
静态内容
默认文档
目录浏览
HTTP 错误
健康和诊断
HTTP 日志记录
请求监视
安全性
请求筛选
性能
静态内容压缩
管理工具
IIS 管理控制台
FTP 服务器
FTP Service
FTP 扩展
打印、保存或通过电子邮件发送安装报告
< 上一步(P)
下一步(N) >
关闭(O)
取消
角色摘要帮助
添加角色
删除角色
Web 服务器(IIS) 帮助
转到 Web 服务器(IIS)
添加角色服务
删除角色服务

进入IIS管理器。

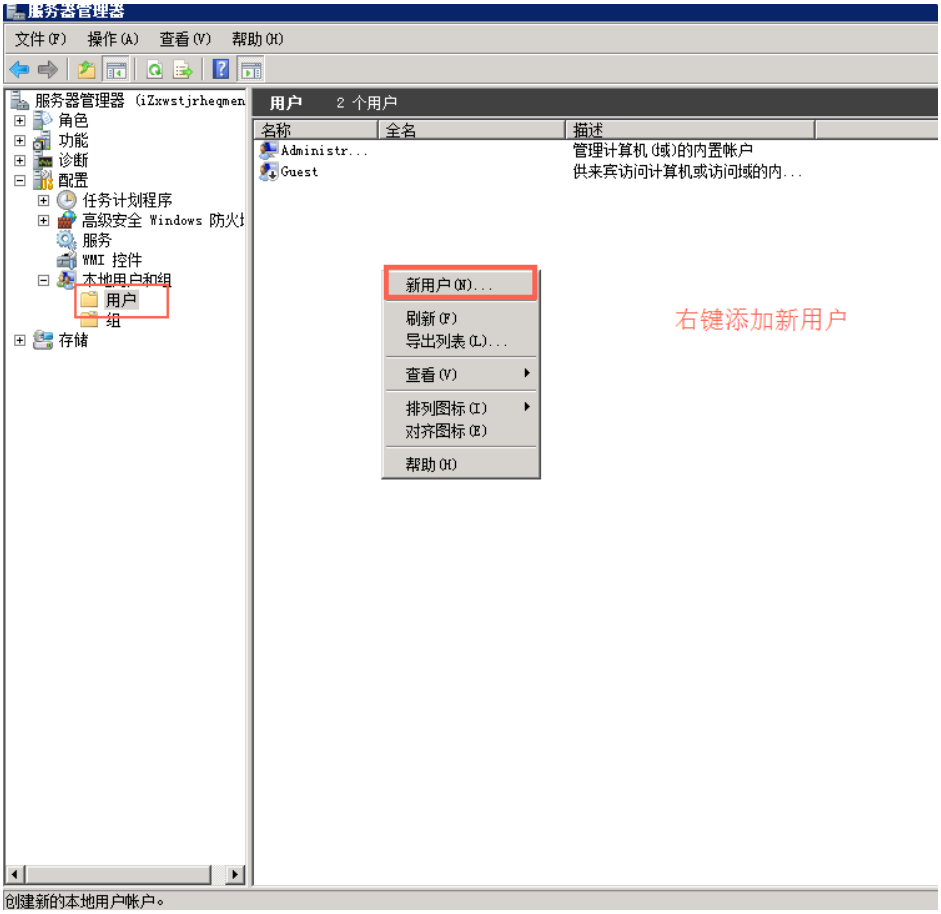


右键网站出现添加**FTP站点**就表示FTP服务安装成功。

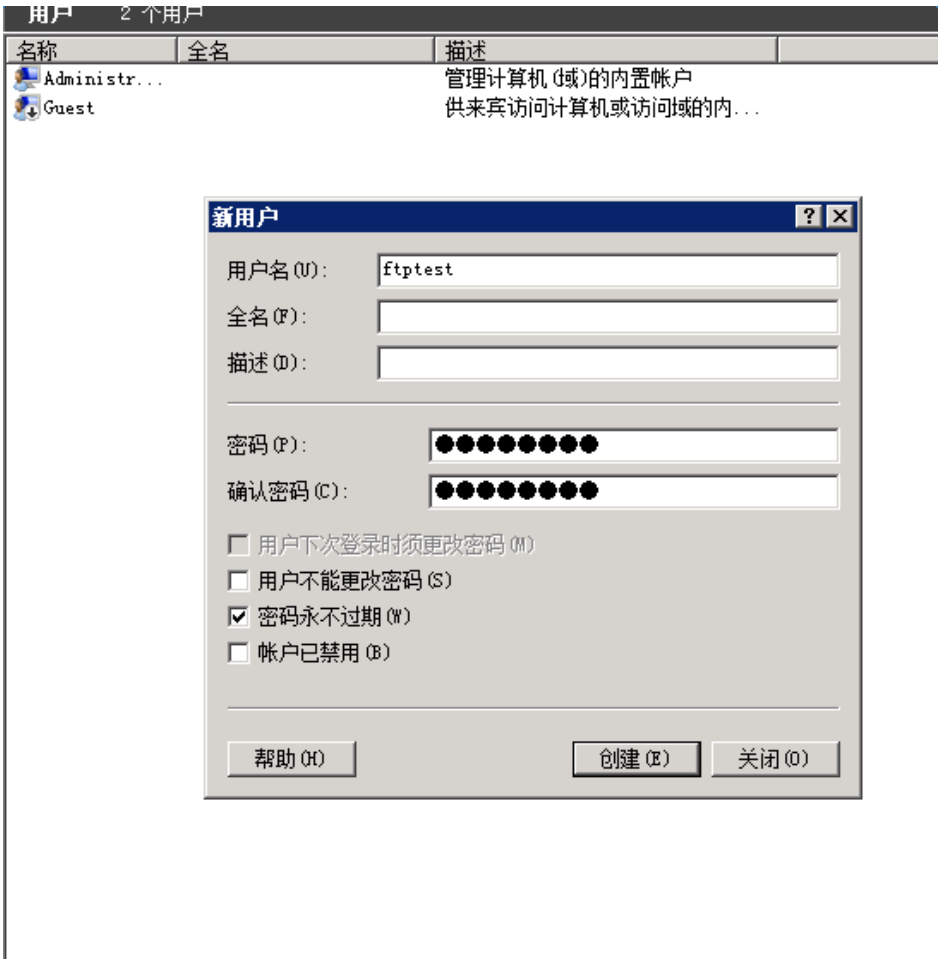


创建Windows用户名和密码，用于FTP使用。

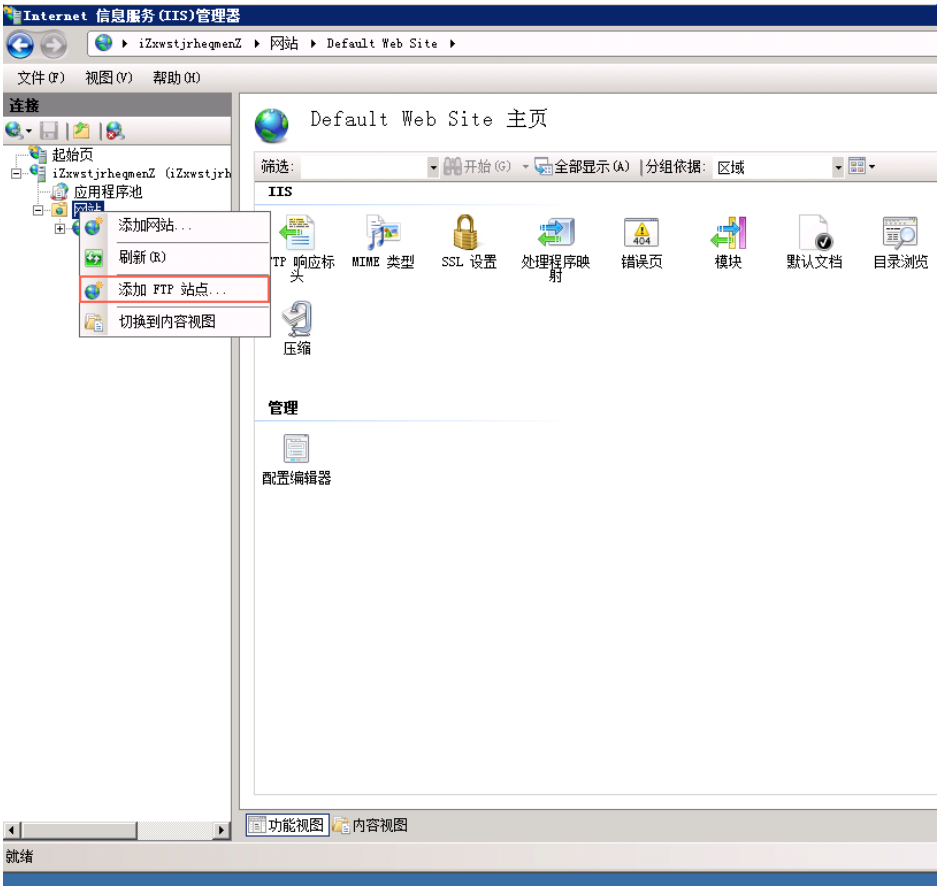
开始>管理工具>服务器管理器，添加用户，如下图：本实例使用ftptest。



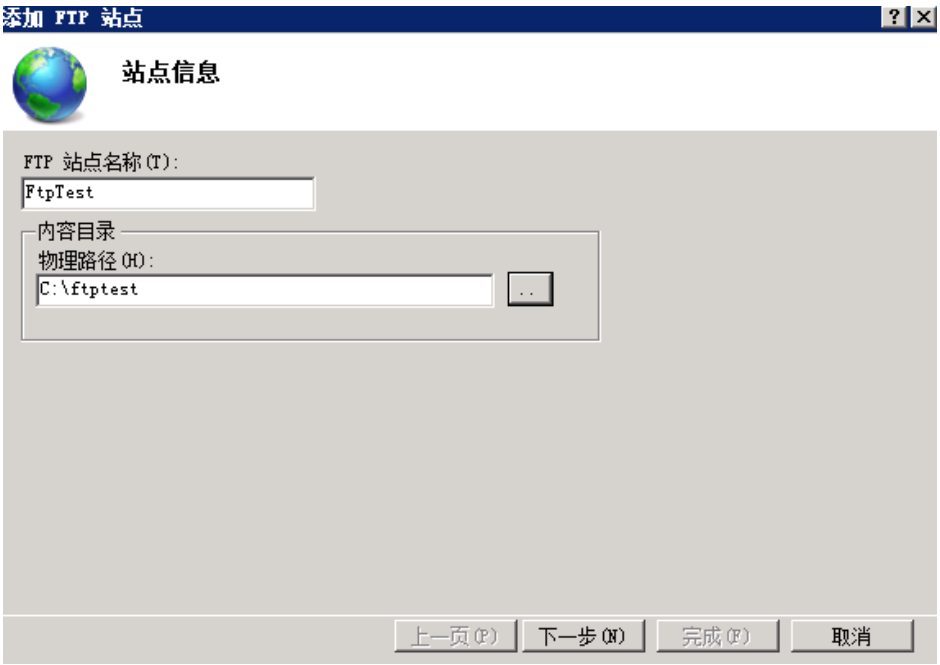
在设置密码时要采用大写字母加小写字母加数字的组合，否则会显示无法通过密码策略。



在服务器磁盘上创建一个供FTP使用的文件夹，创建FTP站点，指定刚刚创建的用户ftptest，赋予读写权限。



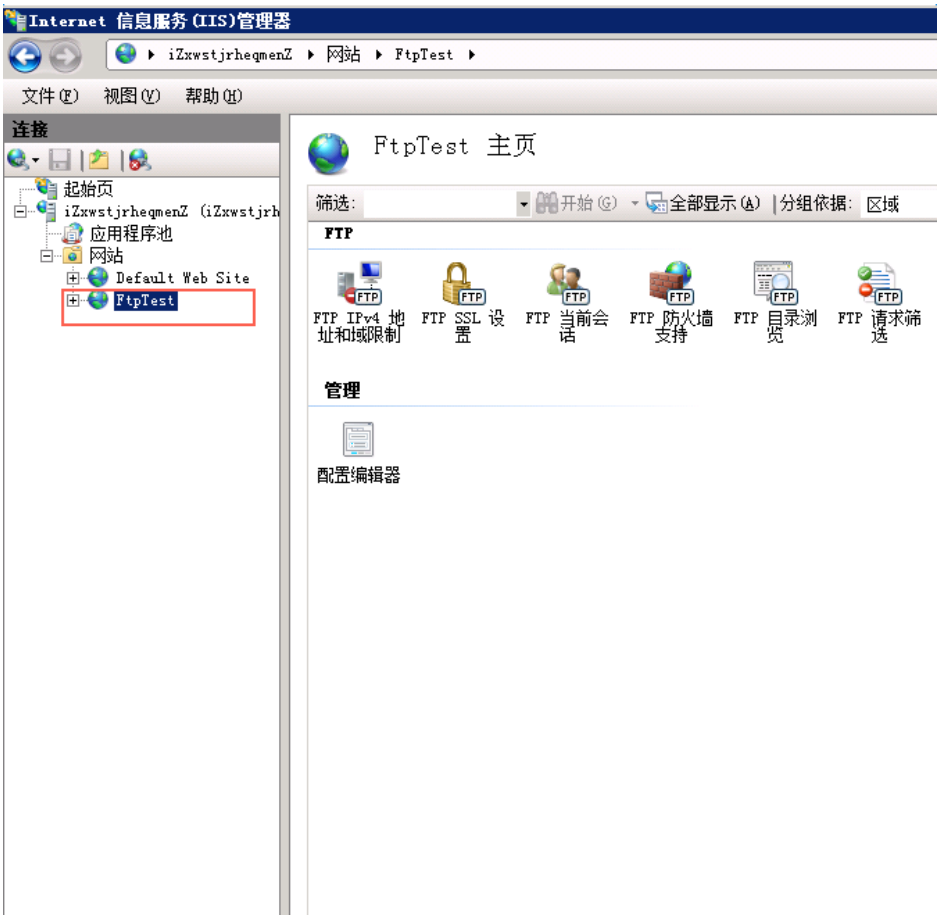
填写FTP站点名称与默认目录。



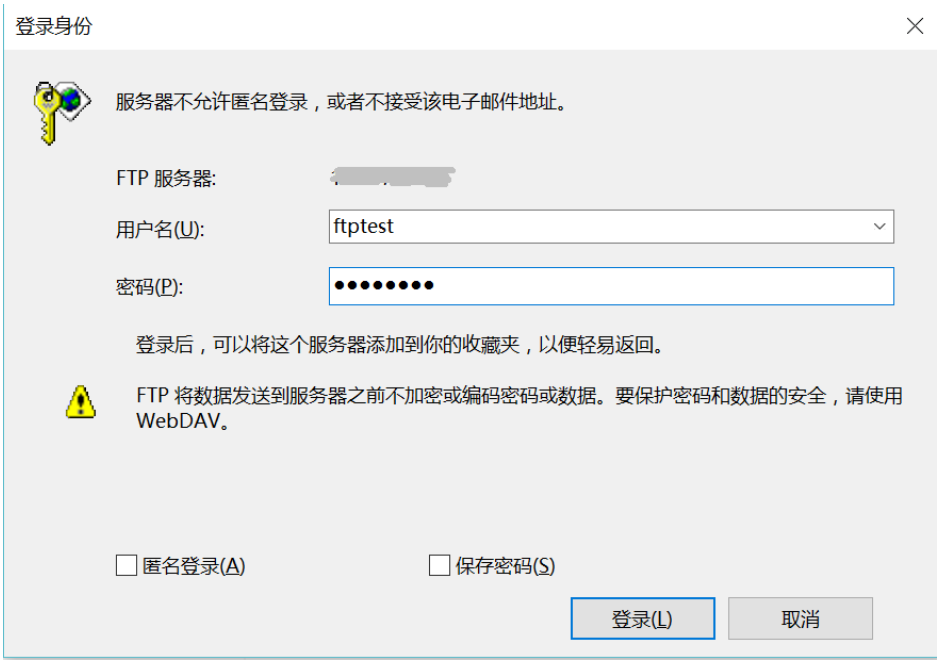
绑定21端口(也可自行设置)。

授权之前创建的ftptest用户允许访问和读写权限。

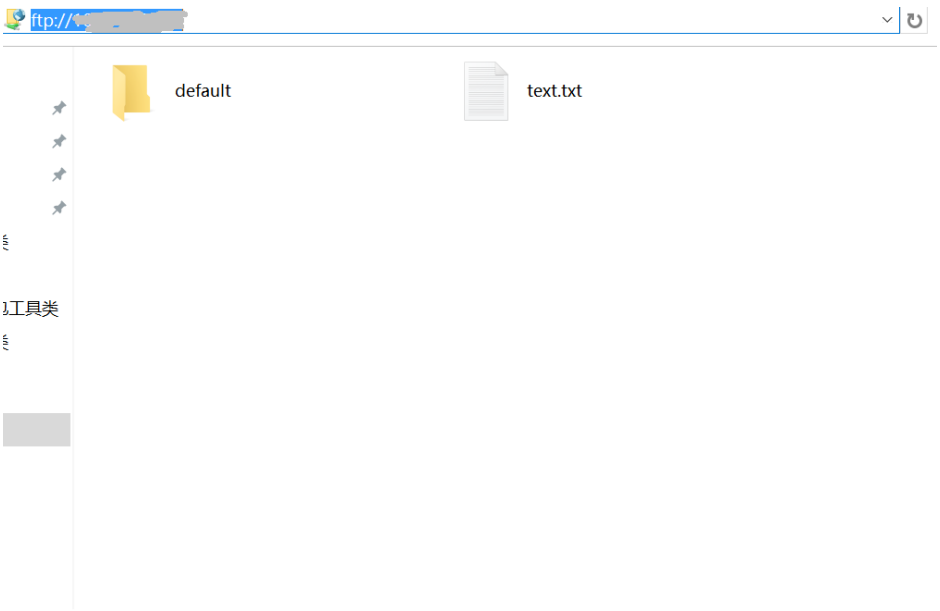
完成后看到设置的FTP站点。



客户端测试。直接使用ftp://服务器ip地址:ftp端口(如果不填端口则默认访问21端口)，如图。弹出输入用户名和密码的对话框表示配置成功，正确的输入用户名和密码后，即可对FTP文件进行相应权限的操作。



登录成功。



CentOS 7.2

安装前准备

选用CentOS 7.2 64位的系统，阿里云在公共镜像中提供了该系统镜像，用户可直接在控制台中更换此系统。并通过远程链接进入到系统中。

vsftpd是linux下的一款小巧轻快，安全易用的FTP服务器软件，是一款在各个Linux发行版中最受推崇的FTP服务器软件。

安装vsftpd，直接yum 安装就可以了

```
yum install -y vsftpd
```

```
[root@iZbp1g1kolvxh5k8l00z6cZ ~]# yum install -y vsftpd
```

出现下图表示安装成功。

![图片21](http://docs-aliyun.cn-hangzhou.oss.aliyun-inc.com/assets/pic/51998/cn_zh/1490260739524/%E5%9B%BE%E7%89%8721.png)

相关配置文件：

```
cd /etc/vsftpd
```

```
[root@iZbp1g1kolvxh5k8l00z6cZ ~]# cd /etc/vsftpd/
[root@iZbp1g1kolvxh5k8l00z6cZ vsftpd]# ls
ftpusers  user_list  vsftpd.conf  vsftpd_conf_migrate.sh
[root@iZbp1g1kolvxh5k8l00z6cZ vsftpd]#
```

/etc/vsftpd/vsftpd.conf //主配置文件，核心配置文件

/etc/vsftpd/ftpusers //黑名单，这个里面的用户不允许访问FTP服务器

/etc/vsftpd/user_list //白名单，允许访问FTP服务器的用户列表

启动服务。

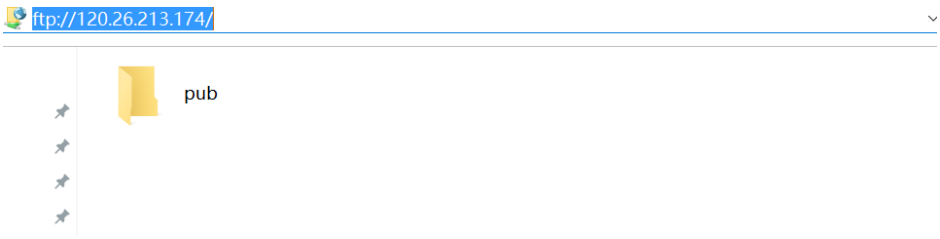
systemctl enable vsftpd.service //设置开机自启动

systemctl start vsftpd.service //启动ftp服务

netstat -antup | grep ftp //查看ftp服务端口

```
[root@iZbp1g1kolvxh5k8l00z6cZ vsftpd]# systemctl enable vsftpd.service
[root@iZbp1g1kolvxh5k8l00z6cZ vsftpd]# systemctl start vsftpd.service
[root@iZbp1g1kolvxh5k8l00z6cZ vsftpd]# netstat -antup | grep ftp
tcp6       0      0 :::21                   :::*                    LISTEN      9379/vsftpd
```

登录ftp服务器。



匿名ftp的基本配置

使用匿名FTP，用户无需输入用户名密码即可登录FTP服务器，vsftpd安装后默认开启了匿名ftp的功能，用户无需额外配置即可使用匿名登录ftp服务器。

匿名ftp的配置在/etc/vsftpd/vsftpd.conf中设置。

anonymous_enable=YES //默认即为YES

```
# Allow anonymous FTP? (Beware - allowed by default if you comment this out).
anonymous_enable=YES
```

这个时候任何用户都可以通过匿名方式登录ftp服务器，查看并下载匿名账户主目录下的各级目录和文件，但是不能上传文件或者创建目录。

为了演示效果，我们安装一个lftp软件。

```
yum -y install lftp              //安装lftp
```

```
Running transaction
  Installing : lftp-4.4.8-8.el7_3.2.x86_64                                  1/1
  Verifying  : lftp-4.4.8-8.el7_3.2.x86_64                                  1/1

Installed:
  lftp.x86_64 0:4.4.8-8.el7_3.2

Complete!
[root@iZbp1g1kolvxh5k8l00z6cZ vsftpd]# yum -y install lftp
```

利用lftp 公网ip连接到ftp服务器，可以看到只能查看和下载，不能进行上传操作

```
lftp 公网ip                    #连接到ftp服务器
cd pub/ #切换到pub目录
put /etc/issue #上传文件
get test.1 #下载文件
```

```
[root@iZbp1g1kolvxh5k8l00z6cZ ~]# lftp 120.26.213.174
lftp 120.26.213.174:~> ls
drwxr-xr-x    2 0        0            4096 Mar 22 06:13 pub
lftp 120.26.213.174:/> cd pub/
lftp 120.26.213.174:/pub> ls
-rw-r--r--    1 0        0               6 Mar 22 06:13 test.1
lftp 120.26.213.174:/pub> put /etc/issue
put: Access failed: 550 Permission denied. (issue)      拒绝上传
lftp 120.26.213.174:/pub> get test.1
6 bytes transferred                                     可以下载
lftp 120.26.213.174:/pub>
```

匿名ftp的其他设置

出于安全方面的考虑，vsftpd在默认情况下不允许用户通过匿名FTP上传文件，创建目录等更改操作，但是可以

修改vsftpd.conf配置文件的选项，可以赋予匿名ftp更多的权限。

允许匿名ftp上传文件。

修改/etc/vsftpd/vsftpd.conf。

write_enable=YES

anon_upload_enable=YES

```
anonymous_enable=YES
#
# Uncomment this to allow local users to log in.
# When SELinux is enforcing check for SE bool ftp_home_dir
local_enable=YES
#
# Uncomment this to enable any form of FTP write command.
write_enable=YES
#
# Default umask for local users is 077. You may wish to change this to 022,
# if your users expect that (022 is used by most other ftpd's)
local_umask=022
#
# Uncomment this to allow the anonymous FTP user to upload files. This only
# has an effect if the above global write enable is activated. Also, you will
# obviously need to create a directory writable by the FTP user.
# When SELinux is enforcing check for SE bool allow_ftpd_anon_write, allow_ftpd_
anon_upload_enable=YES
                                                                        29,1
```

更改/var/ftp/pub目录的权限，为ftp用户添加写权限，并重新加载配置文件。

```
 chmod o+w /var/ftp/pub/                #更改/var/ftp/pub目录的权限
systemctl restart vsftpd.service #重启ftp服务
```

```
[root@iZbp1g1kolvxh5k8l00z6cZ ~]# chmod o+w /var/ftp/pub/
[root@iZbp1g1kolvxh5k8l00z6cZ ~]# systemctl restart vsftpd.service
[root@iZbp1g1kolvxh5k8l00z6cZ ~]#
```

测试。

```
lftp 120.26.213.174:/pub> ls
-rw-r--r--    1 0        0               6 Mar 22 06:13 test.1
lftp 120.26.213.174:/pub> put /etc/issue
23 bytes transferred
lftp 120.26.213.174:/pub> ls
-rw-------    1 14       50             23 Mar 22 07:05 issue
-rw-r--r--    1 0        0               6 Mar 22 06:13 test.1
lftp 120.26.213.174:/pub>
```

配置本地用户登录

本地用户登录就是指使用Linux操作系统中的用户账号和密码登录ftp服务器，vsftp安装后默只支持匿名ftp登录，用户如果试图使用Linux操作系统中的账号登录服务器，将会被vsftpd拒绝

创建ftptest用户。

useradd ftptest #创建ftptest用户 passwd ftptest #修改ftptest用户密码

```
[root@iZbp1g1kolvxh5k8l00z6cZ ~]# useradd ftptest
[root@iZbp1g1kolvxh5k8l00z6cZ ~]# passwd ftptest
Changing password for user ftptest.
New password:
BAD PASSWORD: The password fails the dictionary check - it is too simplistic/systematic
Retype new password:
passwd: all authentication tokens updated successfully.
[root@iZbp1g1kolvxh5k8l00z6cZ ~]#
```

修改/etc/vsftpd/vsftpd.conf。

anonymous enable=NO

local_enable=YES

```
# Allow anonymous FTP? (Beware - allowed by default if you comment this out).
anonymous_enable=NO
#
# Uncomment this to allow local users to log in.
# When SELinux is enforcing check for SE bool ftp_home_dir
local_enable=YES
#
# Uncomment this to enable any form of FTP write command.
write_enable=YES
#
# Default umask for local users is 077. You may wish to change this to 022,
# if your users expect that (022 is used by most other ftpd's)
local_umask=022
#
```

还是通过lftp连接到ftp服务器。

```
[root@iZbp1g1kolvxh5k8l00z6cZ ~]# lftp ftptest@120.26.213.174
Password:
lftp ftptest@120.26.213.174:~> ls
lftp ftptest@120.26.213.174:~> mkdir test
mkdir ok, `test' created
lftp ftptest@120.26.213.174:~> ls
drwxr-xr-x    2 1000     1000         4096 Mar 22 07:17 test
lftp ftptest@120.26.213.174:~> put /etc/issue
23 bytes transferred
```

另外简单介绍下vsftpd.conf的配置文件参数说明。

```
cat /etc/vsftpd/vsftpd.conf
```

用户登陆控制

参数	说明
anonymous_enable=YES	接受匿名用户
no_anon_password=YES	匿名用户login时不询问口令
anon_root=(none)	匿名用户主目录
local_enable=YES	接受本地用户
local_root=(none)	本地用户主目录

用户权限控制

参数	说明
write_enable=YES	可以上传(全局控制)
local_umask=022	本地用户上传文件的umask
file_open_mode=0666	上传文件的权限配合umask使用
anon_upload_enable=NO	匿名用户可以上传
anon_mkdir_write_enable=NO	匿名用户可以建目录
anon_other_write_enable=NO	匿名用户修改删除
chown_username=lightwiter	匿名上传文件所属用户名

相关连接

更多开源软件尽在云市场：https://market.aliyun.com/software

RedHat/CentOS使用 **yum update** 更新时，默认会升级内核。但有些服务器硬件在升级内核后，新的内核可能会认不出某些硬件，要重新安装驱动，很麻烦。所以在生产环境中不要轻易的升级内核，除非您确定升级内核后不会出现麻烦的问题。

如果使用yum update更新时不升级内核，有两种方法：

方法一

直接在yum的命令后面加参数，这个命令只生效一次：

```
# yum update --exclude=kernel*
```

方法二

修改yum命令的配置文件，永久生效。

这里以 CentOS 6.6 为例来进行说明：

1、首先检查内核版本以及系统版本。

```
[root@localhost ~]# uname -r
2.6.32-504.el6.x86_64
[root@localhost ~]# cat /etc/issue
CentOS release 6.6 (Final)
```

```
Kernel \r on an \m
```

2、将配置文件保存备份。

```
[root@localhost ~]# cp /etc/yum.conf /etc/yum.conf.bak
```

3、编辑/etc/yum.conf文件。

```
[root@localhost ~]# vi /etc/yum.conf
```

```
[main]
cachedir=/var/cache/yum/$basearch/$releasever
keepcache=0
debuglevel=2
logfile=/var/log/yum.log
exactarch=1
obsoletes=1
gpgcheck=1
plugins=1
installonly_limit=5
bugtracker_url=http://bugs.centos.org/set_project.php?project_id=19&ref=http://b
ugs.centos.org/bug_report_page.php?category=yum
distroverpkg=centos-release

#  This is the default, if you make this bigger yum won't see if the metadata
# is newer on the remote and so you'll "gain" the bandwidth of not having to
# download the new metadata and "pay" for it by yum not having correct
# information.
#  It is esp. important, to have correct metadata, for distributions like
# Fedora which don't keep old packages around. If you don't like this checking
# interupting your command line usage, it's much better to have something
# manually check the metadata once an hour (yum-updatesd will do this).
# metadata_expire=90m

# PUT YOUR REPOS HERE OR IN separate files named file.repo
# in /etc/yum.repos.d
~
~
```

4、在[main]的后面加入如下内容：

```
exclude=kernel*
```

```
[main]
cachedir=/var/cache/yum/$basearch/$releasever
keepcache=0
debuglevel=2
logfile=/var/log/yum.log
exactarch=1
obsoletes=1
gpgcheck=1
plugins=1
installonly_limit=5
bugtracker_url=http://bugs.centos.org/set_project.php?project_id=19&ref=http:/
/bugs.centos.org/bug_report_page.php?category=yum
distroverpkg=centos-release
exclude=kernel*
```

5、按下Esc，输入下面命令进行保存：wq。

6、使用 yum update更新。

```
[root@localhost yum.repos.d]# yum update
```

7、等到yum update更新完成之后重启电脑，再来检查内核版本。


```
[root@localhost ~]# uname -r
2.6.32-504.el6.x86_64
[root@localhost ~]# cat /etc/issue
CentOS release 6.8 (Final)
Kernel \r on an \m
```

我们可以看到yum update后系统版本升级了，内核版本没有升级。如果同时要禁止升级系统，则在其 [main] 部分末尾增加 “exclude=kernel *centos-release*” 。

简介

Active Directory（简称AD，即 “ 活动目录 “的意思），是微软下面的核心组件，其主要优势是实现高效管理（例如，批量管理用户，部署应用，更新补丁等等），而且微软很多的套件（Exchange，故障转移群集）也是需要域环境支持。

安装

安装之前我们介绍域里面的几个常见名词以及必要条件。

名词解释

Domain Controllers（DC）域控制器

Organizational Unit（OU）组织单位

Distinguished name（DN）识别名

Canonical Name（CN）正式名称

必要条件

安装者必须拥有管理员权限。

安装分区为NTFS分区。

需要DNS支持。

需要TCP/IP 支持（最好有固定IP，任何服务器都应该使用固定IP，防止重启后IP地址发生变化，我这里服务器网络采用的是阿里云的VPC网络，手动修改IP会导致IP失效，如果想修改IP，可以通过控制

台修改）。

环境

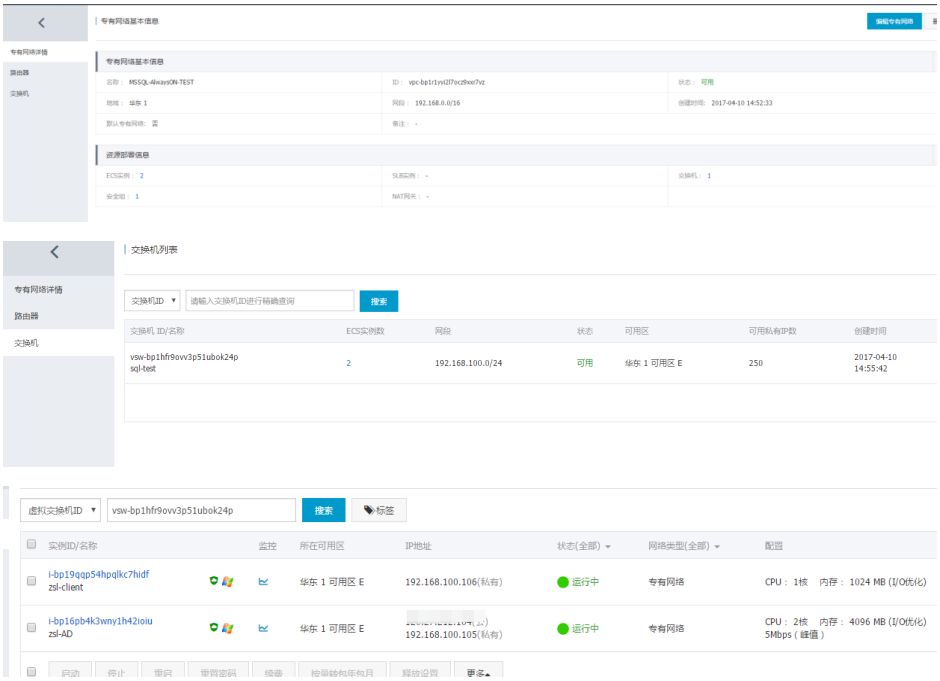
网络采用的是阿里云的VPC网络， 192.168.100.0/24 网关默认。

域名

lyonz.com

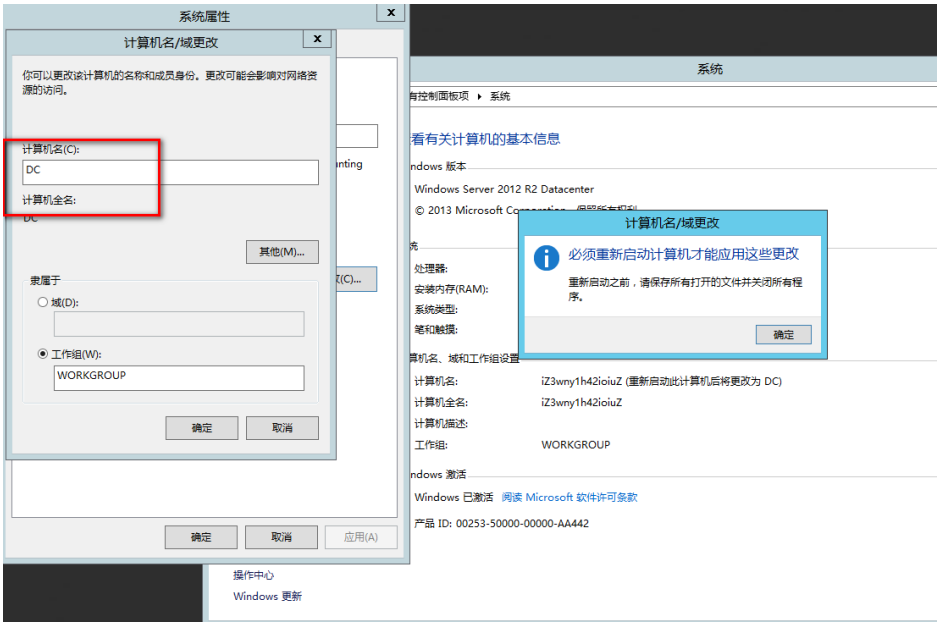
DC: 192.168.100.105

Client: 192.168.100.106 (需要加入域的客户机)

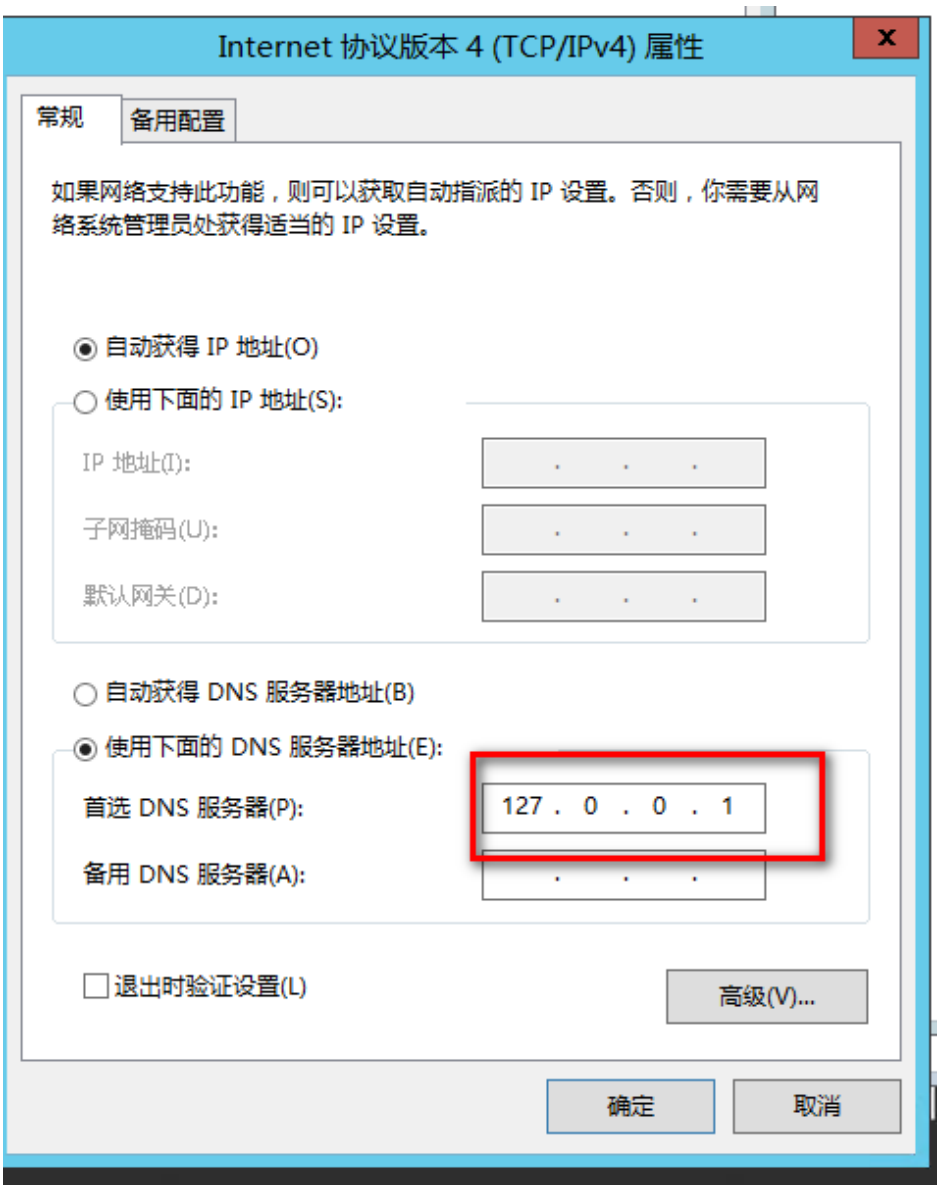


修改DC 的基本信息

修改DC主机名



修改DC 的DNS（将DNS地址指向自己的IP）

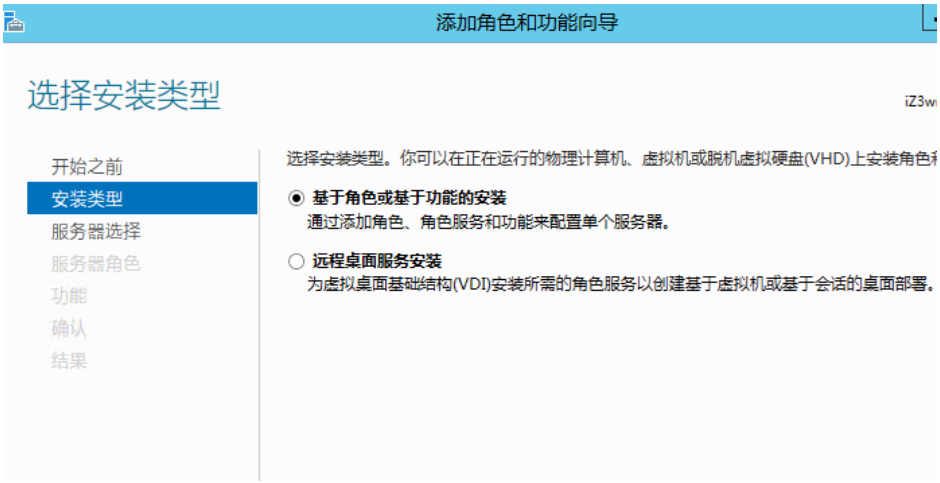


注意

这里不要手动修改服务器的IP地址（手动修改服务器IP不会生效，也无需担心服务器IP会重启发生改变），如果要修改请在控制台操作。

开始安装

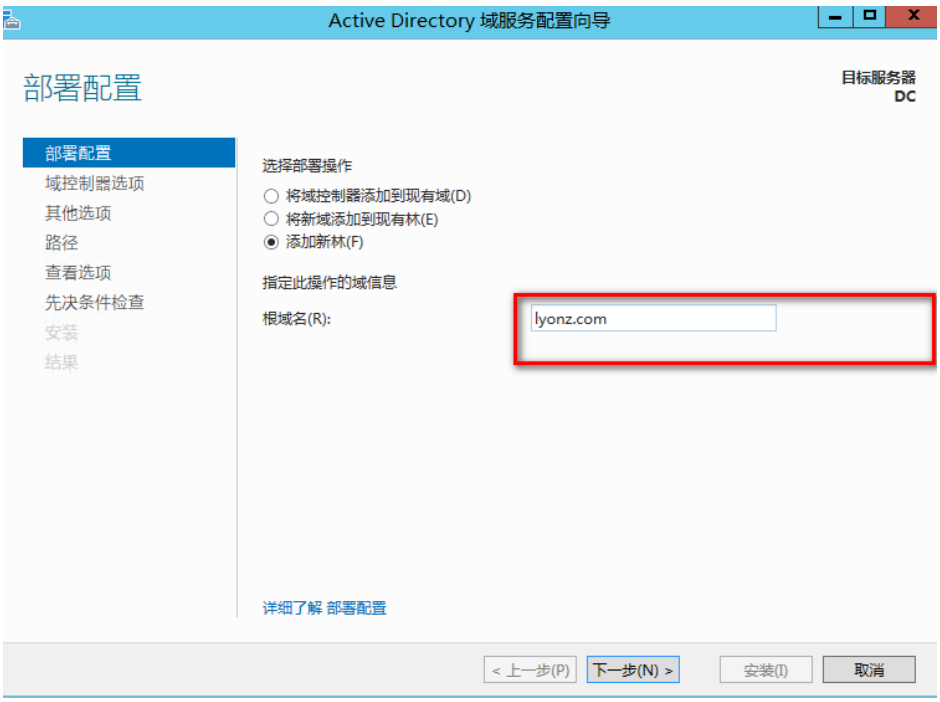

120.27.212.104
服务器管理器 › 仪表板
仪表板
本地服务器
所有服务器
文件和存储服务
欢迎使用服务器管理器
快速启动(Q)
新增功能(W)
了解详细信息(L)
1 配置此本地服务器
2 添加角色和功能
3 添加要管理的其他服务器
4 创建服务器组
5 将此服务器连接到云服务
角色和服务器组
角色: 1 | 服务器组: 1 | 服务器总数: 1
文件和存储服务 1
本地服务器 1
所有服务器 1

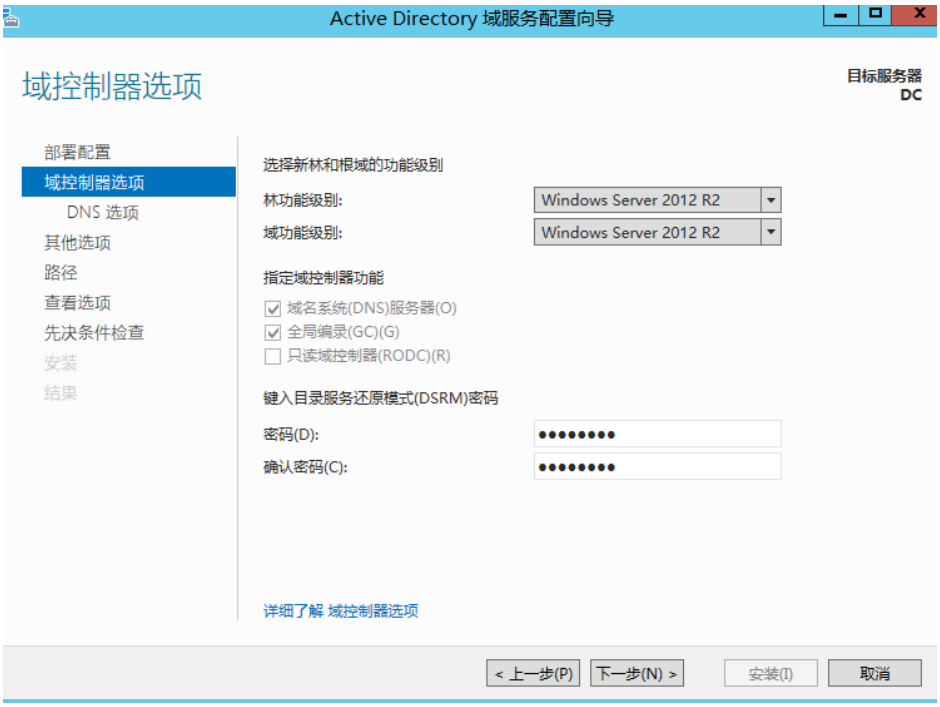

添加角色和功能向导
选择安装类型
开始之前
安装类型
服务器选择
服务器角色
功能
确认
结果
选择安装类型。你可以在正在运行的物理计算机、虚拟机或脱机虚拟硬盘(VHD)上安装角色
基于角色或基于功能的安装
通过添加角色、角色服务和功能来配置单个服务器。
远程桌面服务安装
为虚拟桌面基础结构(VDI)安装所需的角色服务以创建基于虚拟机或基于会话的桌面部署。

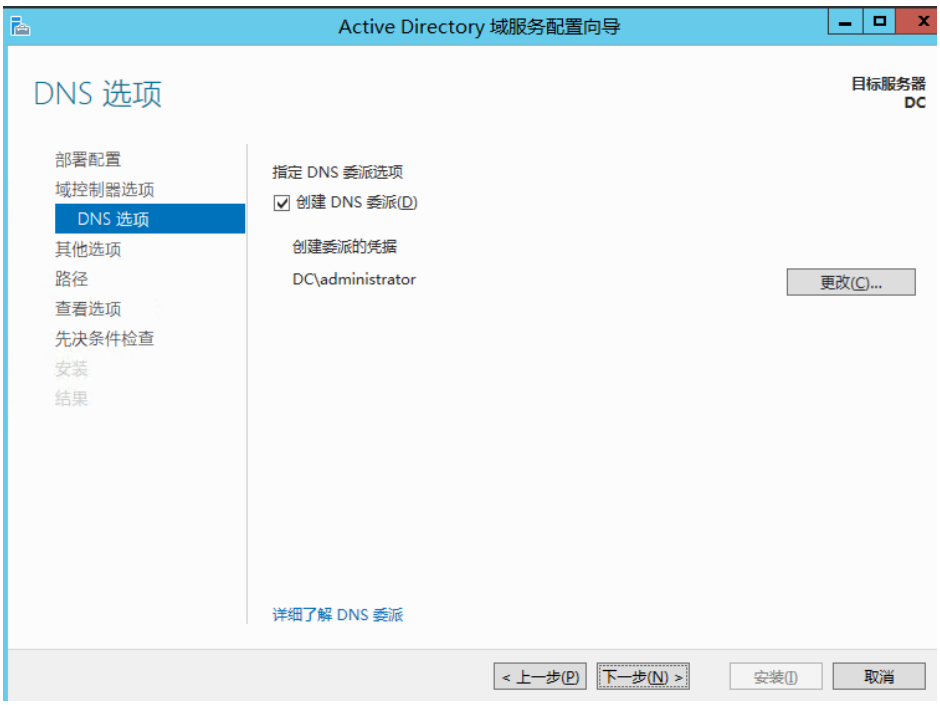

添加角色和功能向导
选择目标服务器
目标服务器
iZ3wny1h42ioiuZ
开始之前
安装类型
服务器选择
服务器角色
功能
确认
结果
选择要安装角色和功能的服务器或虚拟硬盘。
从服务器池中选择服务器
选择虚拟硬盘
服务器池
筛选器:
名称
IP 地址
操作系统
iZ3wny1h42ioiuZ
169.254.60.17...
Microsoft Windows Server 2012 R2 Datacenter


添加角色和功能向导
选择服务器角色
开始之前
安装类型
服务器选择
服务器角色
功能
AD DS
DNS 服务器
确认
结果
选择要安装在所选服务器上的一个或多个角色。
角色
描述
Active Directory Federation Services
Active Directory Rights Management Services
Active Directory 轻型目录服务
Active Directory 域服务
Active Directory 证书服务
DHCP 服务器
DNS 服务器
Hyper-V
Web 服务器(IIS)
Windows Server Essentials 体验
Windows Server 更新服务
Windows 部署服务
传真服务器
打印和文件服务
< 上一步(P)
下一步(N) >

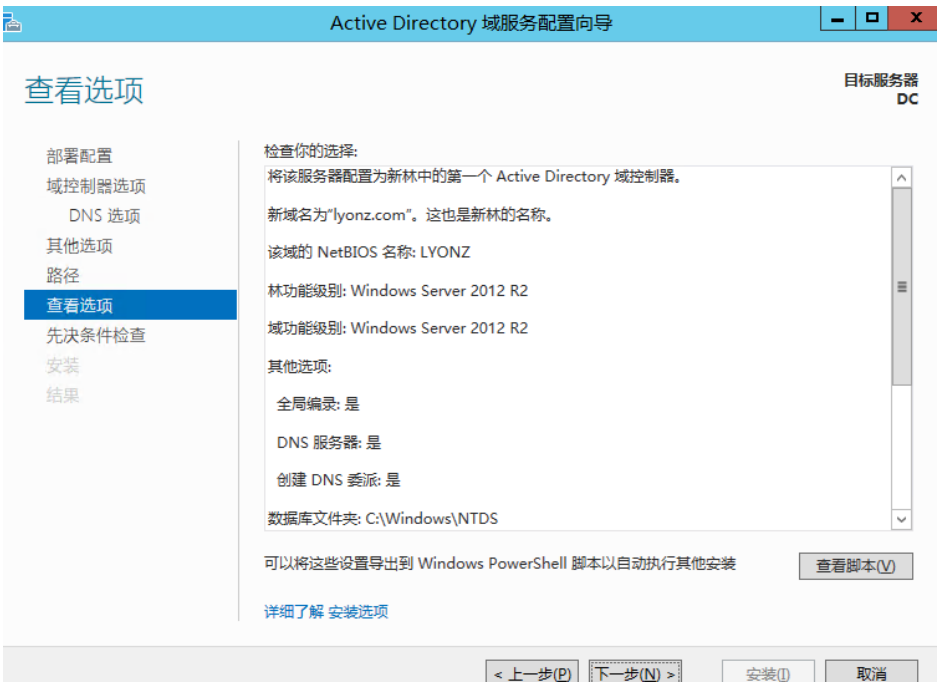

服务器管理器
仪表板
工具(T)
视图(V)
帮助(H
仪表板
本地服务器
所有服务器
AD DS
DNS
文件和存储服务
部署后配置
任务
DC 中的 Active Directory 域服务 所需的配置
将此服务器提升为域控制器
任务详细信息
地服务器
2 添加角色和功能
3 添加要管理的其他服务器
4 创建服务器组
5 将此服务器连接到云服务
隐藏
新增功能(W)
了解详细信息(L)
角色和服务器组

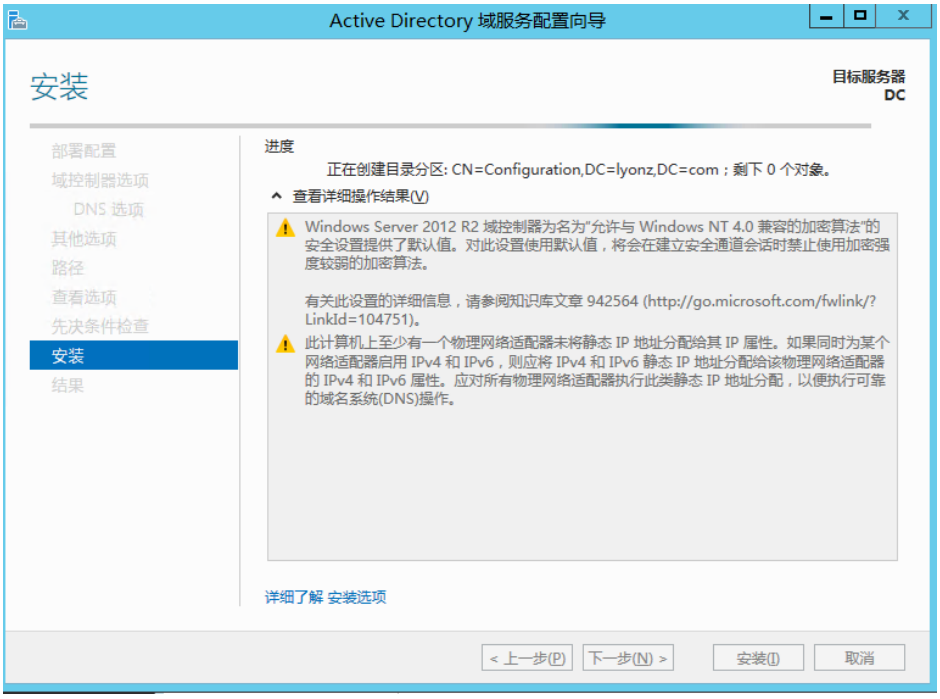

Active Directory 域服务配置向导
部署配置
目标服务器
DC
部署配置
域控制器选项
其他选项
路径
查看选项
先决条件检查
安装
结果
选择部署操作
将域控制器添加到现有域(D)
将新域添加到现有林(E)
添加新林(F)
指定此操作的域信息
根域名(R):
lyonz.com
详细了解 部署配置
< 上一步(P)
下一步(N) >
安装(I)
取消


Active Directory 域服务配置向导
域控制器选项
目标服务器
DC
部署配置
域控制器选项
DNS 选项
其他选项
路径
查看选项
先决条件检查
安装
结果
选择新林和根域的功能级别
林功能级别:
Windows Server 2012 R2
域功能级别:
Windows Server 2012 R2
指定域控制器功能
域名系统(DNS)服务器(O)
全局编录(GC)(G)
只读域控制器(RODC)(R)
键入目录服务还原模式(DSRM)密码
密码(D):
确认密码(C):
详细了解 域控制器选项
< 上一步(P)
下一步(N) >
安装(I)
取消


Active Directory 域服务配置向导
DNS 选项
目标服务器
DC
部署配置
域控制器选项
DNS 选项
其他选项
路径
查看选项
先决条件检查
安装
结果
指定 DNS 委派选项
创建 DNS 委派(D)
创建委派的凭据
DC\administrator
更改(C)...
详细了解 DNS 委派
< 上一步(P)
下一步(N) >
安装(I)
取消


Active Directory 域服务配置向导
其他选项
目标服务器
DC
部署配置
域控制器选项
DNS 选项
其他选项
路径
查看选项
先决条件检查
安装
结果
确保为域分配了 NetBIOS 名称，并在必要时更改该名称
NetBIOS 域名:
LYONZ
详细了解 其他选项


Active Directory 域服务配置向导
查看选项
目标服务器
DC
部署配置
域控制器选项
DNS 选项
其他选项
路径
查看选项
先决条件检查
安装
结果
检查你的选择:
将该服务器配置为新林中的第一个 Active Directory 域控制器。
新域名为"lyonz.com"。这也是新林的名称。
该域的 NetBIOS 名称: LYONZ
林功能级别: Windows Server 2012 R2
域功能级别: Windows Server 2012 R2
其他选项:
全局编录: 是
DNS 服务器: 是
创建 DNS 委派: 是
数据库文件夹: C:\Windows\NTDS
可以将这些设置导出到 Windows PowerShell 脚本以自动执行其他安装
查看脚本(V)
详细了解 安装选项
< 上一步(P)
下一步(N) >
安装(I)
取消


Active Directory 域服务配置向导
安装
目标服务器
DC
部署配置
域控制器选项
DNS 选项
其他选项
路径
查看选项
先决条件检查
安装
结果
进度
正在创建目录分区: CN=Configuration,DC=lyonz,DC=com；剩下 0 个对象。
查看详细操作结果(V)
Windows Server 2012 R2 域控制器为名为"允许与 Windows NT 4.0 兼容的加密算法"的安全设置提供了默认值。对此设置使用默认值，将会在建立安全通道会话时禁止使用加密强度较弱的加密算法。
有关此设置的详细信息，请参阅知识库文章 942564 (http://go.microsoft.com/fwlink/?LinkId=104751)。
此计算机上至少有一个物理网络适配器未将静态 IP 地址分配给其 IP 属性。如果同时为某个网络适配器启用 IPv4 和 IPv6，则应将 IPv4 和 IPv6 静态 IP 地址分配给该物理网络适配器的 IPv4 和 IPv6 属性。应对所有物理网络适配器执行此类静态 IP 地址分配，以便执行可靠的域名系统(DNS)操作。
详细了解 安装选项
< 上一步(P)
下一步(N) >
安装(I)
取消

验证客户端的加入

在云上安装AD和我们线下安装AD步骤其实一样，但客户端加入域的步骤稍有不同，需要先修改客户端的SID，这是因为阿里云ECS Windows Server 2012系统采用的同一个镜像，所以SID是相同的，如果不修改，在加入域的时候会提示SID相同。

修改客户端的SID

Winodws Server 2012：

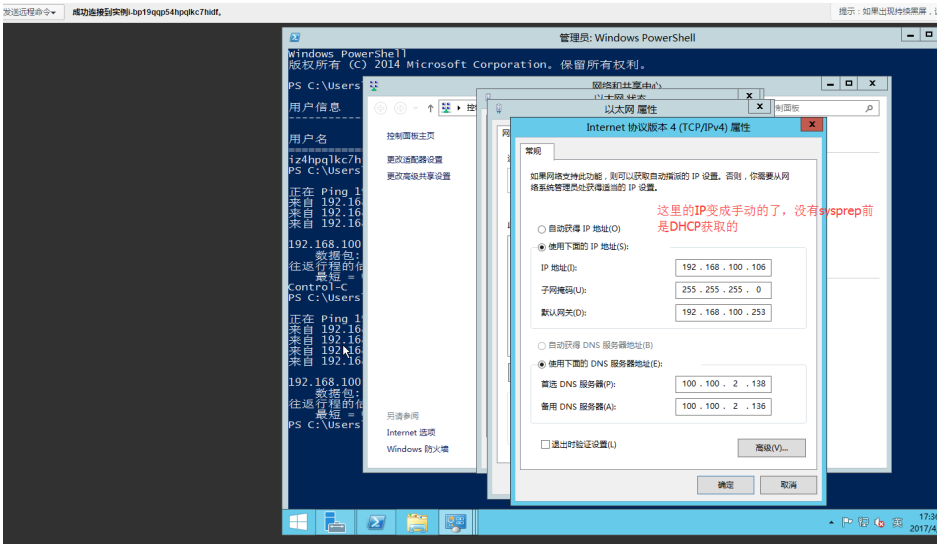
在 powershell 界面执行如下命令：

首先切换到脚本存放的路径，

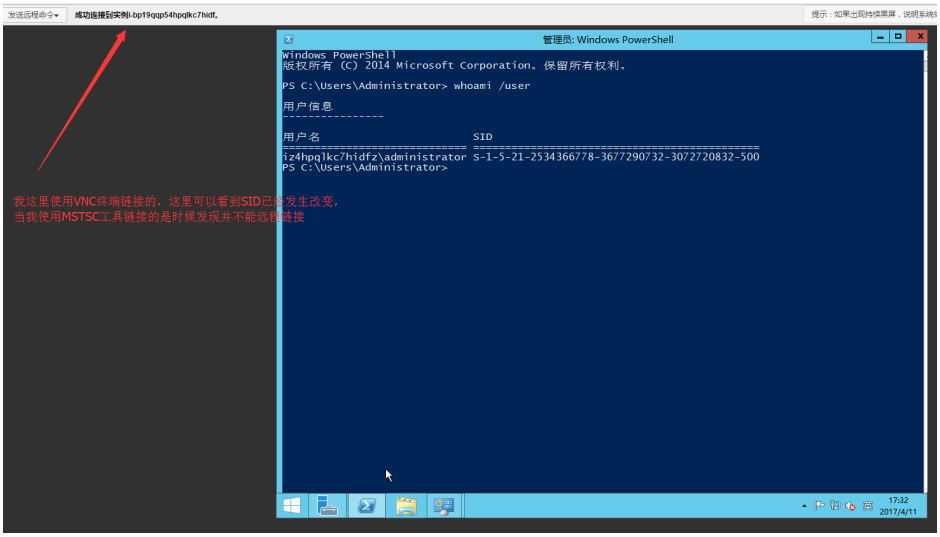
```
.\Sysprep.ps1 -ReserveHostname -ReserveNetwork -skiprearm -post_action "reboot"
```

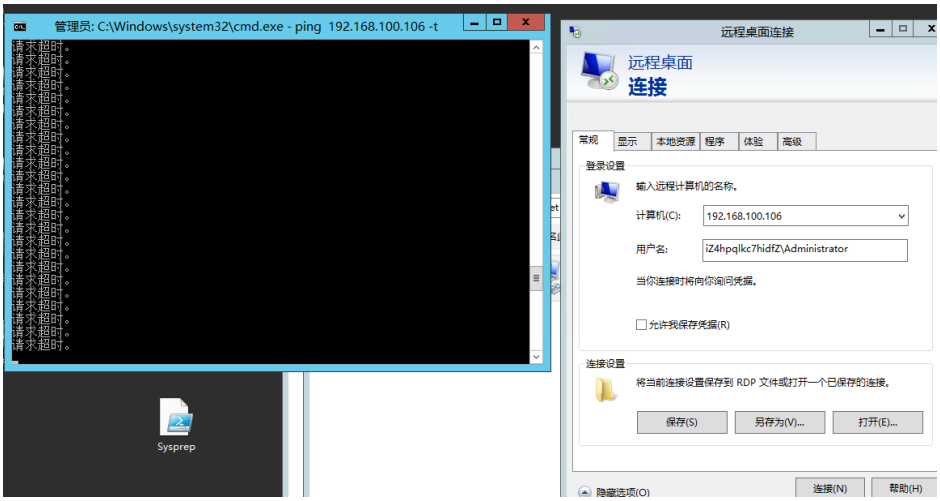
执行上面的命令后，服务器会重新初始化SID，初始化完成后，机器会重启，服务器启动后需要注意两点：

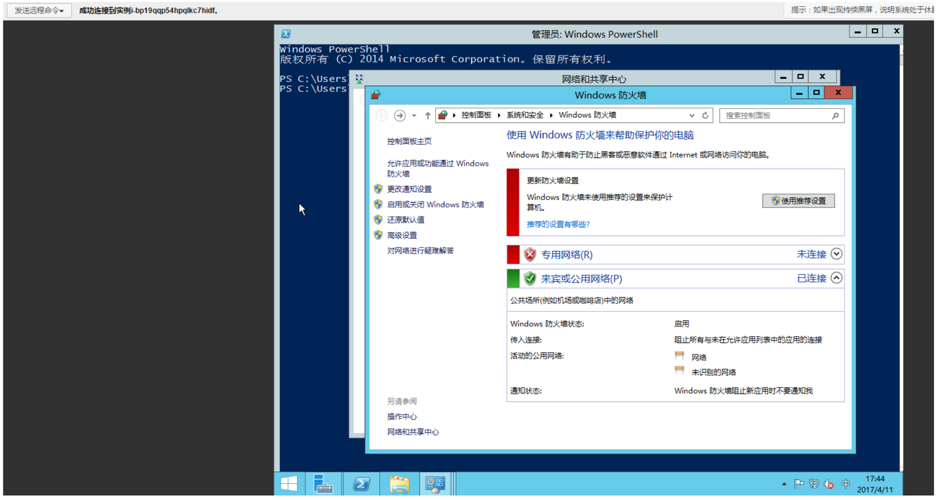
（1）服务器IP地址会从DHCP变成固定IP地址，这里你可以重新改成DHCP，我前面说过，如果想修改ECS 的地址最好从控制台操作。



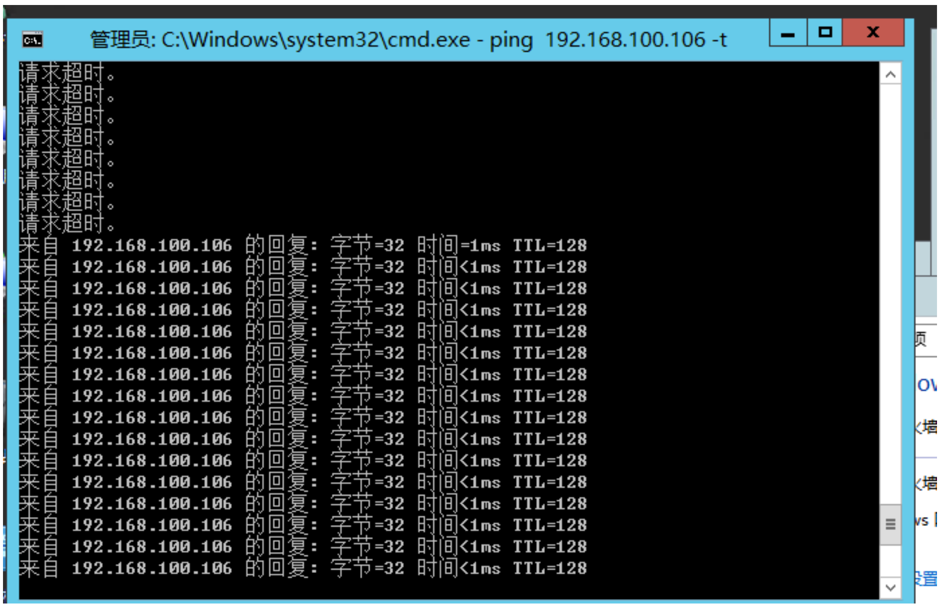
（2）服务器无法PING 通，这是因为服务器SID初始化完成后，也将服务器防火墙的配置修改成微软默认的配置，也就是将“来宾或公用网络”打开，导致无法ping 通服务器和远程。这个时候我们就需要在web console界面将防火墙“来宾或公用网络”关闭，或者放行需要开放的端口。





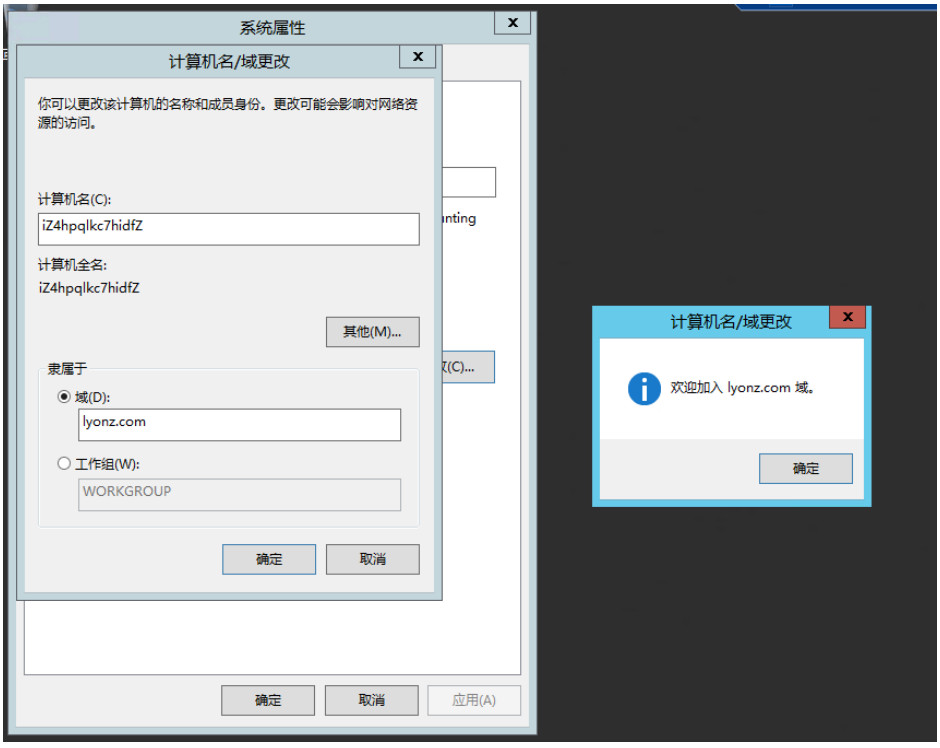

管理员: Windows PowerShell
Windows PowerShell
版权所有 (C) 2014 Microsoft Corporation。保留所有权利。
网络和共享中心
Windows 防火墙
控制面板主页
使用 Windows 防火墙来帮助保护你的电脑
专用网络(R)
未连接
来宾或公用网络(P)
已连接


Windows 防火墙
控制面板 ▸ 所有控制面板项 ▸ Windows 防火墙
控制面板主页
允许应用或功能通过 Windows 防火墙
更改通知设置
启用或关闭 Windows 防火墙
还原默认值
高级设置
对网络进行疑难解答
使用 Windows 防火墙来帮助保护你的电脑
Windows 防火墙有助于防止黑客或恶意软件通过 Internet 或网络访问你的电脑。
更新防火墙设置
Windows 防火墙未使用推荐的设置来保护计算机。
推荐的设置有哪些?
使用推荐设置
域网络(M)
已连接
工作区中连接到域的网络
Windows 防火墙状态:
关闭
传入连接:
阻止所有与未在允许应用列表中的应用的连接
活动的域网络:
lyonz.com
通知状态:
Windows 防火墙阻止新应用时不要通知我
专用网络(R)
未连接
来宾或公用网络(P)
未连接
另请参阅


管理员: C:\Windows\system32\cmd.exe - ping 192.168.100.106 -t
请求超时。
请求超时。
请求超时。
请求超时。
请求超时。
请求超时。
请求超时。
请求超时。
来自 192.168.100.106 的回复: 字节=32 时间=1ms TTL=128
来自 192.168.100.106 的回复: 字节=32 时间<1ms TTL=128
来自 192.168.100.106 的回复: 字节=32 时间<1ms TTL=128
来自 192.168.100.106 的回复: 字节=32 时间<1ms TTL=128
来自 192.168.100.106 的回复: 字节=32 时间<1ms TTL=128
来自 192.168.100.106 的回复: 字节=32 时间<1ms TTL=128
来自 192.168.100.106 的回复: 字节=32 时间<1ms TTL=128
来自 192.168.100.106 的回复: 字节=32 时间<1ms TTL=128
来自 192.168.100.106 的回复: 字节=32 时间<1ms TTL=128
来自 192.168.100.106 的回复: 字节=32 时间<1ms TTL=128
来自 192.168.100.106 的回复: 字节=32 时间<1ms TTL=128
来自 192.168.100.106 的回复: 字节=32 时间<1ms TTL=128
来自 192.168.100.106 的回复: 字节=32 时间<1ms TTL=128
来自 192.168.100.106 的回复: 字节=32 时间<1ms TTL=128
来自 192.168.100.106 的回复: 字节=32 时间<1ms TTL=128
来自 192.168.100.106 的回复: 字节=32 时间<1ms TTL=128

修改客户端的基本信息

(DNS 指向DC 的IP地址，主机名可以根据业务修改相应的名称即可，这里主机名修改不是必要条件。)



以上就是阿里云ECS Windows Server 2012 搭建域以及客户端加入域的过程，如果有在线下（虚拟机）搭建过域的同学，在阿里云上搭建域的时候只需要注意客户端修改SID的问题。

相关链接

域控常见问题配置

更多开源软件尽在云市场，点击此处。

监控

一般来说，在本地数据中心我们会对基础设施进行监控，其中包括对主机实例的监控，以便系统地和随时地了解资源使用情况和性能变化，在出现性能瓶颈的时候合理地调配资源，或者在发生故障时追溯原因等等。

在阿里云上，ECS实例也承载着我们的业务应用，ECS实例的资源使用情况和性能负载直接影响着其上应用的运行稳定性和用户体验度。假如没有进行监控，就很有可能在业务高峰期性能不足却无人问津而导致宕机；也可能在出现异常和故障的时候，因为没有历史性能数据而无法进一步追查到原因，可见，没有监控，当问题出现的时候，都非常被动。

因此，监控是非常有必要的，是构建完整IT系统不可或缺的一个元素，下面就来介绍如何对ECS实例进行监控。

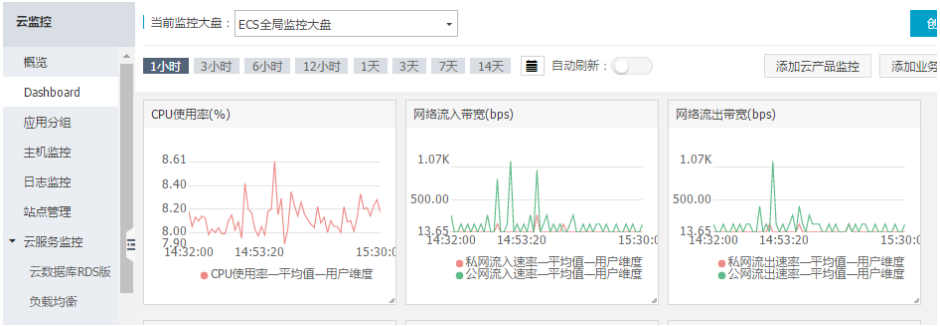
使用Dashboard

云监控的Dashboard功能提供用户自定义查看监控数据的功能。用户可以在一张监控大盘中跨产品、跨实例查看监控数据，将相同业务的不同产品实例集中展现。既能满足排查故障时查看监控细节，又能满足总览大局时查看服务概貌。

操作步骤

1、登录云监控控制台。

2、点击左侧菜单的“Dashboard”选项，进入Dashboard页面。可以看到默认展示的“ECS全局监控大盘”。



3、可以看到默认的“ECS全局监控大盘”已经包含了比较丰富的监控项了，包括CPU使用率、网络流入/流出带宽、系统磁盘BPS、系统盘IOPS、网络流入/流出量。基本已经可以满足日常监控需求。

4、如果业务比较复杂，需要自定义监控可视化需求时，可以创建新的监控大盘，点击页面右上角的“创建监控大盘”，输入监控大盘的名称。

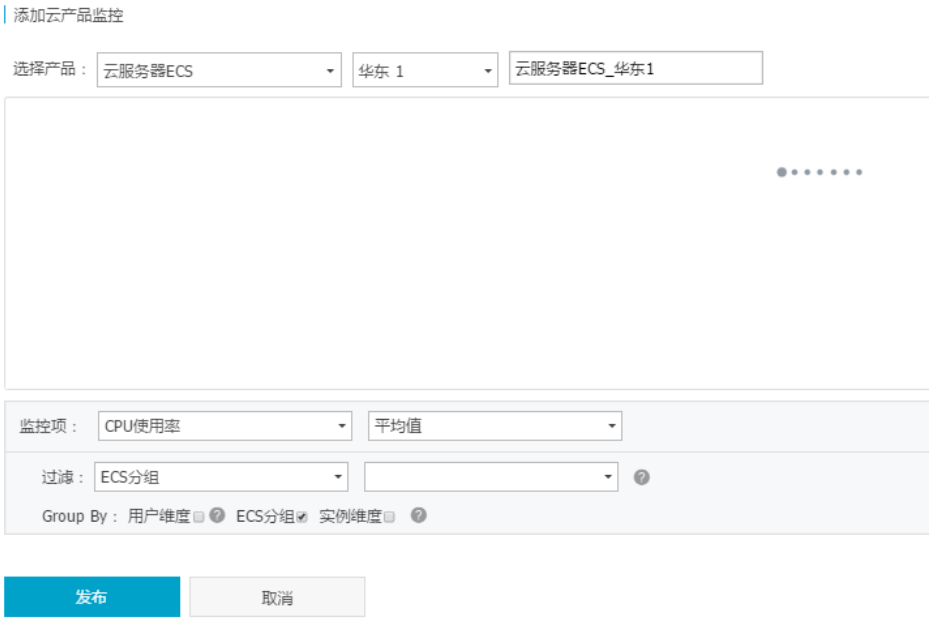
5、然后可以在该大盘上添加云产品指标和用户的业务监控指标。

6、添加云产品指标。

a) 选择需要查看的云产品和实例所在地域；

b) 定义图标名称，图表名称默认为您生成“产品名称+区域”，选择图表展现形式；

c) 选择需要查看的监控项、选择监控数据的聚合方式，常见聚合方式为最大值、最小值、平均值、选择过滤条件、选择Group By的维度。



7、添加业务指标监控。

a) 定义图表名称、指标名称、图表类型；

b) 选择需要查看的监控数据并定义处理方式；

c) 点击发布。

主机监控

云监控主机监控服务通过在服务器上安装插件，为用户提供服务器的系统监控服务。主机监控服务采集丰富的操作系统层面监控指标，可以使用主机监控服务进行服务器资源使用情况的查询和排查故障时的监控数据查询。

操作步骤

1、登录云监控控制台。

2、通过选择左侧菜单的主机监控，进入主机监控页面。

3、点击实例列表中的“点击安装”插件，安装云监控插件。

4、1-3分钟后即可点击实例列表页的“监控图表”查看监控数据。

5、可以看到有操作系统监控、基础监控、进程监控。其中涵盖了CPU、内存、负载、磁盘、网络、进程各面的性能统计，并且可以根据时间范围来展示图标数据。

6、创建报警规则。

a) 切换到报警规则页面；

b) 点击“这里”创建规则；

c) 在新建报警规则页面填写设置报警的具体参数；

d) 保存规则设置，完成报警规则的创建。

站点监控

如果ECS实例提供的主要业务应用是网站类型，可以考虑使用站点监控模拟真实用户访问情况，探测API可用性、端口连通性、DNS解析等问题。可以探测域名、IP、端口的连通性、访问响应时间，并对监控结果报警。

操作步骤

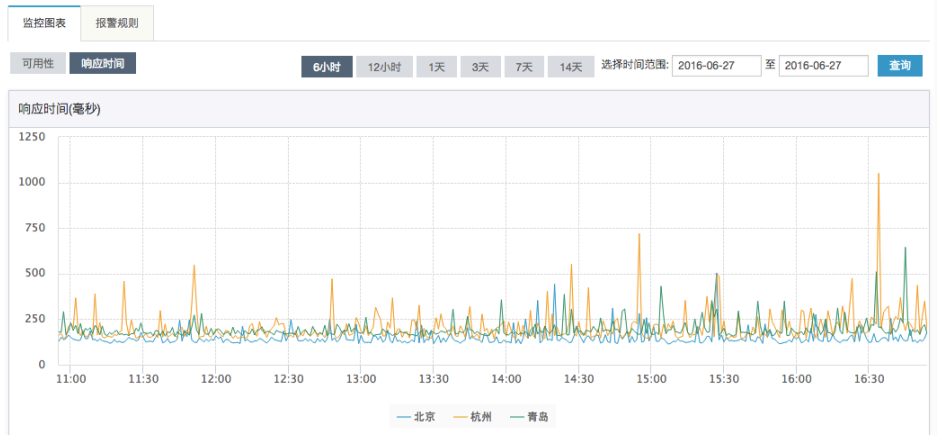
1、登录云监控控制台。

2、点击站点管理，进入站点监控页面。

3、点击页面右上角的创建监控点，添加新的监测点。

4、点击左侧菜单的“站点管理”选项，进入站点监控页面。

5、查看站点监控详情。



开源监控产品介绍

目前业内有不少开源的监控软件，包括zabbix、nagios、zenoss等，每个产品都有各自的特色和优势，下面分别简单介绍一下以上几款产品。

- zabbix

Zabbix是一个基于WEB界面的提供分布式系统监控以及网络监控功能的企业级开源运维平台，也是目前国内互联网用户中使用最广的监控软件，85%以上的泛互联网企业都在使用Zabbix做监控解决方案。

zabbix入门容易、上手简单、功能强大并且开源免费，它易于管理和配置，能生成比较漂亮的数据图，其自动发现功能大大减轻日常管理的工作量，丰富的数据采集方式和API接口可以让用户灵活进行数据采集，而分布式系统架构可以支持监控更多的设备。理论上，通过Zabbix提供的插件式架构，可以满足企业的任何需求。

- nagios

Nagios是一款开源的企业级监控系统，能够实现对系统CPU、磁盘、网络等方面参数的基本系统监控，以及SMTP，POP3，HTTP，NNTP等各种基本的服务类型。另外通过安装插件和编写监控脚本，用户可以实现应用监控，并针对大量的监控主机和多个对象部署层次化监控架构。

Nagios最大的特点是其强大的管理中心，尽管其功能是监控服务和主机的，但Nagios自身并不包括这部分功能代码，所有的监控、告警功能都是由相关插件完成的。

- zenoss

Zenoss Core是Zenoss的开源版本，其商用版本为ZenossEnterprise。作为企业级智能监控软件，Zenoss Core允许IT管理员依靠单一的WEB控制台来监控网络架构的状态和健康度。Zenoss Core的强大能力来自于深入的列表与配置管理数据库，以发现和管理公司IT环境的各类资产。Zenoss同时提供与CMDB关联的事件和错

误管理系统，以协助提高各类事件和提醒的管理效率。

Zabbix vs 云监控

Zabbix是第三方开源监控软件，是一个基于WEB界面的提供分布式系统监视以及网络监视功能的企业级的开源解决方案。

zabbix能监视各种网络参数，保证服务器系统的安全运营；并提供灵活的通知机制以让系统管理员快速定位/解决存在的各种问题。

云监控既指在云端运行的监控工具，也指监控在云端运行的应用程序的工具。通过和云计算平台的整合，针对网络、系统、应用等内容提供可用性、用户体验和安全性方面的监控服务。

云监控的到来，无疑给那些对技术不太熟悉的人员带来了福音，可以通过页面点击就可以创建自己的监控项。

产品	优点	缺点
Zabbix	支持多平台、分布式；安装部署简单，多种数据采集插件灵活集成；可实现复杂多条件告警；自带画图功能，得到的数据可以绘成图形；提供多种API接口，支持调用脚本；出现问题时可自动远程执行命令；	项目批量修改不方便；中文资料较少，服务支持有限；入门容易，但是深层次需要非常熟悉zabbix并进行大量的二次定制开发，难度较大；系统级别报警、报警邮件、自定义项目报警需要自己设置，过程繁琐；缺少数据汇总功能，数据报表也需要进行二次开发；
云监控	无前期成本投入；无需独立服务器；配置及添加监控项简单；页面风格比较适合国人操作；	部分平台免费版功能较少，企业级应用费用较高；账户管理功能较弱；修改监控点配置不方便；自定义监控配置麻烦，部分需写脚本；监控项目单一；部分监控项无法实现图形化显示；

可以看出，各有各的优劣势。云监控降低我们监控的门槛，给我们提供了便利，但是在一定程度上限制了自定义和扩展。而zabbix可以灵活集成并可通过二次开发实现复杂功能，但是对人员和技能的要求也比较高。

对于上监控以更好地保障系统上线后稳定运行，我们还需要关注监控的一些方法。

除了需要了解我们的常规的监控项如硬件资源、性能、带宽、端口、进程、服务的检测机制之外，还要具备安全意识，比如需要知道哪些服务器可能出现问题，可能被入侵等。

另外，需要定义监控策略，包括告警的优先级、告警内容等；对监控的业务系统进行分级，比如一级系统7*24小时告警，二级系统7*12小时告警。

如果架构比较庞大，也可以对监控对象范围进行分类，如服务器监控、应用程序监控、数据库监控、网络监控等，根据监控对象再细分监控项。每个维护人员都可以根据企业环境总结出一套适合于自身的监控体系，并逐渐精细化和智能化。

通过使用阿里云云监控，能较好地对我们的ECS实例进行监控，使我们及时了解业务的运行状态，并及时提供告警，让我们可以快速定位故障，对我们管理和维护ECS提供了可靠的支持。当然，在此基础上我们也可以结合如zabbix之类的开源监控软件，进一步实现对ECS实例更全面和精准的监控。

本文以某门户网站的监控设置为例，讲解云监控服务如何给业务系统做实时护航。

主要内容

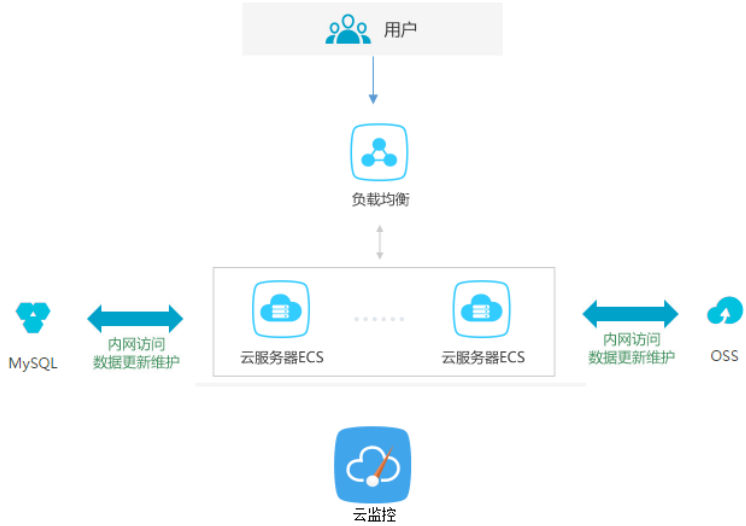
- 监控的必要性
- 云监控配置

监控的必要性

越来越多的用户选择将业务部署在云上，大大减轻了运维成本和压力，其中合理的监控设置功不可没，设置合理的监控不仅可以让用户实时了解系统业务的运行情况，还能帮助用户提前发现问题，避免可能会出现的业务故障；同时有效的告警机制能让用户在故障发生后第一时间发现问题，缩短故障处理时间，以便尽快地恢复业务。

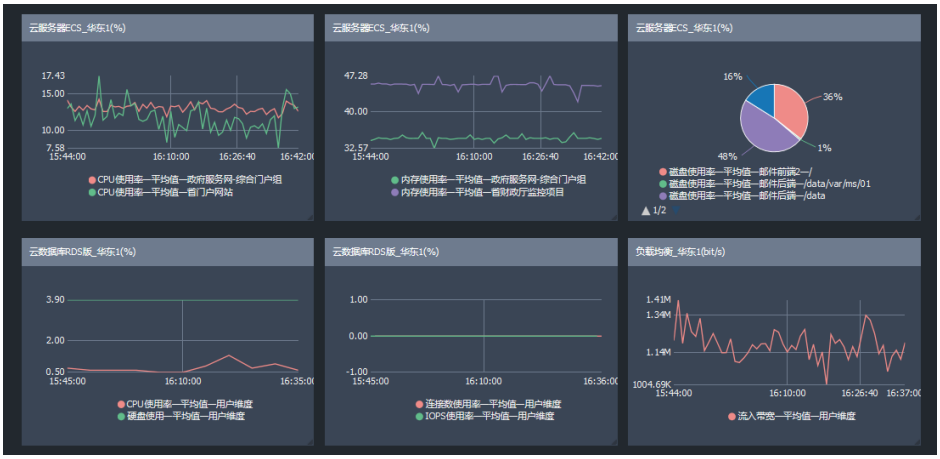
云监控配置

此网站架构如下图所示，其中使用到了阿里云产品ECS，RDS，OSS及负载均衡SLB，下面针对此种类型的架构，说明云监控的配置使用。



在开始设置监控前，需要检查ECS监控插件运行情况，确保监控信息能够正常采集，如安装失败需要手动安装，请参考**云监控插件安装指南**。此外，还需要提前**添加报警联系人和联系组**，建议设置至少2人以上的联系人，互为主备，以便及时响应监控告警。监控选项的设定，具体可参见**云服务资源使用概览和报警概览**。利用云监控的Dashboard功能，给您业务系统的云资源设置一个全局监控总览，可随时检查整个业务系统资源的健康状态。下图根据ECS分组选择添加监控的资源，依次添加内存使用率，CPU使用率等监控项。监控的实例数较少可以选择实例维度作为展示，如有多实例建议以分组或者用户为维度展示；监控数据取平均值。

为了更好的监控大屏展示效果，这里将ECS的CPU、内存、磁盘的使用率单独分组展示；将RDS的四项指标分两组展示。



报警阈值

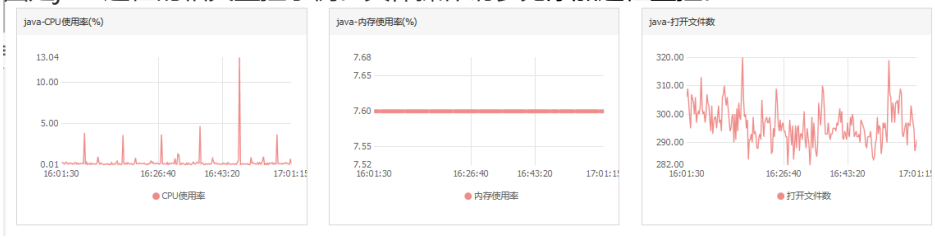
关于各项监控指标的报警阈值说明，建议根据实际业务情况斟酌设置，不要设置太低以免频繁触发报警影响监控服务体验，也不要设置太高以免触发阈值后没有足够的预留时间来响应和处理告警。

报警规则

以CPU使用率为例，由于需要给服务器预留部分处理性能保障服务器正常运行，所以建议将cpu告警阈值设置为70%，连续三次超过阈值后开始报警。如下图所示点击添加报警规则继续设置内存和磁盘的报警规则和报警通知人即可。

进程监控

对于常见的web应用，设置进程监控，不仅可以实时监控应用进程的运行情况，还有助于故障的排查处理，下图是java进程的相关监控示例。具体操作请参见添加进程监控。



站点监控

在云服务器外层的监控服务，站点监控主要用于模拟真实用户访问情况，实时测试业务可用性，有助于的故障排查处理，具体创建方法参见如何创建站点监控。

RDS监控

建议将RDS的CPU使用率告警阈值设置为70%，连续三次超过阈值后开始报警。硬盘使用率，最大IOPS使用率，连接数等其他监控项可根据您的实际情况来设置。

负载均衡监控

为了更好使用负载均衡的云监控服务，需要先开启负载均衡SLB的健康检查，详情参见健康检查机制和配置说明建议设置负载均衡SLB带宽值的70%作为告警阈值，如下图所示。

如以上监控选项不能满足您的实际业务监控需求，可以参见创建自定义监控项和报警规则。

使用 API 管理 ECS

除了可以在ECS控制台或售卖页创建 ECS 外，您还可以使用 OpenAPI 代码来弹性地创建和管理ECS。本页面使用 Python 为例进行说明。

创建 ECS 时需关注以下 API：

- 创建ECS实例
- 查询实例列表
- 启动ECS实例
- 分配公网IP地址

前提条件

开通按量付费产品，您的账户余额不得少于 100 元，更多的需求参见 ECS使用须知。您需要在阿里云的费用中心确保自己的余额充足。

创建按量云服务器

创建云服务器时的必选属性：

- SecurityGroupId：安全组 ID。安全组通过防火墙规则实现对一组实例的配置，保护实例的网络出入请求。在设置安全组出入规则时，建议按需开放而不要默认开放所有的出入规则。您也可以通过 ECS 控制台创建安全组。
- InstanceType：实例规格。参考 ECS 售卖页的选项，界面上 1 核 2GB n1.small则入参为 ecs.n1.small。
- ImageId：镜像 ID。参考ECS控制台的镜像列表，您可以过滤系统公共镜像或者自定义镜像。

更多参数设置请参考创建 ECS 实例。

创建云服务器

如下面的代码所示，创建一台经典网络的ECS，使用系统盘ssd，盘参数为cloud_ssd，选择io优化实例optimized。

```
# create one after pay ecs instance.
def create_after_pay_instance(image_id, instance_type, security_group_id):
request = CreateInstanceRequest();
request.set_ImageId(image_id)
request.set_SecurityGroupId(security_group_id)
request.set_InstanceType(instance_type)
request.set_IoOptimized('optimized')
request.set_SystemDiskCategory('cloud_ssd')
response = _send_request(request)
instance_id = response.get('InstanceId')
logging.info("instance %s created task submit successfully.", instance_id)
return instance_id;
```

创建成功后将返回相应的实例 ID，失败的话也会有对应的 ErrorCode。由于参数较多，您可以参考 ECS 的售卖页进行调整。

```
{"InstanceId":"i-***","RequestId":"006C1303-BAC5-48E5-BCDF-7FD5C2E6395D"}
```

云服务器生命周期

对于云服务器的状态操作, 请参考云服务器实例生命周期。

只有Stopped状态的实例可以执行 Start 操作。也只有Running状态的 ECS 可以执行Stop操作。查询云服务器的状态可以通过查询实例列表传入 InstanceId 进行过滤。在DescribeInstancesRequest时可以通过传入一个 JSON 数组格式的 String 就可以查询这个资源的状态。查询单个实例的状态建议使用DescribeInstances而不要使用DescribeInstanceAttribute, 因为前者比后者返回更多的属性和内容。

下面的代码会检查实例的状态，只有实例的状态符合入参才会返回实例的详情。

```
# output the instance owned in current region.
def get_instance_detail_by_id(instance_id, status='Stopped'):
logging.info("Check instance %s status is %s", instance_id, status)
request = DescribeInstancesRequest()
request.set_InstanceIds(json.dumps([instance_id]))
response = _send_request(request)
instance_detail = None
if response is not None:
instance_list = response.get('Instances').get('Instance')
for item in instance_list:
if item.get('Status') == status:
instance_detail = item
break;
return instance_detail;
```

启动云服务器

创建成功后的 ECS 默认状态是Stopped。如果要启动 ECS 实例为Running状态，只需要发送启动指令即可。

```
def start_instance(instance_id):
request = StartInstanceRequest()
request.set_InstanceId(instance_id)
_send_request(request)
```

停止云服务器

停止云服务器只需传入instanceId即可。

```
def stop_instance(instance_id):
request = StopInstanceRequest()
request.set_InstanceId(instance_id)
_send_request(request)
```

创建时启动“自动启动云服务器”

服务器的启动和停止都是一个异步操作，您可以在脚本创建并同时检测云服务器符合状态时执行相应操作。

创建资源后得到实例ID，首先判断实例是否处于Stopped的状态，如果处于Stopped状态，下发Start服务器的指令，然后等待服务器的状态变成Running。

```
def check_instance_running(instance_id):
detail = get_instance_detail_by_id(instance_id=instance_id, status=INSTANCE_RUNNING)
index = 0
while detail is None and index &lt; 60:
detail = get_instance_detail_by_id(instance_id=instance_id);
time.sleep(10)

if detail and detail.get('Status') == 'Stopped':
logging.info("instance %s is stopped now.")
start_instance(instance_id=instance_id)
logging.info("start instance %s job submit.")

detail = get_instance_detail_by_id(instance_id=instance_id, status=INSTANCE_RUNNING)
while detail is None and index &lt; 60:
detail = get_instance_detail_by_id(instance_id=instance_id, status=INSTANCE_RUNNING);
time.sleep(10)

logging.info("instance %s is running now.", instance_id)
return instance_id;
```

分配公网IP

如果在创建云服务器的过程中，指定了公网带宽，若需要公网的访问权限还要调用API来分配公网IP。详情请参考：分配公网 IP 地址。

包年包月的资源创建

除了创建按量服务的云服务器，您的API还支持创建包年包月的服务器。包年包月的创建和官网的创建流程不同，使用的是自动扣费的模式，也就是说您需要在创建服务器之前确保账号有足够的余额或者信用额度，在创建的时候将直接扣费。

和按量付费的 ECS 相比，只需要指定付费类型和时长即可，下面的时长为1个月。

```
    request.set_Period(1)    request.set_InstanceChargeType（‘PrePaid’）
```

创建包年包月实例的整体的代码如下:

```
# create one prepay ecs instance.
def create_prepay_instance(image_id, instance_type, security_group_id):
request = CreateInstanceRequest();
request.set_ImageId(image_id)
```

```
request.set_SecurityGroupId(security_group_id)
request.set_InstanceType(instance_type)
request.set_IoOptimized('optimized')
request.set_SystemDiskCategory('cloud_ssd')
request.set_Period(1)
request.set_InstanceChargeType('PrePaid')
response = _send_request(request)
instance_id = response.get('InstanceId')
logging.info("instance %s created task submit successfully.", instance_id)
return instance_id;
```

完整的代码

完整的代码如下，您可以按照自己的资源参数进行设置。

```
#  coding=utf-8
# if the python sdk is not install using 'sudo pip install aliyun-python-sdk-ecs'
# if the python sdk is install using 'sudo pip install --upgrade aliyun-python-sdk-ecs'
# make sure the sdk version is 2.1.2, you can use command 'pip show aliyun-python-sdk-ecs' to check

import json
import logging
import time

from aliyunsdkcore import client
from aliyunsdkecs.request.v20140526.CreateInstanceRequest import CreateInstanceRequest
from aliyunsdkecs.request.v20140526.DescribeInstancesRequest import DescribeInstancesRequest
from aliyunsdkecs.request.v20140526.StartInstanceRequest import StartInstanceRequest

# configuration the log output formatter, if you want to save the output to file,
# append ",filename='ecs_invoke.log'" after datefmt.

logging.basicConfig(level=logging.INFO,
format='%(asctime)s %(filename)s[line:%(lineno)d] %(levelname)s %(message)s',
datefmt='%a, %d %b %Y %H:%M:%S')

clt = client.AcsClient('Your Access Key Id', 'Your Access Key Secrect', 'cn-beijing')

IMAGE_ID = 'ubuntu1404_64_40G_cloudinit_20160727.raw'
INSTANCE_TYPE = 'ecs.s2.large' # 2c4g generation 1
SECURITY_GROUP_ID = 'sg-****'
INSTANCE_RUNNING = 'Running'

def create_instance_action():
instance_id = create_after_pay_instance(image_id=IMAGE_ID, instance_type=INSTANCE_TYPE,
security_group_id=SECURITY_GROUP_ID)
check_instance_running(instance_id=instance_id)

def create_prepay_instance_action():
instance_id = create_prepay_instance(image_id=IMAGE_ID, instance_type=INSTANCE_TYPE,
security_group_id=SECURITY_GROUP_ID)
check_instance_running(instance_id=instance_id)

# create one after pay ecs instance.
```

```
def create_after_pay_instance(image_id, instance_type, security_group_id):
request = CreateInstanceRequest();
request.set_ImageId(image_id)
request.set_SecurityGroupId(security_group_id)
request.set_InstanceType(instance_type)
request.set_IoOptimized('optimized')
request.set_SystemDiskCategory('cloud_ssd')
response = _send_request(request)
instance_id = response.get('InstanceId')
logging.info("instance %s created task submit successfully.", instance_id)
return instance_id;

# create one prepay ecs instance.
def create_prepay_instance(image_id, instance_type, security_group_id):
request = CreateInstanceRequest();
request.set_ImageId(image_id)
request.set_SecurityGroupId(security_group_id)
request.set_InstanceType(instance_type)
request.set_IoOptimized('optimized')
request.set_SystemDiskCategory('cloud_ssd')
request.set_Period(1)
request.set_InstanceChargeType('PrePaid')
response = _send_request(request)
instance_id = response.get('InstanceId')
logging.info("instance %s created task submit successfully.", instance_id)
return instance_id;

def check_instance_running(instance_id):
detail = get_instance_detail_by_id(instance_id=instance_id, status=INSTANCE_RUNNING)
index = 0
while detail is None and index &lt; 60:
detail = get_instance_detail_by_id(instance_id=instance_id);
time.sleep(10)

if detail and detail.get('Status') == 'Stopped':
logging.info("instance %s is stopped now.")
start_instance(instance_id=instance_id)
logging.info("start instance %s job submit.")

detail = get_instance_detail_by_id(instance_id=instance_id, status=INSTANCE_RUNNING)
while detail is None and index &lt; 60:
detail = get_instance_detail_by_id(instance_id=instance_id, status=INSTANCE_RUNNING);
time.sleep(10)

logging.info("instance %s is running now.", instance_id)
return instance_id;

def start_instance(instance_id):
request = StartInstanceRequest()
request.set_InstanceId(instance_id)
_send_request(request)

# output the instance owned in current region.
def get_instance_detail_by_id(instance_id, status='Stopped'):
logging.info("Check instance %s status is %s", instance_id, status)
request = DescribeInstancesRequest()
```

```
request.set_InstanceIds(json.dumps([instance_id]))
response = _send_request(request)
instance_detail = None
if response is not None:
instance_list = response.get('Instances').get('Instance')
for item in instance_list:
if item.get('Status') == status:
instance_detail = item
break;
return instance_detail;

# send open api request
def _send_request(request):
request.set_accept_format('json')
try:
response_str = clt.do_action(request)
logging.info(response_str)
response_detail = json.loads(response_str)
return response_detail
except Exception as e:
logging.error(e)

if __name__ == '__main__':
logging.info("Create ECS by OpenApi!")
create_instance_action()
# create_prepay_instance_action()
```

您除了可以通过 ECS 管理控制台 创建或管理 ECS 实例外，您也能通过 OpenAPI 管理或定制开发 ECS 实例。

阿里云提供了 SDK 来包装 OpenAPI，将云服务器 ECS 的管理集成到已有系统中。本文基于 Python 的开发来说明如何通过 OpenAPI 管理 ECS 实例。如果您没有 Python 开发经验，也能通过本文完成云服务的开发。

获取 RAM 子账号 AK 密钥

使用 OpenAPI 管理 ECS 实例，您需要能访问 ECS 资源的 API 密钥（Access Key ID 和 Access Key Secret）。为了保证云服务的安全，您需要创建一个能访问 ECS 资源的 RAM 子账号，获取该子账号的 AK 密钥，并使用这个 RAM 子账号和 OpenAPI 管理 ECS 实例。

以下是获取 RAM 子账号 AK 密钥的操作步骤：

1. 创建 RAM 用户并获取 AK 密钥。
2. 直接给 RAM 用户授权，授予 RAM 子账号 **管理云服务器服务(ECS)的权限。**

安装 ECS Python SDK

首先确保您已经具备 Python 的 Runtime，本文中使用的 Python 版本为 2.7+。

```
pip install aliyun-python-SDK-ecs
```

如果提示您没有权限，请切换sudo继续执行。

```
sudo pip install aliyun-python-SDK-ecs
```

本文使用的 SDK 版本为 **2.1.2**。

Hello Alibaba Cloud

创建文件 **hello_ecs_api.py**。 为了使用 SDK，首先实例化 AcsClient 对象，这里需要 RAM 子账号的 Accesskey 和 Accesskey Secret。

> Access Key ID 和 Access Key Secret 是 RAM 子账号访问阿里云 ECS 服务 API的密钥，具有该账户完全的权限，请妥善保管。

```
from aliyunSDKcore import client
from aliyunSDKecs.request.v20140526.DescribeInstancesRequest import DescribeInstancesRequest
from aliyunSDKecs.request.v20140526.DescribeRegionsRequest import DescribeRegionsRequest
clt = client.AcsClient('Your Access Key Id', 'Your Access Key Secret', 'cn-beijing')
```

完成实例化后可以进行第一个应用的开发。查询当前账号支持的地域列表。具体的文档参见 **查询可用地域列表**。

```
def hello_aliyun_regions():
request = DescribeRegionsRequest()
response = _send_request(request)
region_list = response.get('Regions').get('Region')
assert response is not None
assert region_list is not None
result = map(_print_region_id, region_list)
logging.info("region list: %s", result)

def _print_region_id(item):
region_id = item.get("RegionId")
return region_id

def _send_request(request):
request.set_accept_format('json')
try:
response_str = clt.do_action(request)
logging.info(response_str)
response_detail = json.loads(response_str)
return response_detail
except Exception as e:
logging.error(e)

hello_aliyun_regions()
```

在命令行运行 python hello_ecs_api.py 会得到当前支持的 Region列表。类似的输出如下：

```
[u'cn-shenzhen', u'ap-southeast-1', u'cn-qingdao', u'cn-beijing', u'cn-shanghai', u'us-east-1', u'cn-hongkong', u'me-east-1', u'ap-southeast-2', u'cn-hangzhou', u'eu-central-1', u'ap-northeast-1', u'us-west-1']
```

查询当前的 Region 下的 ECS 实例列表

查询实例列表和查询 Region 列表非常类似，替换入参对象为DescribeInstancesRequest 即可，更多的查询参数参考 查询实例列表。

```
def list_instances():
request = DescribeInstancesRequest()
response = _send_request(request)
if response is not None:
instance_list = response.get('Instances').get('Instance')
result = map(_print_instance_id, instance_list)
logging.info("current region include instance %s", result)

def _print_instance_id(item):
instance_id = item.get('InstanceId');
return instance_id
```

输出结果为如下：

```
current region include instance [u'i-****', u'i-****'']
```

更多的API参考 ECS API 概览，您可以尝试作一个 查询磁盘列表，将实例的参数替换为 DescribeDisksRequest。

完整代码示例

以上操作完整的代码示例如下所示。

```
#  coding=utf-8
# if the python SDK is not install using 'sudo pip install aliyun-python-SDK-ecs'
# if the python SDK is install using 'sudo pip install --upgrade aliyun-python-SDK-ecs'
# make sure the SDK version is 2.1.2, you can use command 'pip show aliyun-python-SDK-ecs' to check

import json
import logging

from aliyunSDKcore import client
from aliyunSDKecs.request.v20140526.DescribeInstancesRequest import DescribeInstancesRequest
from aliyunSDKecs.request.v20140526.DescribeRegionsRequest import DescribeRegionsRequest

# configuration the log output formatter, if you want to save the output to file,
# append ",filename='ecs_invoke.log'" after datefmt.
logging.basicConfig(level=logging.INFO,
format='%(asctime)s %(filename)s[line:%(lineno)d] %(levelname)s %(message)s',
datefmt='%a, %d %b %Y %H:%M:%S')
```

```
clt = client.AcsClient('Your Access Key Id', 'Your Access Key Secret', 'cn-beijing')

# sample api to list aliyun open api.
def hello_aliyun_regions():
request = DescribeRegionsRequest()
response = _send_request(request)
if response is not None:
region_list = response.get('Regions').get('Region')
assert response is not None
assert region_list is not None
result = map(_print_region_id, region_list)
logging.info("region list: %s", result)

# output the instance owned in current region.
def list_instances():
request = DescribeInstancesRequest()
response = _send_request(request)
if response is not None:
instance_list = response.get('Instances').get('Instance')
result = map(_print_instance_id, instance_list)
logging.info("current region include instance %s", result)

def _print_instance_id(item):
instance_id = item.get('InstanceId');
return instance_id

def _print_region_id(item):
region_id = item.get("RegionId")
return region_id

# send open api request
def _send_request(request):
request.set_accept_format('json')
try:
response_str = clt.do_action(request)
logging.info(response_str)
response_detail = json.loads(response_str)
return response_detail
except Exception as e:
logging.error(e)

if __name__ == '__main__':
logging.info("Hello Aliyun OpenAPI!")
hello_aliyun_regions()
list_instances()
```

如您想了解 ECS 中 API 的其它操作，请参考 ECS中的API操作。

云服务器 ECS 的一个重要特性就是按需创建资源。您可以在业务高峰期按需弹性地进行自定义资源创建，完成业务计算时释放资源。本篇将提供若干 Tips 帮助您更加便捷地完成云服务器的释放以及弹性设置。

本文将涉及到几个重要功能和相关API:

- 释放按量付费的云服务器
- 设置按量付费实例的自动释放时间
- 停止服务器
- 查询实例列表

释放后，实例所使用的物理资源将被回收，包括磁盘及快照，相关数据将全部丢失且永久不可恢复。如果您还想继续使用相关的数据，建议您释放云服务器之前一定要对磁盘数据做快照，下次创建 ECS 时可以直接通过快照创建资源。

释放云服务器

释放服务器，首先要求您的服务器处于停止状态。当服务器停止后，若影响到应用，您可以将服务器重新启动。

停止云服务器

停止服务器的指令非常简单，且对于按量付费和包年包月都是一样的。停止云服务器的一个参数是 ForceStop，若属性设置为 true，它将类似于断电，直接停止服务器，但不承诺数据能写到磁盘中。如果仅仅为了释放服务器，这个可以设置为 true。

```
def stop_instance(instance_id, force_stop=False):
'''
stop one ecs instance.
:param instance_id: instance id of the ecs instance, like 'i-***'.
:param force_stop: if force stop is true, it will force stop the server and not ensure the data
write to disk correctly.
:return:
'''
request = StopInstanceRequest()
request.set_InstanceId(instance_id)
request.set_ForceStop(force_stop)
logging.info("Stop %s command submit successfully.", instance_id)
_send_request(request)
```

释放云服务器

如果您没有停止服务器直接执行释放，可能会有如下报错：

```
{"RequestId":"3C6DEAB4-7207-411F-9A31-6ADE54C268BE","HostId":"ecs-cn-
hangzhou.aliyuncs.com","Code":"IncorrectInstanceStatus","Message":"The current status of the resource does not
support this operation."}
```

当服务器处于Stopped状态时，您可以执行释放服务器。释放服务器的方法比较简单，参数如下：

- InstanceId: 实例的 ID

- force: 如果将这个参数设置为 true，将会执行强制释放。即使云服务器不是Stopped状态也可以释放。执行的时候请务必小心，以防错误释放影响您的业务。

python def release_instance(instance_id, force=False): ''' delete instance according instance id, only support after pay instance. :param instance_id: instance id of the ecs instance, like 'i-***'. :param force: if force is false, you need to make the ecs instance stopped, you can execute the delete action. If force is true, you can delete the instance even the instance is running. :return: ''' request = DeleteInstanceRequest(); request.set_InstanceId(instance_id) request.set_Force(force) _send_request(request)
释放云服务器成功的 Response 如下:

```
{ “RequestId”:” 689E5813-D150-4664-AF6F-2A27BB4986A3” }
```

设置云服务器的自动释放时间

为了更加简化对云服务器的管理，您可以自定义云服务器的释放时间。当定时时间到后，阿里云将自动为您完成服务器的释放, 无需手动执行释放。

注意：自动释放时间按照 ISO8601 标准表示，并需要使用 UTC 时间。 格式为：yyyy-MM-ddTHH:mm:ssZ。 如果秒不是 00，则自动取为当前分钟开始时。自动释放的时间范围：当前时间后 30 分钟 ~ 当前时间起 3 年。

```
def set_instance_auto_release_time(instance_id, time_to_release = None):
'''
setting instance auto delete time
:param instance_id: instance id of the ecs instance, like 'i-***'.
:param time_to_release: if the property is setting, such as '2017-01-30T00:00:00Z'
it means setting the instance to be release at that time.
if the property is None, it means cancel the auto delete time.
:return:
'''
request = ModifyInstanceAutoReleaseTimeRequest()
request.set_InstanceId(instance_id)
if time_to_release is not None:
request.set_AutoReleaseTime(time_to_release)
_send_request(request)
```

执行 set_instance_auto_release_time(‘i-1111’ , ‘2017-01-30T00:00:00Z’) 后完成设置。

执行设置成功后，您可以通过DescribeInstances来查询自动释放的时间设置。

```
def describe_instance_detail(instance_id):
'''
describe instance detail
:param instance_id: instance id of the ecs instance, like 'i-***'.
:return:
'''
```

```
request = DescribeInstancesRequest()
request.set_InstanceIds(json.dumps([instance_id]))
response = _send_request(request)
if response is not None:
instance_list = response.get('Instances').get('Instance')
if len(instance_list) > 0:
return instance_list[0]


def check_auto_release_time_ready(instance_id):
detail = describe_instance_detail(instance_id=instance_id)
if detail is not None:
release_time = detail.get('AutoReleaseTime')
return release_time
```

取消自动释放设置

如果您的业务有变化，需要取消自动释放设置。只需执行命令将自动释放时间设置为空即可。

```
set_instance_auto_release_time('i-1111')
```

完整代码如下:

注意：释放云服务器需谨慎。

```
#  coding=utf-8

# if the python sdk is not install using 'sudo pip install aliyun-python-sdk-ecs'
# if the python sdk is install using 'sudo pip install --upgrade aliyun-python-sdk-ecs'
# make sure the sdk version is 2.1.2, you can use command 'pip show aliyun-python-sdk-ecs' to check

import json
import logging

from aliyunsdkcore import client
from aliyunsdkecs.request.v20140526.DeleteInstanceRequest import DeleteInstanceRequest
from aliyunsdkecs.request.v20140526.DescribeInstancesRequest import DescribeInstancesRequest
from aliyunsdkecs.request.v20140526.ModifyInstanceAutoReleaseTimeRequest import \
ModifyInstanceAutoReleaseTimeRequest
from aliyunsdkecs.request.v20140526.StopInstanceRequest import StopInstanceRequest

# configuration the log output formatter, if you want to save the output to file,
# append ",filename='ecs_invoke.log'" after datefmt.
logging.basicConfig(level=logging.INFO,
format='%(asctime)s %(filename)s[line:%(lineno)d] %(levelname)s %(message)s',
datefmt='%a, %d %b %Y %H:%M:%S')

clt = client.AcsClient('Your Access Key Id', 'Your Access Key Secrect', 'cn-beijing')

def stop_instance(instance_id, force_stop=False):
'''
stop one ecs instance.
```

```
:param instance_id: instance id of the ecs instance, like 'i-***'.
:param force_stop: if force stop is true, it will force stop the server and not ensure the data
write to disk correctly.
:return:
'''
request = StopInstanceRequest()
request.set_InstanceId(instance_id)
request.set_ForceStop(force_stop)
logging.info("Stop %s command submit successfully.", instance_id)
_send_request(request)

def describe_instance_detail(instance_id):
'''
describe instance detail
:param instance_id: instance id of the ecs instance, like 'i-***'.
:return:
'''
request = DescribeInstancesRequest()
request.set_InstanceIds(json.dumps([instance_id]))
response = _send_request(request)
if response is not None:
instance_list = response.get('Instances').get('Instance')
if len(instance_list) &gt; 0:
return instance_list[0]

def check_auto_release_time_ready(instance_id):
detail = describe_instance_detail(instance_id=instance_id)
if detail is not None:
release_time = detail.get('AutoReleaseTime')
return release_time

def release_instance(instance_id, force=False):
'''
delete instance according instance id, only support after pay instance.
:param instance_id: instance id of the ecs instance, like 'i-***'.
:param force:
if force is false, you need to make the ecs instance stopped, you can
execute the delete action.
If force is true, you can delete the instance even the instance is running.
:return:
'''
request = DeleteInstanceRequest();
request.set_InstanceId(instance_id)
request.set_Force(force)
_send_request(request)

def set_instance_auto_release_time(instance_id, time_to_release = None):
'''
setting instance auto delete time
:param instance_id: instance id of the ecs instance, like 'i-***'.
:param time_to_release: if the property is setting, such as '2017-01-30T00:00:00Z'
it means setting the instance to be release at that time.
if the property is None, it means cancel the auto delete time.
:return:
'''
request = ModifyInstanceAutoReleaseTimeRequest()
```

```
request.set_InstanceId(instance_id)
if time_to_release is not None:
request.set_AutoReleaseTime(time_to_release)
_send_request(request)
release_time = check_auto_release_time_ready(instance_id)
logging.info("Check instance %s auto release time setting is %s. ", instance_id, release_time)

def _send_request(request):
'''
send open api request
:param request:
:return:
'''
request.set_accept_format('json')
try:
response_str = clt.do_action(request)
logging.info(response_str)
response_detail = json.loads(response_str)
return response_detail
except Exception as e:
logging.error(e)

if __name__ == '__main__':
logging.info("Release ecs instance by Aliyun OpenApi!")
set_instance_auto_release_time('i-1111', '2017-01-28T06:00:00Z')
# set_instance_auto_release_time('i-1111')
# stop_instance('i-1111')
# release_instance('i-1111')
# release_instance('i-1111', True)
```

如您想了解 ECS 中 API 的其它操作，请参考 ECS中的API操作。

除了通过 ECS控制台 或 售卖页 进行云服务器续费外，阿里云还支持直接通过 API 进行续费查询和续费管理。

本文主要涉及如下关键功能：

- 按照过期时间查询云服务器
- 续费实例
- 查询云服务器自动续费时间
- 设置云服务器自动续费时间

对于包年包月的云服务器，生命周期非常重要。如果云服务器资源不能按时续费，将可能导致服务器被锁定甚至被释放，从而影响业务持续性。API 帮助您及时了解和检查资源的到期时间，并完成续费充值功能。

本篇需关注如下 API：

- 查询实例列表
- 续费实例

查询指定范围内到期的云服务器

查询实例列表的 API，通过过滤参数，您可以查询一定时间范围内到期的实例信息。通过设置过滤参数

ExpiredStartTime 和 ExpiredEndTime（时间参数 按照 ISO8601 标准表示，并需要使用 UTC 时间。 格式为：yyyy-MM-ddTHH:mmZ。），可以方便地查询该时间范围内到期的实例列表。如果需要通过安全组进行过滤，只需加上安全组 ID 即可。

```
INSTANCE_EXPIRED_START_TIME_IN_UTC_STRING = '2017-01-22T00:00Z'
INSTANCE_EXPIRE_END_TIME_IN_UTC_STRING = '2017-01-28T00:00Z'

def describe_need_renew_instance(page_size=100, page_number=1, instance_id=None,
check_need_renew=True, security_group_id=None):
request = DescribeInstancesRequest()
if check_need_renew is True:
request.set_Filter3Key("ExpiredStartTime")
request.set_Filter3Value(INSTANCE_EXPIRED_START_TIME_IN_UTC_STRING)
request.set_Filter4Key("ExpiredEndTime")
request.set_Filter4Value(INSTANCE_EXPIRE_END_TIME_IN_UTC_STRING)
if instance_id is not None:
request.set_InstanceIds(json.dumps([instance_id]))
if security_group_id:
request.set_SecurityGroupId(security_group_id)
request.set_PageNumber(page_number)
request.set_PageSize(page_size)
return _send_request(request)
```

续费云服务器

续费实例只支持包年包月的服务器类型，不支持按量付费的服务器，同时要求用户必须支持账号的余额支付或信用支付。执行 API 的时候将执行同步的扣费和订单生成。因此，执行 API 的时候必须保证您的账号有足够的资金支持自动扣费。

```
def _renew_instance_action(instance_id, period='1'):
request = RenewInstanceRequest()
request.set_Period(period)
request.set_InstanceId(instance_id)
response = _send_request(request)
logging.info('renew %s ready, output is %s ', instance_id, response)
```

续费实例将会自动完成扣费。在完成续费后，您可以根据InstanceId查询实例的资源到期时间。由于 API 为异步任务，查询资源到期时间可能需要延迟 10 秒才会变化。

开启云服务器自动续费

为了减少您的资源到期维护成本，针对包年包月的 ECS 实例，阿里云还推出了**自动续费功能**。 自动续费扣款日为服务器到期前第 7 天的 08:00:00。如果前一日执行自动扣费失败，将会继续下一日定时执行，直到完成扣费或者 7 天后到期资源锁定。您只需要保证自己的账号余额或者信用额度充足即可。

查询自动续费设置

您可以通过 OpenAPI 来查询和设置自动续费。该 API 仅支持包年包月的实例，按量付费的实例执行将会报错

。查询实例的自动续费状态支持一次最多查询 100 个包年包月的实例，多个实例 ID 以逗号连接。

DescribeInstanceAutoRenewAttribut 的入参为实例 ID.

- InstanceId：支持最多查询 100 个包年包月的实例，多个实例 ID 以逗号连接。

python # check the instances is renew or not def describe_auto_renew(instance_ids, expected_auto_renew=True): describe_request = DescribeInstanceAutoRenewAttributeRequest() describe_request.set_InstanceId(instance_ids) response_detail = _send_request(request=describe_request) failed_instance_ids = '' if response_detail is not None: attributes = response_detail.get('InstanceRenewAttributes').get('InstanceRenewAttribute') if attributes: for item in attributes: auto_renew_status = item.get('AutoRenewEnabled') if auto_renew_status != expected_auto_renew: failed_instance_ids += item.get('InstanceId') + ',' describe_auto_renew('i-1111,i-2222')
返回内容如下：

```
{"InstanceRenewAttributes":{"InstanceRenewAttribute":[{"Duration":0,"InstanceId":"i-1111","AutoRenewEnabled":false},{"Duration":0,"InstanceId":"i-2222","AutoRenewEnabled":false}]},"RequestId":"71FBB7A5-C793-4A0D-B17E-D6B426EA746A"}
```

如果设置自动续费，则返回的属性AutoRenewEnabled为 true，否则返回 false。

设置和取消云服务器的自动续费

设置自动续费有三个入参：

- InstanceId： 支持最多查询100个包年包月的实例，多个实例 ID 以逗号连接。
- Duration：支持 1、2、3、6、12，单位为月。
- AutoRenew：true/false， true为开启自动续费，false为取消自动续费。

python def setting_instance_auto_renew(instance_ids, auto_renew = True): logging.info('execute enable auto renew ' + instance_ids) request = ModifyInstanceAutoRenewAttributeRequest(); request.set_Duration(1); request.set_AutoRenew(auto_renew); request.set_InstanceId(instance_ids) _send_request(request)
执行成功返回 Response 如下:

python {"RequestId":"7DAC9984-AAB4-43EF-8FC7-7D74C57BE46D"}
续费成功后，您可以再执行一次查询。如果续费成功将返回续费时长以及是否开启自动续费。

python {"InstanceRenewAttributes":{"InstanceRenewAttribute":[{"Duration":1,"InstanceId":"i-1111","AutoRenewEnabled":true},{"Duration":1,"InstanceId":"i-2222","AutoRenewEnabled":true}]},"RequestId":"7F4D14B0-D0D2-48C7-B310-B1DF713D4331"}

完整的代码如下：

```
# coding=utf-8
```

```
# if the python sdk is not install using 'sudo pip install aliyun-python-sdk-ecs'
# if the python sdk is install using 'sudo pip install --upgrade aliyun-python-sdk-ecs'
# make sure the sdk version is 2.1.2, you can use command 'pip show aliyun-python-sdk-ecs' to check

import json
import logging

from aliyunsdkcore import client
from aliyunsdkecs.request.v20140526.DescribeInstanceAutoRenewAttributeRequest import \
DescribeInstanceAutoRenewAttributeRequest
from aliyunsdkecs.request.v20140526.DescribeInstancesRequest import DescribeInstancesRequest
from aliyunsdkecs.request.v20140526.ModifyInstanceAutoRenewAttributeRequest import \
ModifyInstanceAutoRenewAttributeRequest
from aliyunsdkecs.request.v20140526.RenewInstanceRequest import RenewInstanceRequest

logging.basicConfig(level=logging.INFO,
format='%(asctime)s %(filename)s[line:%(lineno)d] %(levelname)s %(message)s',
datefmt='%a, %d %b %Y %H:%M:%S')

clt = client.AcsClient('Your Access Key Id', 'Your Access Key Secrect', 'cn-beijing')

# data format in UTC, only support passed the value for minute, seconds is not support.
INSTANCE_EXPIRED_START_TIME_IN_UTC_STRING = '2017-01-22T00:00Z'
INSTANCE_EXPIRE_END_TIME_IN_UTC_STRING = '2017-01-28T00:00Z'

def renew_job(page_size=100, page_number=1, check_need_renew=True, security_group_id=None):
response = describe_need_renew_instance(page_size=page_size, page_number=page_number,
check_need_renew=check_need_renew,
security_group_id=security_group_id)
response_list = response.get('Instances').get('Instance')
logging.info("%s instances need to renew", str(response.get('TotalCount')))
if response_list > 0:
instance_ids = ''
for item in response_list:
instance_id = item.get('InstanceId')
instance_ids += instance_id + ','
renew_instance(instance_id=instance_id)
logging.info("%s execute renew action ready", instance_ids)

def describe_need_renew_instance(page_size=100, page_number=1, instance_id=None,
check_need_renew=True, security_group_id=None):
request = DescribeInstancesRequest()
if check_need_renew is True:
request.set_Filter3Key("ExpiredStartTime")
request.set_Filter3Value(INSTANCE_EXPIRED_START_TIME_IN_UTC_STRING)
request.set_Filter4Key("ExpiredEndTime")
request.set_Filter4Value(INSTANCE_EXPIRE_END_TIME_IN_UTC_STRING)
if instance_id is not None:
request.set_InstanceIds(json.dumps([instance_id]))
if security_group_id:
request.set_SecurityGroupId(security_group_id)
request.set_PageNumber(page_number)
request.set_PageSize(page_size)
return _send_request(request)

# check the instances is renew or not
```

```
def describe_instance_auto_renew_setting(instance_ids, expected_auto_renew=True):
describe_request = DescribeInstanceAutoRenewAttributeRequest()
describe_request.set_InstanceId(instance_ids)
response_detail = _send_request(request=describe_request)
failed_instance_ids = ''
if response_detail is not None:
attributes = response_detail.get('InstanceRenewAttributes').get('InstanceRenewAttribute')
if attributes:
for item in attributes:
auto_renew_status = item.get('AutoRenewEnabled')
if auto_renew_status != expected_auto_renew:
failed_instance_ids += item.get('InstanceId') + ','
if len(failed_instance_ids) > 0:
logging.error("instance %s auto renew not match expect %s.", failed_instance_ids,
expected_auto_renew)

def setting_instance_auto_renew(instance_ids, auto_renew=True):
logging.info('execute enable auto renew ' + instance_ids)
request = ModifyInstanceAutoRenewAttributeRequest();
request.set_Duration(1);
request.set_AutoRenew(auto_renew);
request.set_InstanceId(instance_ids)
_send_request(request)
describe_instance_auto_renew_setting(instance_ids, auto_renew)

# if using the instance id can be found means the instance is not renew successfully.
def check_instance_need_renew(instance_id):
response = describe_need_renew_instance(instance_id=instance_id)
if response is not None:
return response.get('TotalCount') == 1
return False

# 续费一个实例一个月
def renew_instance(instance_id, period='1'):
need_renew = check_instance_need_renew(instance_id)
if need_renew:
_renew_instance_action(instance_id=instance_id, period=period)
# describe_need_renew_instance(instance_id=instance_id, check_need_renew=False)

def _renew_instance_action(instance_id, period='1'):
request = RenewInstanceRequest()
request.set_Period(period)
request.set_InstanceId(instance_id)
response = _send_request(request)
logging.info('renew %s ready, output is %s ', instance_id, response)

def _send_request(request):
request.set_accept_format('json')
try:
response_str = clt.do_action(request)
logging.info(response_str)
response_detail = json.loads(response_str)
return response_detail
except Exception as e:
logging.error(e)
```

```
if __name__ == '__main__':
logging.info("Renew ECS Instance by OpenApi!")
# 查询在指定的时间范围内是否有需要续费的实例。
describe_need_renew_instance()
# 续费一个实例, 直接执行扣费
renew_instance('i-1111')
# 查询实例自动续费的状态
# describe_instance_auto_renew_setting('i-1111,i-2222')
# 设置实例自动续费
# setting_instance_auto_renew('i-1111,i-2222')
```

如您想了解 ECS 中 API 的其它操作，请参考 ECS中的API操作。

本文介绍了如何使用阿里云 ECS SDK 合理快速地创建并管理竞价实例。

准备工作

在执行操作之前，您需要：

- 了解能满足您业务要求的实例规格和地域。
- 熟悉了解阿里云 ECS SDK 的基础知识和调用方法。详细信息，请参考 SDK 使用说明。

注意：
竞价实例代码需要依赖的 ECS SDK 版本 4.2.0 以上。以 Java POM 依赖为例，修改引入 pom 依赖：

```
<dependency>
<groupId>com.aliyun</groupId>
<artifactId>aliyun-java-sdk-core</artifactId>
<version>3.2.8</version>
</dependency>
<dependency>
<groupId>com.aliyun</groupId>
<artifactId>aliyun-java-sdk-ecs</artifactId>
<version>4.2.0</version>
</dependency>
```

查询地域及可用的实例规格

使用 OpenAPI DescribeZones 查询可以创建竞价实例的地域以及可用的实例规格。示例代码如下所示。

OpenApiCaller.java

```
public class OpenApiCaller {
IClientProfile profile;
```

```
IAcsClient client;
public OpenApiCaller() {
profile = DefaultProfile.getProfile("cn-hangzhou", AKSUtil.accessKeyId, AKSUtil.accessKeySecret);
client = new DefaultAcsClient(profile);
}
public <T extends AcsResponse> T doAction(AcsRequest<T> var1) {
try {
return client.getAcsResponse(var1);
} catch (Throwable e) {
e.printStackTrace();
return null;
}
}
 }
```

DescribeZonesSample.java

```
public class DescribeZonesSample {
public static void main(String[] args) {
OpenApiCaller caller = new OpenApiCaller();
DescribeZonesRequest request = new DescribeZonesRequest();
request.setRegionId("cn-zhangjiakou");//可以通过 DescribeRegionsRequest 获取每个地域的 RegionId
request.setSpotStrategy("SpotWithPriceLimit");//对于查询是否可购买竞价实例此项必填
request.setInstanceChargeType("PostPaid");//后付费模式，竞价实例必须是后付费模式
DescribeZonesResponse response = caller.doAction(request);
System.out.println(JSON.toJSONString(response));
}
}
```

以下为输出结果，可以查看每个地域各个地域可供选择的实例规格、磁盘类型、网络类型等信息。

```
{
"requestId": "388D6321-E587-470C-8CFA-8985E2963DAE",
"zones": [
{
"localName": "华北 3 可用区 A",
"zoneId": "cn-zhangjiakou-a",
"availableDiskCategories": [
"cloud_ssd",
"cloud_efficiency"
],
"availableInstanceTypes": [
"ecs.e4.large",
"ecs.n4.4xlarge",
"ecs.sn2.medium",
"ecs.i1.2xlarge",
"ecs.se1.2xlarge",
"ecs.n4.xlarge",
"ecs.se1ne.2xlarge",
"ecs.se1.large",
"ecs.sn2.xlarge",
"ecs.se1ne.xlarge",
"ecs.xn4.small",
```

```
"ecs.sn2ne.4xlarge",
"ecs.se1ne.4xlarge",
"ecs.sn1.medium",
"ecs.n4.8xlarge",
"ecs.mn4.large",
"ecs.e4.2xlarge",
"ecs.mn4.2xlarge",
"ecs.mn4.8xlarge",
"ecs.n4.2xlarge",
"ecs.e4.xlarge",
"ecs.sn2ne.large",
"ecs.sn2ne.xlarge",
"ecs.sn1ne.large",
"ecs.n4.large",
"ecs.sn1.3xlarge",
"ecs.e4.4xlarge",
"ecs.sn1ne.2xlarge",
"ecs.e4.small",
"ecs.i1.4xlarge",
"ecs.se1.4xlarge",
"ecs.sn2ne.2xlarge",
"ecs.sn2.3xlarge",
"ecs.i1.xlarge",
"ecs.n4.small",
"ecs.sn1ne.4xlarge",
"ecs.mn4.4xlarge",
"ecs.sn1ne.xlarge",
"ecs.se1ne.large",
"ecs.sn2.large",
"ecs.i1-c5d1.4xlarge",
"ecs.sn1.xlarge",
"ecs.sn1.large",
"ecs.mn4.small",
"ecs.mn4.xlarge",
"ecs.se1.xlarge"
],
"availableResourceCreation": [
"VSwitch",
"IoOptimized",
"Instance",
"Disk"
],
"availableResources": [
{
"dataDiskCategories": [
"cloud_ssd",
"cloud_efficiency"
],
"instanceGenerations": [
"ecs-3",
"ecs-2"
],
"instanceTypeFamilies": [
"ecs.mn4",
"ecs.sn1",
"ecs.sn2",
```

```
"ecs.sn1ne",
"ecs.xn4",
"ecs.i1",
"ecs.se1",
"ecs.e4",
"ecs.n4",
"ecs.se1ne",
"ecs.sn2ne"
],
"instanceTypes": [
"ecs.n4.4xlarge",
"ecs.sn2.medium",
"ecs.i1.2xlarge",
"ecs.se1.2xlarge",
"ecs.n4.xlarge",
"ecs.se1ne.2xlarge",
"ecs.se1.large",
"ecs.sn2.xlarge",
"ecs.se1ne.xlarge",
"ecs.xn4.small",
"ecs.sn2ne.4xlarge",
"ecs.se1ne.4xlarge",
"ecs.sn1.medium",
"ecs.n4.8xlarge",
"ecs.mn4.large",
"ecs.mn4.2xlarge",
"ecs.mn4.8xlarge",
"ecs.n4.2xlarge",
"ecs.sn2ne.large",
"ecs.sn2ne.xlarge",
"ecs.sn1ne.large",
"ecs.n4.large",
"ecs.sn1.3xlarge",
"ecs.sn1ne.2xlarge",
"ecs.e4.small",
"ecs.i1.4xlarge",
"ecs.se1.4xlarge",
"ecs.sn2ne.2xlarge",
"ecs.sn2.3xlarge",
"ecs.i1.xlarge",
"ecs.n4.small",
"ecs.sn1ne.4xlarge",
"ecs.mn4.4xlarge",
"ecs.sn1ne.xlarge",
"ecs.se1ne.large",
"ecs.sn2.large",
"ecs.i1-c5d1.4xlarge",
"ecs.sn1.xlarge",
"ecs.sn1.large",
"ecs.mn4.small",
"ecs.mn4.xlarge",
"ecs.se1.xlarge"
],
"ioOptimized": true,
"networkTypes": [
"vpc"
```

```
],
"systemDiskCategories": [
"cloud_ssd",
"cloud_efficiency"
]
}
],
"availableVolumeCategories": [
"san_ssd",
"san_efficiency"
]
}
]
}
```

查询竞价实例的历史价格

使用 OpenAPI DescribeSpotPriceHistory 查询竞价实例最近 30 天的价格变化数据，获得最佳性价比的地域和规格信息，示例代码（DescribeSpotPriceHistorySample.java）如下。

```
public class DescribeSpotPriceHistorySample {
public static void main(String[] args) {
OpenApiCaller caller = new OpenApiCaller();
List<DescribeSpotPriceHistoryResponse.SpotPriceType> result = new
ArrayList<DescribeSpotPriceHistoryResponse.SpotPriceType>();
int offset = 0;
while (true) {
DescribeSpotPriceHistoryRequest request = new DescribeSpotPriceHistoryRequest();
request.setRegionId("cn-hangzhou");//可以通过 DescribeRegionsRequest 获取可购买的每个地域的 RegionId
request.setZoneId("cn-hangzhou-b");//可用区必填
request.setInstanceType("ecs.sn2.medium");//参考 DescribeZones 返回的实例类型，必填
request.setNetworkType("vpc");//参考 DescribeZones 返回的网络类型，必填
// request.setIoOptimized("optimized");//是否 I/O 优化类型，DescribeZones 返回的 IoOptimized，选填
// request.setStartTime("2017-09-20T08:45:08Z");//价格开始时间，选填，默认 3 天内数据
// request.setEndTime("2017-09-28T08:45:08Z");//价格结束时间，选填
request.setOffset(offset);
DescribeSpotPriceHistoryResponse response = caller.doAction(request);
if (response != null && response.getSpotPrices() != null) {
result.addAll(response.getSpotPrices());
}
if (response.getNextOffset() == null || response.getNextOffset() == 0) {
break;
} else {
offset = response.getNextOffset();
}
}
if (!result.isEmpty()) {
for (DescribeSpotPriceHistoryResponse.SpotPriceType spotPriceType : result) {
System.out.println(spotPriceType.getTimestamp() + "--->spotPrice:" + spotPriceType.getSpotPrice() + "----
>originPrice:" + spotPriceType.getOriginPrice());
}
System.out.println(result.size());
```

```
} else {
}
}
}
```

以下为返回结果示例。

```
2017-09-26T06:28:55Z--->spotPrice:0.24---->originPrice:1.2
2017-09-26T14:00:00Z--->spotPrice:0.36---->originPrice:1.2
2017-09-26T15:00:00Z--->spotPrice:0.24---->originPrice:1.2
2017-09-27T14:00:00Z--->spotPrice:0.36---->originPrice:1.2
2017-09-27T15:00:00Z--->spotPrice:0.24---->originPrice:1.2
2017-09-28T14:00:00Z--->spotPrice:0.36---->originPrice:1.2
2017-09-28T15:00:00Z--->spotPrice:0.24---->originPrice:1.2
2017-09-29T06:28:55Z--->spotPrice:0.24---->originPrice:1.2
```

重复以上步骤，您可以判断出该规格资源在可用区的价格变化趋势和最近价格。

> **说明**：
> 您可以通过平均价格和最高价格来决定是否可以接受购买该竞价实例，也可以通过更加合理的数据模型来分析历史价格数据，随时调整创建资源的规格和可用区，到达最佳性价比。

创建竞价实例

在创建竞价实例之前，您需要完成以下工作：

- 如果您使用自定义镜像创建竞价实例，必须已经 创建自定义镜像。
- 在控制台 创建安全组，或者使用 OpenAPI CreateSecurityGroup 创建安全组，并获取安全组 ID（SecurityGroupId）。
- 在控制台创建 VPC 和 交换机，或者使用 OpenAPI CreateVpc 和 CreateVSwitch 创建，并获取交换机 ID（VSwitchId）。

使用 OpenAPI CreateInstance 创建竞价实例。示例代码（CreateInstaneSample.java）如下。

```
public class CreateInstaneSample {
public static void main(String[] args) {
OpenApiCaller caller = new OpenApiCaller();
CreateInstanceRequest request = new CreateInstanceRequest();
request.setRegionId("cn-hangzhou");//地域 ID
request.setZoneId("cn-hangzhou-b"); //可用区ID
request.setSecurityGroupId("sg-bp11nhf94ivkdxwb2gd4");//提前创建的安全组 ID
request.setImageId("centos_7_03_64_20G_alibase_20170818.vhd");//建议选择您自己在该地域准备的自定义镜像
request.setVSwitchId("vsw-bp164cyonthfudn9kj5br");//VPC 类型需要交换机 ID
request.setInstanceType("ecs.sn2.medium"); //填入您询价后需要购买的规格
request.setIoOptimized("optimized");//参考 DescirbeZones 返回参数
request.setSystemDiskCategory("cloud_ssd");//参考 DescirbeZones 返回参数，多选一 cloud_ssd, cloud_efficiency, cloud
request.setSystemDiskSize(40);
request.setInstanceChargeType("PostPaid");//竞价实例必须后付费
```

```
request.setSpotStrategy("SpotWithPriceLimit");//SpotWithPriceLimit 出价模式，SpotAsPriceGo 不用出价，最高按量付费价格
request.setSpotPriceLimit(0.25F);//SpotWithPriceLimit 出价模式生效，您能接受的最高价格，单位为元每小时，必须高于当前的市场成交价才能成功
CreateInstanceResponse response = caller.doAction(request);
System.out.println(response.getInstanceId());
}
}
```

回收竞价实例

当竞价实例可能会因为价格因素或者市场供需变化而被强制回收。此时会触发竞价实例的中断。释放前，竞价实例会进入锁定状态，提示实例将会被自动回收。您可以针对实例回收状态自动化处理实例的退出逻辑。

目前，您可以通过以下任一种方式来获取竞价实例的中断锁定状态：

通过 实例元数据 获取。运行以下命令：

```
curl 'http://100.100.100.200/latest/meta-data/instance/spot/termination-time'
```

如果返回为空，说明实例可持续使用。如果返回类似 2015-01-05T18:02:00Z 格式的信息（UTC 时间），说明实例将于这个时间释放。

使用 OpenAPI DescribeInstances，根据返回的 OperationLocks 判断实例是否进入 **待回收** 状态。代码示例如下（DescribeInstancesSample.java）。

```
public class DescribeInstancesSample {
public static void main(String[] args) throws InterruptedException {
OpenApiCaller caller = new OpenApiCaller();
JSONArray allInstances = new JSONArray();
allInstances.addAll(Arrays.asList("i-bp18hgfai8ekoqwo0y2n", "i-bp1ecbyds24ij63w146c"));
while (!allInstances.isEmpty()) {
DescribeInstancesRequest request = new DescribeInstancesRequest();
request.setRegionId("cn-hangzhou");
request.setInstanceIds(allInstances.toJSONString());//指定实例 ID，效率最高
DescribeInstancesResponse response = caller.doAction(request);
List<DescribeInstancesResponse.Instance> instanceList = response.getInstances();
if (instanceList != null && !instanceList.isEmpty()) {
for (DescribeInstancesResponse.Instance instance : instanceList) {
System.out.println("result:instance:" + instance.getInstanceId() + ",az:" + instance.getZoneId());
if (instance.getOperationLocks() != null) {
for (DescribeInstancesResponse.Instance.LockReason lockReason : instance.getOperationLocks()) {
System.out.println("instance:" + instance.getInstanceId() + "-->lockReason:" +
lockReason.getLockReason() + ",vmStatus:" + instance.getStatus());
if ("Recycling".equals(lockReason.getLockReason())) {
//do your action
System.out.println("spot instance will be recycled immediately, instance id:" + instance.getInstanceId());
allInstances.remove(instance.getInstanceId());
}
```

```
}
}
}
System.out.print("try describeInstances again later ...");
Thread.sleep(2 * 60 * 1000);
} else {
break;
}
}
}
}
```

触发回收时输出结果如下：

```
instance:i-bp1ecbyds24ij63w146c-->lockReason:Recycling,vmStatus:Stopped
spot instance will be recycled immediately, instance id:i-bp1ecbyds24ij63w146c
```

其他操作

您还可以启动、停止、释放竞价实例。具体的操作与一般按量付费实例没有区别。可以参考 OpenAPI 文档：

- 启动实例：StartInstance
- 停止实例：StopInstance
- 释放实例：DeleteInstance

实例自定义数据

实例自定义脚本是阿里云 ECS 为用户提供的一种自定义实例启动行为的脚本，详细信息请参考阿里云线上帮助文档：实例自定义数据。

本文档主要介绍在创建实例时，您怎么使用这个自定义脚本来配置自己的 yum 源、NTP 服务和 DNS 服务。您也可以使用这个脚本自定义 Windows 实例的 NTP 服务和 DNS 服务。

场景

目前，实例启动时，阿里云会为实例自动配置预定义的 yum 源、NTP 服务和 DNS 服务。但是，您可能想拥有自己的 yum 源、NTP 服务和 DNS 服务，此时，您就可以使用实例自定义脚本来实现这个需求，此时您要注意：

- 如果您自定义了 yum 源，阿里云官方将不再提供 yum 源相关支持。
- 如果您自定义了 NTP 服务，阿里云官方不再提供相关时间服务。

配置方法

您可以按以下步骤实现上述场景需求。

登录 阿里云 ECS 控制台，创建实例，配置如下：

- **网络类型**：VPC 网络
- **实例规格**：I/O 优化实例
- **镜像**：公共镜像的 CentOS 7.2

在创建页面的 **自定义数据** 输入框中输入如下内容：

```
#!/bin/sh
# Modify DNS
echo "nameserver 8.8.8.8" | tee /etc/resolv.conf
# Modify yum repo and update
rm -rf /etc/yum.repos.d/*
touch myrepo.repo
echo "[base]" | tee /etc/yum.repos.d/myrepo.repo
echo "name=myrepo" | tee -a /etc/yum.repos.d/myrepo.repo
echo "baseurl=http://mirror.centos.org/centos" | tee -a /etc/yum.repos.d/myrepo.repo
echo "gpgcheck=0" | tee -a /etc/yum.repos.d/myrepo.repo
echo "enabled=1" | tee -a /etc/yum.repos.d/myrepo.repo
yum update -y
# Modify NTP Server
echo "server ntp1.aliyun.com" | tee /etc/ntp.conf
systemctl restart ntpd.service
```

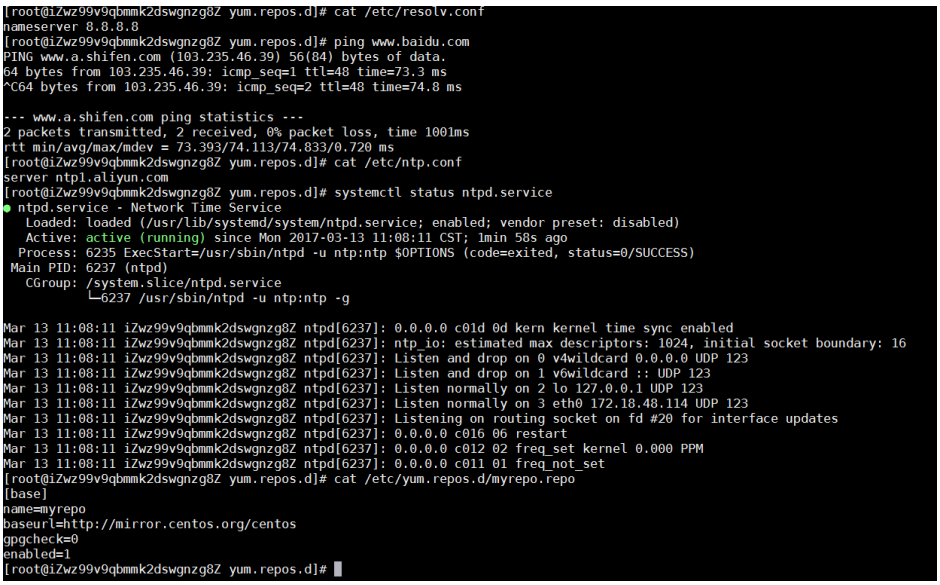
注意：

- 第一行必须是 #!/bin/sh，前面不能带空格。
- 全文不能有多余的空格和回车。
- 您可以根据实例情况定制具体的 DNS、NTP Server 和 yum 源 URL。
- 上述内容适用于 CentOS 7.2 镜像，如果是其他镜像，请根据需要修改实例自定义脚本。
- 您也可以使用 cloud config 类脚本更改 yum 源设置，但是不够灵活，不能适配阿里云对部分 yum 源进行预配置的情况。建议大家使用 script 类的脚本修改 yum 源设置。

根据需要完成 **安全设置**。

完成上述配置后，再单击 **立即购买**，并按页面指示开通实例。

实例购买完成后，您就可以登录实例查看具体的效果，如下图所示。



由上图可知，您已经成功自定义了 DNS 服务、NTP 服务和 yum 源。

实例自定义脚本是阿里云 ECS 为用户提供的一种自定义实例启动行为的脚本，详细信息请参考阿里云线上帮助文档：实例自定义数据。

本文档以 Linux 实例为例，说明在创建实例时，您应该怎样使用实例自定义脚本自定义实例的管理员账号。您也可以使用脚本自定义 Windows 实例的管理员账号。

场景

购买 ECS 实例时，如果您想达到如下效果，您就需要使用实例自定义脚本。

- 不使用 ECS 实例默认自带的 root 用户作为管理员。您可以在实例自定义脚本中自定义具体的禁用方式和禁用程度。
- 创建一个新的管理员账号，并自定义用户名。
- 新创建的管理员账号在管理该实例的时候只使用 SSH 密钥对进行远程登录，不使用用户密码。
- 该用户如果需要进行与管理员权限相关的操作，可在免密码的情况下使用 sudo 提权。

配置方法

您可以按以下步骤实现上述场景需求。

登录 阿里云 ECS 控制台，创建一个实例，配置如下：

- 网络类型：VPC 网络
- 实例规格：I/O 优化的实例
- 镜像：公共镜像的 CentOS 7.2

在创建页面的 **自定义数据** 输入框中输入如下内容：

```
#!/bin/sh
useradd test
echo "test ALL=(ALL) NOPASSWD:ALL" | tee -a /etc/sudoers
mkdir /home/test/.ssh
touch /home/test/.ssh/authorized_keys
echo "ssh-rsa
AAAAB3NzaC1yc2EAAAABJQAAAQEAhGqhEh/rGbIMCGItFVtYpsXPQrCaunGJKZVIWtINrGZwusLc290qDZ
93KCeb8o6X1Iby1Wm+psZY8THE+/BsXq0M0HzfkQZD2vXuhRb4xi1z98JHskX+0jnbjqYGY+Brgai9BvKDX
TTSyJtCYUnEKxvcK+d1ZwxbNuk2QZ0ryHESDbSaczlNFgFQEDxhCrvko+zWLjTVnomVUDhdMP2g6fZ0tgF
VwkJFV0bE7oob3NOVcrx2TyhfcAjA4M2/Ry7U2MFADDC+EVkpoVDm0SOT/hYJgaVM1xMDlSeE7kzX7yZ
bJLR1XAWV1xzZkNclY5w1kPnW8qMYuSwhpXzt4gsF0w== rsa-key-20170217" | tee -a
/home/test/.ssh/authorized_keys
```

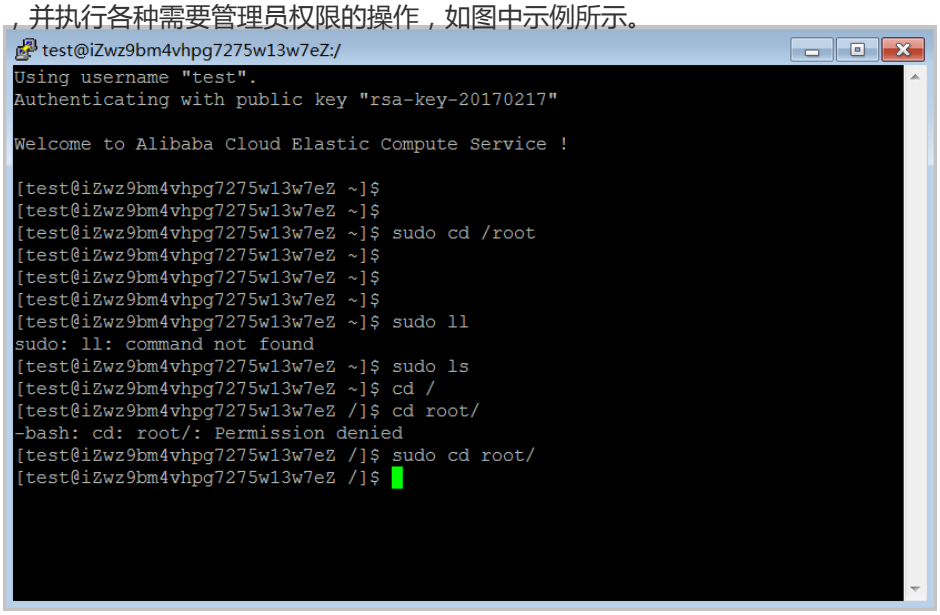
注意：

- 第一行必须是 #!/bin/sh，前面不能带空格。
- 全文不要有多余的空格和回车。
- 最后一行的密钥为您的公钥，您可以自定义。
- 如果需要做其他的配置，可以直接在脚本中添加。
- 示例脚本仅限于 CentOS 7.2 镜像，其他镜像请根据操作系统类型进行自定义修改。

在 **安全设置** 中选择 **创建后设置**。

完成上述配置后，再单击 **立即购买**，并按页面指示开通实例。

实例购买完成后，您可以使用自定义的 **test** 用户通过 SSH 私钥登录到实例中，同时也可以使用 sudo 提权，并执行各种需要管理员权限的操作，如图中示例所示。



概述

以往部署在 ECS 实例中的应用程序如果需要访问阿里云其他云产品，您通常需要借助 Access Key ID 和

Access Key Secret（下文简称 AK）来实现。AK 是您访问阿里云 API 的密钥，具有相应账号的完整权限。为了方便应用程序对 AK 的管理，您通常需要将 AK 保存在应用程序的配置文件中或以其他方式保存在 ECS 实例中，这在一定程度上增加了 AK 管理的复杂性，并且降低了 AK 的保密性。甚至，如果您需要实现多地域一致性部署，AK 会随着镜像以及使用镜像创建的实例扩散出去。这种情况下，当您需要更换 AK 时，您就需要逐台更新和重新部署实例和镜像。

现在借助于 ECS 实例 RAM 角色，您可以将 RAM 角色 和 ECS 实例关联起来，实例内部的应用程序可以通过 STS 临时凭证访问其他云产品。其中 STS 临时凭证由系统自动生成和更新，应用程序可以使用指定的 实例元数据 URL 获取 STS 临时凭证，无需特别管理。同时借助于 RAM，通过对角色和授权策略的管理，您可以达到不同实例对不同云产品或相同云产品具有各自访问权限的目的。

本文以部署在 ECS 实例上的 Python 访问 OSS 为例，详细介绍了如何借助 ECS 实例 RAM 角色，使实例内部的应用程序可以使用 STS 临时凭证访问其他云产品。

> **注意：**
> 为了方便您随本文样例快速入门，文档里所有操作均在 OpenAPI Explorer 完成。OpenAPI Explorer 通过已登录用户信息获取当前账号临时 AK，对当前账号发起线上资源操作，请谨慎操作。创建实例操作会产生费用。操作完成后请及时释放实例。

操作步骤

为了使 ECS 借助实例 RAM 角色，实现内部 Python 可以使用 STS 临时凭证访问 OSS，您需要完成以下步骤：

步骤 1. 创建 RAM 角色并配置授权策略
步骤 2. 指定 RAM 角色创建并设置 ECS 实例
步骤 3. 在实例内部访问实例元数据 URL 获取 STS 临时凭证
步骤 4. 基于临时凭证，使用 Python SDK 访问 OSS

步骤 1. 创建 RAM 角色并配置授权策略

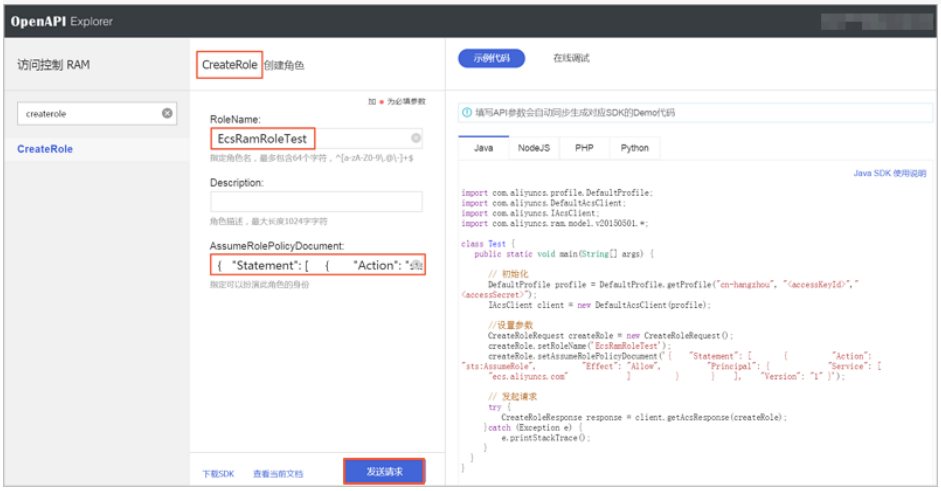
按以下步骤创建 RAM 角色并配置授权策略。

创建 RAM 角色。找到 OpenAPI Explorer RAM 产品下 CreateRole API。其中：

- **RoleName**：设置角色的名称。根据自己的需要填写，本示例中为 *EcsRamRoleTest*。
- **AssumeRolePolicyDocument**： 填写如下内容，表示该角色为一个服务角色，受信云服务（本示例中为 ECS）可以扮演该角色。

```
{
"Statement": [
{
"Action": "sts:AssumeRole",
"Effect": "Allow",
"Principal": {
```

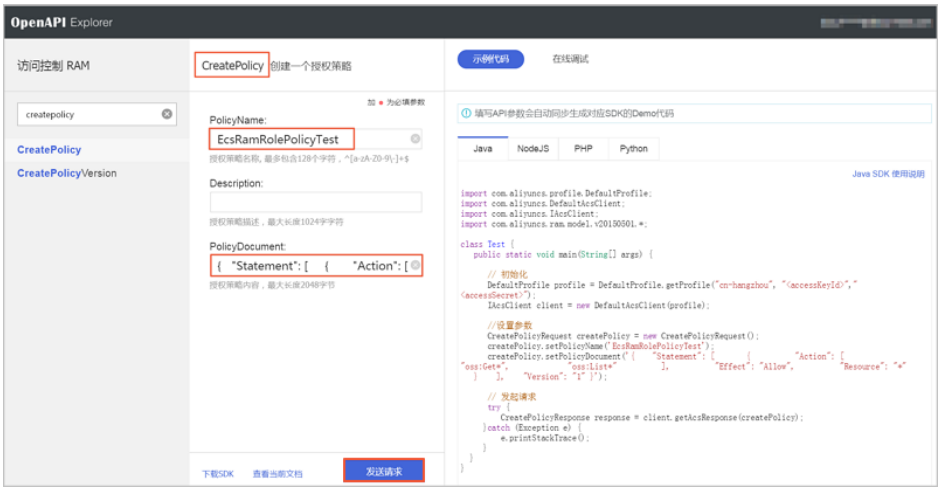
```
"Service": [
"ecs.aliyuncs.com"
]
}
}
],
"Version": "1"
}
```



创建授权策略。找到 OpenAPI Explorer RAM 产品下的 CreatePolicy API。其中：

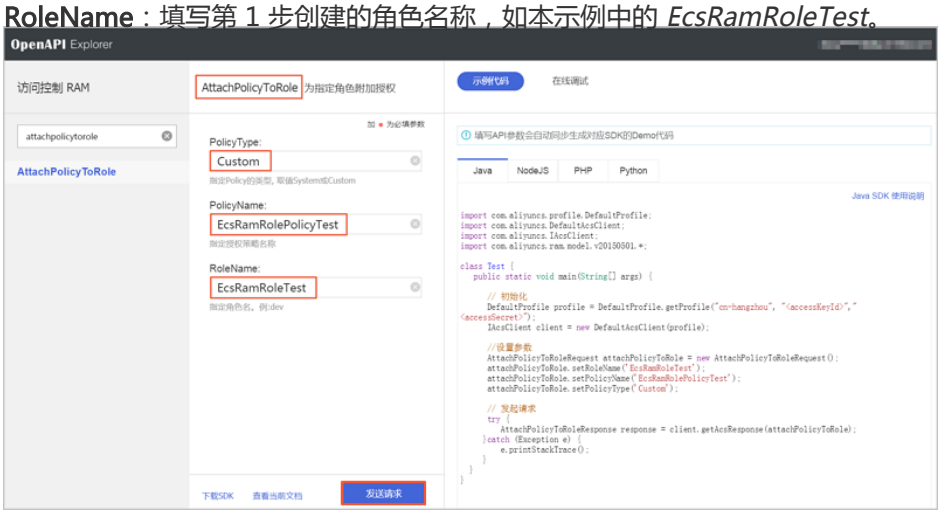
- **PolicyName**：设置授权策略的名称。本示例中为 *EcsRamRolePolicyTest*。
- **PolicyDocument**：输入授权策略内容。本示例中填写如下内容，表示该角色具有 OSS 只读权限。

```
{
"Statement": [
{
"Action": [
"oss:Get*",
"oss:List*"
],
"Effect": "Allow",
"Resource": "*"
}
],
"Version": "1"
}
```



为角色附加授权。找到 OpenAPI Explorer RAM 产品下 AttachPolicyToRole API。其中：

- **PolicyType**：填写 *Custom*。
- **PolicyName**：填写第 2 步创建的策略名称，如本示例中的 *EcsRamRolePolicyTest*。
- **RoleName**：填写第 1 步创建的角色名称，如本示例中的 *EcsRamRoleTest*。



步骤 2. 为 ECS 实例指定 RAM 角色

您可以通过以下任一种方式为 ECS 实例指定 RAM 角色：

- 将实例 RAM 角色附加到一个已有的 VPC 网络实例上
- 指定 RAM 角色创建并设置 ECS 实例

将实例 RAM 角色附加到一个已有的 VPC 网络实例上

您可以使用 ECS 的 AttachInstanceRamRole API 附加实例 RAM 角色到已有的 VPC 网络 ECS 实例授权访问，设置信息如下：

- **RegionId**：为实例所在的地域 ID。
- **RamRoleName**：RAM 角色的名称。本示例中为 *EcsRamRoleTest*。
- **InstanceIds**：需要附加实例 RAM 角色的 VPC 网络 ECS 实例 ID。本示例中为 [“i-

bXXXXXXXX”]。

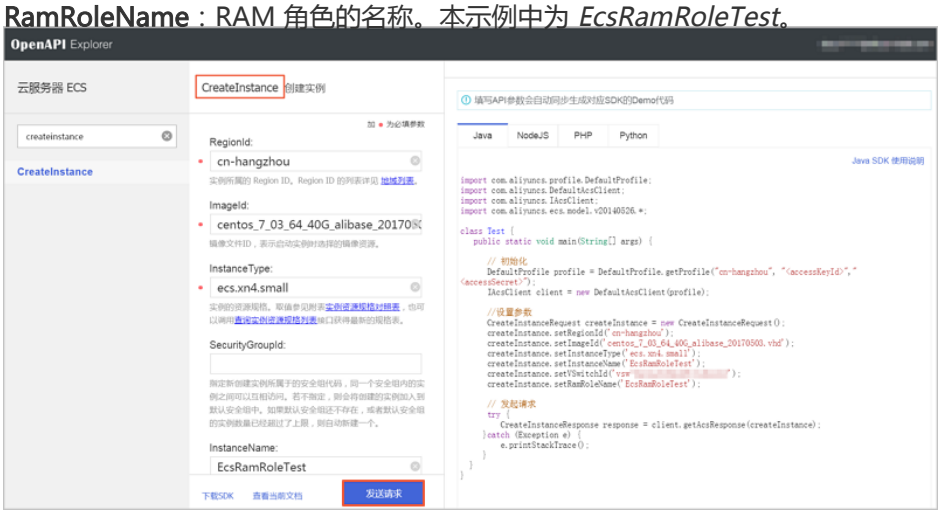
指定 RAM 角色创建并设置 ECS 实例

按以下步骤指定 RAM 角色创建并设置 ECS 实例。

创建实例。找到 OpenAPI Explorer ECS 产品下的 CreateInstance API，根据实际情况填写请求参数。必须填写的参数包括：

- **RegionId**：实例所在地域。本示例中为 *cn-hangzhou*。
- **ImageId**：实例的镜像。本示例中为 *centos_7_03_64_40G_alibase_20170503.vhd*。
- **InstanceType**：实例的规格。本示例中为 *ecs.xn4.small*。
- **VSwitchId**：实例所在的 VPC 网络虚拟交换机。因为 ECS 实例 RAM 角色目前只支持 VPC 网络的实例，所以 VSwitchId 是必需的。

RamRoleName：RAM 角色的名称。本示例中为 *EcsRamRoleTest*。



如果您希望授权子账号创建指定 RAM 角色的 ECS 实例，那么子账号除了拥有创建 ECS 实例的权限之外，还需要增加 PassRole 权限。所以，您需要创建一个如下所示的自定义授权策略并绑定到子账号上。如果是创建 ECS 实例，[ECS RAM Action] 可以是 ecs:CreateInstance，您也可以根据实际情况添加更多的权限，详见 RAM 中可对 ECS 资源进行授权的 Action。如果您需要为子账号授予所有 ECS 操作权限，[ECS RAM Action] 应该替换为 ecs:*。

```
{
"Statement": [
{
"Action": "[ECS RAM Action]",
"Resource": "*",
"Effect": "Allow"
},
{
"Action": "ram:PassRole",
"Resource": "*",
"Effect": "Allow"
],
```

```
"Version": "1"
}
```

设置密码并启动实例。

使用 API 或在控制台设置 ECS 实例能访问公网。关于在控制台设置 VPC 网络的 ECS 实例访问公网，请参考专有网络 VPC 用户指南弹性公网 IP。

步骤 3. 在实例内部访问实例元数据 URL 获取 STS 临时凭证

按以下步骤获取实例的 STS 临时凭证。

远程连接实例。

访问 http://100.100.100.200/latest/meta-data/ram/security-credentials/EcsRamRoleTest 获取 STS 临时凭证。路径最后一部分是 RAM 角色名称，您应替换为自己的创建的 RAM 角色名称。

> 本示例中使用 curl 命令访问上述 URL。如果您使用的是 Windows ECS 实例，参考 *ECS 用户指南* 的 实例元数据 获取 STS 临时凭证。

示例输出结果如下。

```
[root@local ~]# curl http://100.100.100.200/latest/meta-data/ram/security-credentials/EcsRamRoleTest
{
"AccessKeyId" : "STS.J8XXXXXXXXXX4",
"AccessKeySecret" : "9PjfXXXXXXXXXBf2XAW",
"Expiration" : "2017-06-09T09:17:19Z",
"SecurityToken" : "CAIXXXXXXXXXXXwmBkleCTkyI+",
"LastUpdated" : "2017-06-09T03:17:18Z",
"Code" : "Success"
}
```

步骤 4. 基于临时凭证，使用 Python SDK 访问 OSS

本示例中，我们基于 STS 临时凭证使用 Python SDK 列举实例所在地域的某个 OSS 存储空间（Bucket）里的 10 个文件。

前提条件

您已经远程连接到 ECS 实例。

您的 ECS 实例已经安装了 Python。如果您用的是 Linux ECS 实例，必须安装 pip。

您在实例所在的地域已经创建了存储空间（Bucket），并已经获取 Bucket 的名称和 Endpoint。本示例中

，Bucket 名称为 ramroletest，Endpoint 为 oss-cn-hangzhou.aliyuncs.com。

操作步骤

按以下步骤使用 Python SDK 访问 OSS。

运行命令 pip install oss2，安装 OSS Python SDK。

如果您用的是 Windows ECS 实例，*参考 对象存储 OSS SDK 参考* 的 安装 Python SDK。

执行下述命令进行测试，其中：

- oss2.StsAuth 中的 3 个参数分别对应于上述 URL 返回的 AccessKeyId、AccessKeySecret 和 SecurityToken。
- oss2.Bucket 中后 2 个参数是 Bucket 的名称和 Endpoint。

```
import oss2
from itertools import islice
auth = oss2.StsAuth(<AccessKeyId>, <AccessKeySecret>, <SecurityToken>)
bucket = oss2.Bucket(auth, <您的 Endpoint>, <您的 Bucket 名称>)
for b in islice(oss2.ObjectIterator(bucket), 10):
print(b.key)
```

示例输出结果如下。

```
[root@local ~]# python
Python 2.7.5 (default, Nov 6 2016, 00:28:07)
[GCC 4.8.5 20150623 (Red Hat 4.8.5-11)] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import oss2
>>> from itertools import islice
>>> auth = oss2.StsAuth("STS.J8XXXXXXXXXX4", "9PjfXXXXXXXXXBf2XAW",
"CAIXXXXXXXXXXXwmBkleCTkyI+")
>>> bucket = oss2.Bucket(auth, "oss-cn-hangzhou.aliyuncs.com", "ramroletest")
>>> for b in islice(oss2.ObjectIterator(bucket), 10):
... print(b.key)
...
ramroletest.txt
test.sh
```

FaaS 实例最佳实践

本文介绍如何在f1实例上使用OpenCL（Open Computing Language）制作镜像文件，并烧写到FPGA芯片中。

> **注意**：
> 强烈建议您使用RAM用户操作FPGA实例。为了防止意外操作，您需要让RAM用户仅执行必要的操作。在操作及下载FPGA镜像时，因为您需要从指定的OSS空间下载原始DCP工程，所以您需要为FaaS账号创建一个角色，并授予临时权限，让FaaS账号可以访问指定的OSS空间。如果需要对IP加密，需要授予RAM用户一些KMS相关的权限。如果需要做权限检查，还需要授予查看用户资源的权限。

前提条件

您已经 创建了f1实例。

如果使用RAM用户操作FPGA，需要 创建RAM用户 并 授权，创建RAM角色 并 授权。您必须 获取AccessKeyID和AccessKeySecret。

登录 ECS管理控制台，在f1实例的详情页上，获取实例ID。

操作步骤

按以下步骤在f1实例上使用OpenCL Example制作镜像文件，并烧写到FPGA芯片中。

第 1 步. 远程连接实例

远程连接Linux实例。

第 2 步. 安装基础环境

运行以下脚本安装基础环境。

```
source /opt/dcp1_0/script/f1_env_set.sh
```

第 3 步. 下载官方的OpenCL Example

按以下步骤下载官方的OpenCL Example。

创建并切换到/opt/tmp目录。

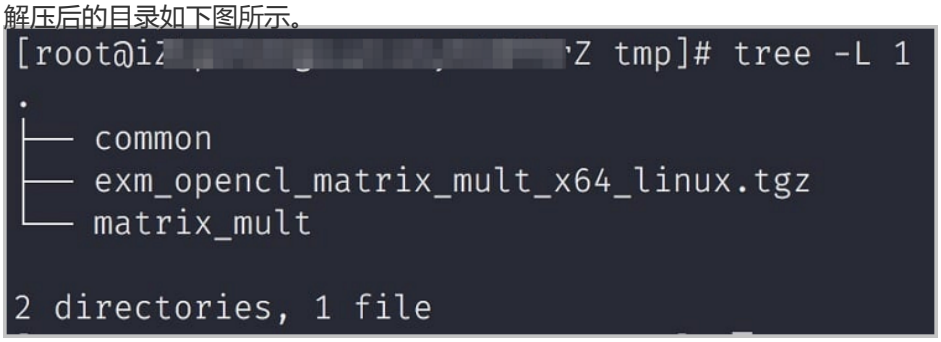
```
mkdir -p /opt/tmp
cd /opt/tmp
```

此时，您在/opt/tmp目录下。

依次执行以下命令下载并解压Example文件。

```
wget https://www.altera.com/content/dam/altera-www/global/en_US/others/support/examples/download/exm_opencl_matrix_mult_x64_linux.tgz
tar -zxvf exm_opencl_matrix_mult_x64_linux.tgz
```

解压后的目录如下图所示。



进入matrix_mult目录下，执行编译命令。

```
cd matrix_mult
aoc -v -g --report ./device/matrix_mult.cl
```

编译过程可能会持续数个小时，您可以再开一个console窗口，使用top命令监控系统占用，确定编译状态。

第 4 步. 上传配置文件

按以下步骤上传配置文件。

运行以下命令初始化faascmd。

```
# 如果需要，要添加环境变量及运行权限
export PATH=$PATH:/opt/dcp1_0/script/
chmod +x /opt/dcp1_0/script/faascmd
# 将hereIsMySecretId换为您的AccessKey ID，hereIsMySecretKey替换为您的AccessKey Secret
faascmd config --id=hereIsMySecretId --key=hereIsMySecretKey
# 将hereIsMyBucket换为华东1区的OSS的Bucket名
faascmd auth --bucket=hereIsMyBucket
```

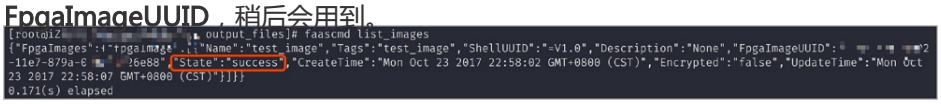
进入matrix_mult/output_files，上传配置文件。

```
cd matrix_mult/output_files # 此时您应该在/opt/tmp/matrix_mult/matrix_mult/output_files
faascmd upload_object --object=afu_fit.gbs --file=afu_fit.gbs
```

使用gbs制作FPGA镜像。

```
# 将hereIsFPGAImageName换为您的镜象名，将hereIsFPGAImageTag替换为您的镜像标签
faascmd create_image --object=afu_fit.gbs --fpgatype=intel --name=hereIsFPGAImageName --
tags=hereIsFPGAImageTag --encrypted=false --shell =V1.0
```

查看镜像是否制作成功：运行命令faascmd list_images。
返回结果里，如果显示 "State":"success"，表示镜像制作成功。请记录返回结果里显示的FpgaImageUUID，稍后会用到。



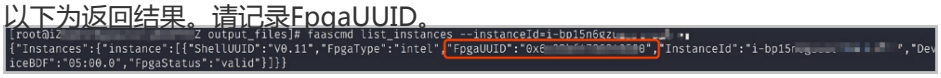
第 5 步. 下载镜像到f1实例

按以下步骤将镜像下载到f1实例。

运行命令获取FPGA ID。

```
# 将hereIsYourInstanceId替换为您的FPGA实例ID
faascmd list_instances --instanceId=hereIsYourInstanceId
```

以下为返回结果。请记录FpgaUUID。



运行命令下载镜像到f1实例。

```
# 将hereIsYourInstanceID替换为刚刚保存的实例ID；将hereIsFpgaUUID替换为上一条命令中记下的
FpgaUUID；将hereIsImageUUID替换为上一步记下的FpgaImageUUID
faascmd download_image --instanceId=hereIsYourInstanceID --fpgauuid=hereIsFpgaUUID --
fpgatype=intel --imageuuid=hereIsImageUUID --imagetype=afu --shell=V1.0
```

运行命令检查是否下载成功。

```
# 将hereIsYourInstanceID替换为刚刚保存的实例ID；将hereIsFpgaUUID替换为上一条命令中记下的
FpgaUUID；
faascmd fpga_status --fpgauuid=hereIsFpgaUUID --instanceId=hereIsYourInstanceID
```

如果返回结果里显示"TaskStatus":"valid"，说明下载成功。

第 6 步. 将FPGA镜像烧录到FPGA芯片

按以下步骤将FPGA镜像烧录到FPGA芯片。

打开第1步环境的窗口。如果已关闭，重新执行第1步操作。

运行命令配置OpenCL的运行环境。

```
sh /opt/dcp1_0/opencl/dcp_opencl_bsp/linux64/libexec/setup_permissions.sh
```

返回上上级目录。

```
cd ../.. # 此时您在/opt/tmp/matrix_mult
```

执行编译命令。

```
make
# 输出环境配置
export CL_CONTEXT_COMPILER_MODE_ALTERA=3
cp matrix_mult.aocx ./bin/matrix_mult.aocx
cd bin
host matrix_mult.aocx
```

当您看到如下输出时，说明配置完成。请注意，最后一行必须为Verification: PASS。

```
[root@iZbpXXXXXZ bin]# ./host matrix_mult.aocx
Matrix sizes:
A: 2048 x 1024
B: 1024 x 1024
C: 2048 x 1024
Initializing OpenCL
Platform: Intel(R) FPGA SDK for OpenCL(TM)
Using 1 device(s)
skx_fpga_dcp_ddr : SKX DCP FPGA OpenCL BSP (acl0)
Using AOCX: matrix_mult.aocx
Generating input matrices
Launching for device 0 (global size: 1024, 2048)
Time: 40.415 ms
Kernel time (device 0): 40.355 ms
Throughput: 106.27 GFLOPS
Computing reference output
```

```
Verifying
Verification: PASS
```

本文介绍了如何生成并下载自定义bitstream文件到一个指定的FPGA芯片里。

注意：
强烈建议您使用RAM用户操作FPGA实例。为了防止意外操作，您需要让RAM用户仅执行必要的操作。在操作FPGA镜像及下载时，因为您需要从指定的OSS空间下载原始DCP工程，所以您需要为FaaS账号创建一个角色，并授予临时权限，让FaaS账号可以访问指定的OSS空间。如果需要对IP加密，需要授予RAM用户一些KMS相关的权限。如果需要做权限检查，还需要授予查看用户资源的权限。

前提条件

创建f1实例。

您必须先 开通OSS服务，用于上传您自定义的bitstream文件。

如果需要加密bitstream，您还需要 开通密钥管理服务（KMS）。

使用RAM用户操作FPGA，需要 创建RAM用户 并 授权，创建RAM角色 并 授权。

操作步骤

您可以按以下步骤生成并下载bitstream。

第 1 步. 生成bitstream

上传工程到指定的OSS空间。这个空间必须与授权RAM用户里使用的OSS空间相同。

- 如果正在使用Intel FPGA，您需要先将最终的gbs文件上传到您的OSS空间里。
- 如果使用Xilinx FPGA，您需要先将布局布线后的DCP文件上传到您的OSS空间里。

调用Python SDK里的CreateFpgaImageTask接口，创建最终的bitstream。可以参考以下示例。

```
from aliyunsdkcore import client
clt = client.AcsClient(<您的AccessKeyID>,<您的AccessKeySecret>,'cn-hangzhou')
from aliyunsdkfaas.request.v20170824 import CreateFpgaImageTaskRequest
request = CreateFpgaImageTaskRequest.CreateFpgaImageTaskRequest()
request.set_Bucket(<DCP/bitstream所在的OSS bucket>)
request.set_Object(<DCP/bitstream在OSS中的object name>)
request.set_FpgaType(<Fpga类型>)
request.set_ShellUUID(<shell类型>)
request.set_Name(<给镜像取个方便记的名字>)
request.set_RoleArn(<给faas-admin账号创建的角色>)
```

```
request.set_Encrypted(<是否加密, True/False>)
request.set_KeyId(<如果加密，指定KMS中key的ID>)
result = clt.do_action_with_exception(request)
print result
```

调用Python SDK里的DescribeFpgaImages接口，查看bitstream是否已经生成。

> **说明**：
> CreateFpgaImageTask是个异步操作。您提交请求后，后台服务器会做一些安全检查，如果是Xilinx工程，后台服务器还需要从DCP工程生成bitstream，这需要一段时间。

可以参考以下示例。

```
from aliyunsdkcore import client
clt = client.AcsClient(<您的AccessKeyID>,<您的AccessKeySecret>,'cn-hangzhou')
from aliyunsdkfaas.request.v20170824 import DescribeFpgaImagesRequest
request = DescribeFpgaImagesRequest.DescribeFpgaImagesRequest()
result = clt.do_action_with_exception(request)
print result
```

第 2 步. 下载bitstream

生成bitstream后，您可以按以下步骤将bitstream下载到指定的FPGA。

您可以调用Python SDK里的DescribeFpgaInstances接口，查看当前实例下你的FpgaUUID（FPGA唯一识别标识）。请参考以下示例。

```
from aliyunsdkcore import client
clt = client.AcsClient(<您的AccessKeyID>,<您的AccessKeySecret>,'cn-hangzhou')
from aliyunsdkfaas.request.v20170824 import DescribeFpgaInstancesRequest
request = DescribeFpgaInstancesRequest.DescribeFpgaInstancesRequest()
request.set_InstanceId(<指定实例ID>)
request.set_RoleArn(<给faas-admin账号创建的角色>)
result = clt.do_action_with_exception(request)
print result
```

您的bitstream使用fpgaImageUUID作为唯一标识，通过调用DescribeFpgaImages接口可以查看您账号下所有bitstream的相关信息。请参考以下示例。

```
from aliyunsdkcore import client
clt = client.AcsClient(<您的AccessKeyID>,<您的AccessKeySecret>,'cn-hangzhou')
from aliyunsdkfaas.request.v20170824 import DescribeFpgaImagesRequest
request = DescribeFpgaImagesRequest.DescribeFpgaImagesRequest()
result = clt.do_action_with_exception(request)
print result
```

调用Python SDK里的LoadFpgaImageTask接口，将指定的bitstream下载到指定的FPGA里。请参考以下示例。

```
from aliyunsdkcore import client
clt = client.AcsClient(<您的AccessKeyID>,<您的AccessKeySecret>,'cn-hangzhou')
from aliyunsdkfaas.request.v20170824 import LoadFpgaImageTaskRequest
request = LoadFpgaImageTaskRequest.LoadFpgaImageTaskRequest()
request.set_InstanceId(<指定实例ID>)
request.set_FpgaUUID(<需要操作的FPGA>)
request.set_FpgaType(<Fpga类型>)
request.set_FpgaImageUUID(<需要下载的镜像UUID>)
request.set_FpgaImageType(<镜像类型>)
request.set_ShellUUID(<指定shell>)
request.set_RoleArn(<给faas-admin账号创建的角色>)
result = clt.do_action_with_exception(request)
print result
```

调用Python SDK里的DescribeLoadTaskStatus接口，查看下载是否成功。请参考以下示例。

```
from aliyunsdkcore import client
clt = client.AcsClient(<您的AccessKeyID>,<您的AccessKeySecret>,'cn-hangzhou')
from aliyunsdkfaas.request.v20170824 import DescribeLoadTaskStatusRequest
request = DescribeLoadTaskStatusRequest.DescribeLoadTaskStatusRequest()
request.set_FpgaUUID(<需要操作的FPGA>)
request.set_InstanceId(<指定实例ID>)
request.set_RoleArn(<给faas-admin账号创建的角色>)
result = clt.do_action_with_exception(request)
print result
```

至此，您已经将自定义的bitstream文件下载到指定的FPGA里。

本文描述如何使用f1 RTL（Register Transfer Level）。

注意：
强烈建议您使用RAM用户操作FPGA实例。为了防止意外操作，您需要让RAM用户仅执行必要的操作。在操作及下载FPGA镜像时，因为您需要从指定的OSS空间下载原始DCP工程，所以您需要为FaaS账号创建一个角色，并授予临时权限，让FaaS账号可以访问指定的OSS空间。如果需要对IP加密，需要授予RAM用户一些KMS相关的权限。如果需要做权限检查，还需要授予查看用户资源的权限。

前提条件

您已经 创建了f1实例。

使用RAM用户操作FPGA，需要 创建RAM用户 并 授权，创建RAM角色 并 授权。您必须 获取AccessKeyID和AccessKeySecret。

登录 ECS管理控制台，在f1实例的详情页上，获取实例ID。

操作步骤

按以下步骤使用f1 RTL。

第 1 步. 远程连接f1实例

远程连接Linux实例。

第 2 步. 配置基础环境

运行以下脚本配置基础环境。

```
source /opt/dcp1_0/script/f1_env_set.sh
```

第 3 步. 编译工程

运行以下命令：

```
cd /opt/dcp1_0/hw/green_bits/dma_afu/src
run.sh
```

说明：
编译时间很长。

第 4 步. 制作镜像

按以下步骤制作镜像：

运行命令初始化faascmd。

```
#如果需要，添加环境变及运权限
export PATH=$PATH:/opt/dcp1_0/script/
chmod +x /opt/dcp1_0/script/faascmd
# 将hereIsMySecretId替换为您的AccessKey ID，hereIsMySecretKey替换为您的AccessKey Secret
faascmd config --id=hereIsMySecretId --key=hereIsMySecretKey
# 将hereIsMyBucket换为华东1区的OSS的Bucket名
faascmd auth --bucket=hereIsMyBucket
```

确认在/opt/dcp1_0/hw/green_bits/dma_afu/src目录下，运行以下命令上传abs文件。

```
faascmd upload_object --object=dma_afu.gbs --file=dma_afu.gbs
```

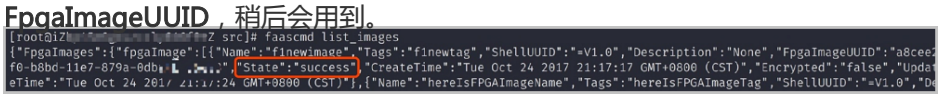
运行以下命令制作镜像。

```
# 将hereIsYourImageName替换为您的镜像名
faascmd create_image --object=dma_afu.gbs --fpgatype=intel --name=hereIsYourImageName --
tags=hereIsYourImageTag --encrypted=false --shell =V1.0
```

第 5 步. 下载镜像

按以下步骤下载镜像到f1实例：

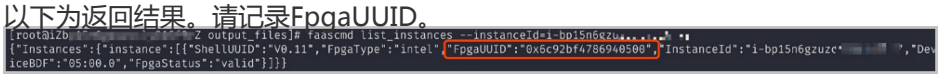
查看镜像是否制作成功：运行命令 faascmd list_images。
返回结果里，如果出现"State":"success"，表示镜像制作成功。请记录返回结果里显示的FpgaImageUUID，稍后会用到。



运行命令获取FPGA ID。

```
# 将hereIsYourInstanceId替换为您的f1实例ID
faascmd list_instances --instanceId=hereIsYourInstanceId
```

以下为返回结果。请记录FpgaUUID。



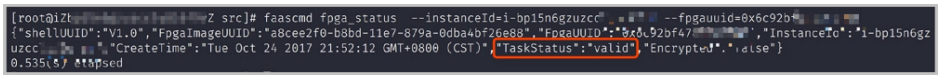
运行命令下载FPGA镜像到f1实例。

```
# 将hereIsYourInstanceID替换为刚刚保存的实例ID；将hereIsFpgaUUID替换为上一条命令中记下的
FpgaUUID；将hereIsImageUUID替换为上一步记下的FpgaImageUUID
faascmd download_image --instanceId=hereIsYourInstanceID --fpgauuid=hereIsFpgaUUID --
fpgatype=intel --imageuuid=hereIsImageUUID --imagetype=afu --shell=V1.0
```

运行命令检查是否下载成功。

```
# 将hereIsYourInstanceID替换为刚刚保存的实例ID；将hereIsFpgaUUID替换为上一条命令中记下的
FpgaUUID；
faascmd fpga_status --instanceId=hereIsYourInstanceID --fpgauuid=hereIsFpgaUUID
```

如果返回结果里出现"TaskStatus":"valid"时，且FpgaImageUUID和下载镜像时的FpgaImageUUID一致，说明下载成功。



第 6 步. 测试

依次运行以下命令。

```
cd /opt/dcp1_0/hw/green_bits/dma_afu/src/sw
make
./fpga_dma_test use_ase=0
```

如果您看到如图所示的输出结果，说明测试完成。

```
[root@iZ                Z sw]# ./fpga_dma_test use_ase=0
Running test in HW mode
Buffer Verification Success!
Buffer Verification Success!
Running DDR sweep test
Allocated test buffer
Fill test buffer
DDR Sweep Host to FPGA
Measured bandwidth = 5726.623061 Megabytes/sec
Clear buffer
DDR Sweep FPGA to Host
Measured bandwidth = 4473.924267 Megabytes/sec
Verifying buffer..
Buffer Verification Success!
```

注意：
如果没有开启Huge pages，运行以下命令启用Huge pages。

```
sudo bash -c "echo 20 > /sys/kernel/mm/hugepages/hugepages-2048kB/nr_hugepages"
```

本文介绍如何在 f2 实例上使用 OpenCL（Open Computing Language）制作镜像文件，并烧写到 FPGA 芯片中。

注意：
强烈建议您使用 RAM 用户操作 FaaS 实例。为了防止意外操作，您需要让 RAM 用户仅执行必要的操作。您需要为 FaaS 账号创建一个角色，并授予临时权限，让 FaaS 账号可以访问指定的 OSS 空间。

前提条件

您已经 创建了 f2 实例。

使用 RAM 用户操作 FPGA，需要 创建 RAM 用户 并 授权，创建 RAM 角色 并 授权。您必须 获取

AccessKeyID 和 AccessKeySecret。

您已经 开通 OSS 服务，并 创建了 Bucket。

您已经 登录 ECS 管理控制台，在 f2 实例的详情页上，获取实例 ID。

操作步骤

按以下步骤在 f2 实例上使用 OpenCL 制作镜像文件，并烧写到 FPGA 芯片中。

步骤 1. 配置环境

按以下步骤配置环境：

远程连接 f2 实例。

使用 vim 修改 /root/xbinst_oem/setup.sh：在第 5 行前加一个 #，注释掉 unset XILINX_SDX，再保存退出。

```
export XILINX_OPENCL=/root/2pf_acs_4ddr_normal_0906/xbinst_oem
export LD_LIBRARY_PATH=$XILINX_OPENCL/runtime/lib/x86_64:$LD_LIBRARY_PATH
export PATH=$XILINX_OPENCL/runtime/bin:$PATH
unset XILINX_SDACCEL
#unset XILINX_SDX
unset XCL_EMULATION_MODE
```

运行以下命令安装 Screen，用于后续的持续链接。

```
yum install screen -y
```

运行以下命令进入 Screen。

```
screen -S f2opencl
```

运行以下命令配置安全烧写环境。

```
source /root/xbinst_oem/F2_env_setup.sh
```

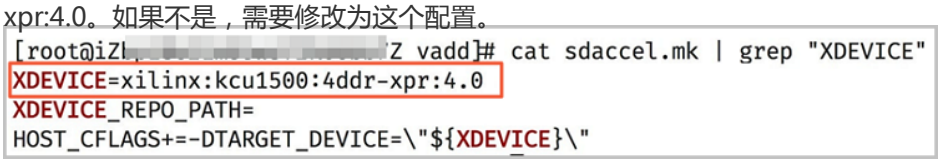
步骤 2. 编译二进制文件

按以下步骤编译二进制文件：

进入命令目录 。

```
cd /opt/Xilinx/SDx/2017.2/examples/vadd
```

运行命令 cat sdaccel.mk | grep "XDEVICE"，查看 XDEVICE 配置是否为xilinx:kcu1500:4ddr-xpr:4.0。如果不是，需要修改为这个配置。



使用 vim 修改 common.mk 文件。

```
vim ../common/common.mk
```

将第 63 行代码（如下所示）

```
CLCC_OPT += $(CLCC_OPT_LEVEL) ${DEVICE_REPO_OPT} --platform ${XDEVICE} -o ${XCLBIN}
${KERNEL_DEFS} ${KERNEL_INCS}
```

修改为

```
CLCC_OPT += $(CLCC_OPT_LEVEL) ${DEVICE_REPO_OPT} --platform ${XDEVICE} -o ${XCLBIN}
${KERNEL_DEFS} ${KERNEL_INCS} --xp param:compiler.acceleratorBinaryContent=dcp
```

说明：
参数可能在 60-62 行，由您的文件确定。

运行以下命令编译程序。

```
export XILINX_SDX=/opt/Xilinx/SDx/2017.2
make -f sdaccel.mk xbin_hw
```

如果您看到如下界面，说明二进制文件编译已经开始。编译过程可能会持续数个小时，请您耐心等待。

```
[root@iZ[redacted]'Z vadd]# make -f sdaccel.mk xbin_hw
make SDA_FLOW=hw xbin -f sdaccel.mk
make[1]: Entering directory `/opt/Xilinx/SDx/2017.2/examples/vadd'
xocc -t hw   --platform xilinx:kcu1500:4ddr-xpr:4.0 -o bin_vadd_hw.xclbin   --xp param
:compiler.acceleratorBinaryContent=dcp -s --kernel krnl_vadd krnl_vadd.cl

****** xocc v2017.2_sdx (64-bit)
  **** SW Build 1972098 on Wed Aug 23 11:34:38 MDT 2017
    ** Copyright 1986-2017 Xilinx, Inc. All Rights Reserved.

INFO: [XOCC 60-585] Compiling for hardware target
INFO: [XOCC 60-895]    Target platform: /opt/Xilinx/SDx/2017.2/platforms/xilinx_kcu150
0_4ddr-xpr_4_0/xilinx_kcu1500_4ddr-xpr_4_0.xpfm
```

步骤 3. 检查打包脚本

运行以下命令检查打包脚本是否存在。

```
file /root/xbinst_oem/sdaccel_package.sh
```

说明：
如果返回结果中包含 cannot open (No such file or directory)，说明不存在该文件，您需要手动下载打包脚本。

```
wget http://fpga-tools.oss-cn-shanghai.aliyuncs.com/sdaccel_package.sh -O
/root/xbinst_oem/sdaccel_package.sh
chmod a+x /root/xbinst_oem/sdaccel_package.sh
```

步骤 4. 制作镜像

按以下步骤制作镜像文件。

1. 运行命令配置 OSS 环境。

```
# 将此处的 hereIsMySecretId、hereIsMySecretKey、hereIsMyBucket 分别替换为您的 AccessKeyID、
AccessKeySecret 和 Bucket 名称
faascmd config --id=hereIsMySecretId --key=hereIsMySecretKey
faascmd auth --bucket=hereIsMyBucket
```

运行 ls，获取文件名。

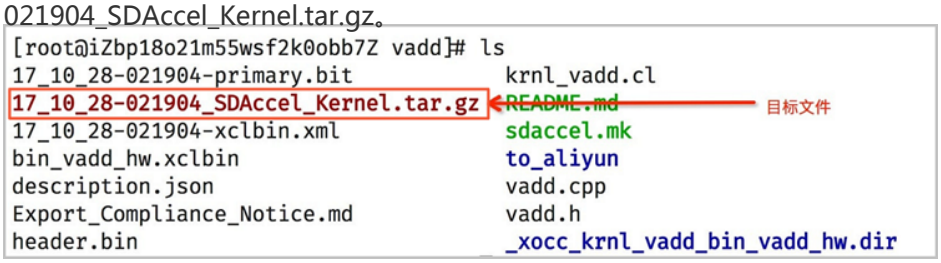
```
[root@iZbp18o21m55wsf2k0obb7Z vadd]# ls
bin_vadd_hw.xclbin          krnl_vadd.cl  vadd.cpp
description.json            README.md     vadd.h
Export_Compliance_Notice.md sdaccel.mk    _xocc_krnl_vadd_bin_vadd_hw.dir
```

打包二进制文件。

```
/root/xbinst_oem/sdaccel_package.sh -
```

```
xclbin=/opt/Xilinx/SDx/2017.2/examples/vadd/bin_vadd_hw.xclbin
```

打包完成后，在同一目录下，您会看到一个打包好的文件，如本示例中的 17_10_28-021904_SDAccel_Kernel.tar.gz。



运行命令将打包好的文件上传到您指定的 OSS Bucket 中。

```
# 将文件名改为打包好的文件名，您需要根据您的 ls 命令修改
faascmd upload_object --object=bit.tar.gz --file=bit.tar.gz
```

运行如下命令制作镜像。

```
# bit.tar.gz、hereIsFPGAImageName、hereIsFPGAImageTag 分别替换为刚创建的压缩包文件名、镜像名和镜像的 tag
faascmd create_image --object=bit.tar.gz --fpgatype=xilinx --name=hereIsFPGAImageName --tags=hereIsFPGAImageTag --encrypted=false --shell=V1.0
```

返回结果示例如下图所示。如果出现 "State":"queued"，说明这个任务已经加入队列，开始制作镜像。

说明：

制作镜像比较耗时。等待一段时间后，您可以运行以下命令，查看镜像状态。faascmd list_images返回结果里，如果出现 “State”:” success” ，说明镜制作成功。

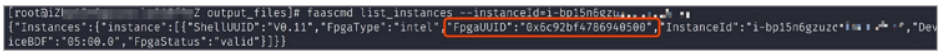
记录返回结果中的 FPGAImageUUID。

步骤 5. 烧写镜像

按以下步骤将镜像烧写入 FPGA 芯片。

运行以下命令获取 FpgaUUID。

```
# 将 hereIsYourInstanceId 替换为你的 FPGA 云服务器的实例 ID
faascmd list_instances --instanceId=hereIsYourInstanceId
```



说明：
记录返回结果中的 FpgaUUID。

运行以下命令下载镜像。

```
# 将 hereIsYourInstanceID 替换为这个 f2 实例的 ID，将 hereIsFpgaUUID 替换为您记录的 FpgaUUID，将 hereIsImageUUID 替换为您记录的 FpgaImageUUID
faascmd download_image --instanceId=hereIsYourInstanceID --fpgauuid=hereIsFpgaUUID --fpgatype=intel --imageuuid=hereIsImageUUID --imagetype=afu --shell=V1.0
```

在返回结果里，如果看到 "State":"committed"，说明镜像下载成功。

说明：
您也可以运行以下命令查看镜像是否下载成功。

```
# 将 hereIsYourInstanceID 替换为这个 f2 实例的 ID，将 hereIsFpgaUUID 替换为您记录的 FpgaUUID
faascmd fpga_status --instanceId=hereIsYourInstanceID --fpgauuid=hereIsFpgaUUID
```

返回结果里，如果看到 "TaskStatus":"valid"，而且 FpgaImageUUID 和下载镜像时的 FpgaImageUUID 一致，说明镜像下载正常。

步骤 6. 运行 Host 程序

执行以下命令运行 Host 程序。

```
make -f sdaccel.mk host
unset XILINX_SDX
./vadd bin_vadd_hw.xclbin
```

如果返回结果中出现 Test Passed，说明测试通过。

其他操作

这里介绍 FPGA 实例部分常用的操作。

任务	命令
查看帮助文档	make -f ./sdaccel.mk help
软件仿真	make -f ./sdaccel.mk run_cpu_em
硬件仿真	ake -f ./sdaccel.mk run_hw_em
只编译 host 代码	make -f ./sdaccel.mk host
编译生成可以下载的文件	make -f sdaccel.mk xbin_hw
清理工作目录	make -f sdaccel.mk clean
强力清除工作目录	make -f sdaccel.mk cleanall

说明：
sdx2017.2 在仿真时，device 需要用 xilinx_kcu1500_4ddr-xpr_4_0 。

本文描述如何在 f2 实例上使用 RTL（Register Transfer Level）设计。

注意：
强烈建议您使用 RAM 用户操作 FaaS 实例。为了防止意外操作，您需要让 RAM 用户仅执行必要的操作。在操作 FPGA 镜像及下载时，因为您需要从指定的 OSS 空间下载原始 DCP 工程，所以您需要为 FaaS 账号创建一个角色，并授予临时权限，让 FaaS 账号可以访问指定的 OSS 空间。如果需要对 IP 加密，需要授予 RAM 用户一些 KMS 相关的权限。如果需要做权限检查，还需要授予查看用户资源的权限。

前提条件

您已经 创建了 f2 实例。

使用 RAM 用户操作 FPGA，需要 创建 RAM 用户 并 授权，创建 RAM 角色 并 授权。您必须 获取 AccessKeyID 和 AccessKeySecret。

您已经 开通 OSS 服务，并 创建了 Bucket。

您已经 登录 ECS 管理控制台，在 f2 实例的详情页上，获取实例 ID。

操作步骤

按以下步骤使用 f2 RTL。

远程连接 f2 实例 。

> **说明**：
> 后续编译工程时可能需要耗时 2–3 小时。建议您使用 nohup 或者 VNC 连接，防止意外退出。

获取 RTL 参考设计。下载地址：http://faas-ref-design.oss-cn-hangzhou.aliyuncs.com/f2_ddr_prbs_rtl.tar。

运行 f2 的环境脚本。

```
source /root/xbinst_oem/F2_env_setup.sh
```

运行命令 vim /source/misc/xclbin.xml，修改 xclbin.xml 文件，将频率 300MHz 改成 200MHz。

运行以下命令编译工程。

```
sh compiling.sh
```

> **说明**：
> 编译工程可能需要耗时 2–3 小时。

运行以下命令将压缩包上传到您自己的 OSS Bucket。

```
# hereIsMySecretId 替换为您的 AccessKeyId，hereIsMySecretKey 替换为您的
AccessKeySecret，hereIsMyBucket 替换为您的 OSS Bucket 名称。
faascmd config --id=hereIsMySecretId --key=hereIsMySecretKey
faascmd auth --bucket=hereIsMyBucket
faascmd upload_object --object=bit.tar.gz --file=bit.tar.gz
```

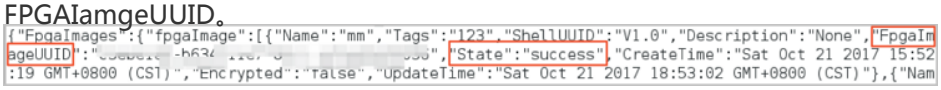
运行以下命令将存储在您指定 OSS Bucket 中的压缩包上传到 FaaS 管理单元 的 OSS 中。

```
faascmd create_image --object=bit.tar.gz --fpgatype=xilinx --name=hereIsFPGAImageName --
tags=hereIsFPGAImageTag --encrypted=false --shell=V1.0
```

运行命令查看 FPGA Image 是否已经可以下载。

```
faascmd list_images
```

在返回结果中，如果看到 "State":"success"，表示 FPGA Image 已经可以下载。您需要记录 FPGAIamgeUUID。



运行以下命令获取 FpgaUUID。

```
# hereIsYourInstanceId 替换为 f2 实例的 ID
faascmd list_instances --instanceId=hereIsYourInstanceId
```

在返回结果中，您需要记录 FpgaUUID。

运行以下命令下载 FPGA Image。

```
# hereIsYourInstanceId 替换为 f2 实例的 ID，hereIsFpgaUUID 替换为您获取的 FpgaUUID，hereIsImageUUID 替换为您获取的 FPGAImageUUID。
faascmd download_image --instanceId=hereIsYourInstanceId --fpgauuid=hereIsFpgaUUID --fpgatype=xilinx --imageuuid=hereIsImageUUID --imagetype=afu --shell=V1.0
```

运行以下命令查看镜像是否下载成功。

```
# hereIsFpgaUUID 替换为您获取的 FpgaUUID，hereIsYourInstanceId 替换为 f2 实例 ID。
faascmd fpga_status --fpgauuid=hereIsFpgaUUID --instanceId=hereIsYourInstanceId
```

以下为返回结果示例。如果显示的 FpgaImageUUID 与您获取的 FpgaImageUUID 一致，而且显示 "TaskStatus":"valid"，说明镜像下载成功。

运行 Host 程序，执行以下测试：

DDR 冒烟测试：

运行命令 ./reg_rw /dev/xdma0_user 0x14 w 1：在 0x14 地址写入 1，启动测试逻辑。

```
[root@F10A-KVM ddr_prbs_rtl]# ./reg_rw /dev/xdma0_user 0x14 w 1
argc = 5
device: /dev/xdma0_user
address: 0x00000014
access type: write
access width given.
access width: word (32-bits)
character device /dev/xdma0_user opened.
Memory mapped at address 0x7fb83ba97000.
Write 32-bits value 0x00000001 to 0x00000014 (0x0x7fb83ba97014)
```

运行命令 ./reg_rw /dev/xdma0_user 0x24 w：访问 0x24 地址。
以下是返回结果示例。如果返回 1，说明逻辑运行结束。

```
[root@F10A-KVM ddr_prbs_rtl]# ./reg_rw /dev/xdma0_user 0x24 w
argc = 4
device: /dev/xdma0_user
address: 0x00000024
access type: write
access width given.
access width: word (32-bits)
character device /dev/xdma0_user opened.
Memory mapped at address 0x7f94898a7000.
Read 32-bit value at address 0x00000024 (0x7f94898a7024): 0x00000001
```

运行命令 ./reg_rw /dev/xdma0_user 0x30 w：访问 0x30 地址。
以下是返回结果示例。如果返回 0，说明没有错误。

```
[root@F10A-KVM ddr_prbs_rtl]# ./reg_rw /dev/xdma0_user 0x30 w
argc = 4
device: /dev/xdma0_user
address: 0x00000030
access type: write
access width given.
access width: word (32-bits)
character device /dev/xdma0_user opened.
Memory mapped at address 0x7f65bd8d1000.
Read 32-bit value at address 0x00000030 (0x7f65bd8d1030): 0x00000000
```

DDR PRBS 测试：

运行命令 ./reg_rw /dev/xdma0_user 0x1c w 1：在 0x1c 地址写入 1，启动测试逻辑。等待运行一段时间后（您可以自行决定时间长短），运行命令 ./reg_rw /dev/xdma0_user 0x20 w 1：在 0x20 地址写入 1，停止测试逻辑。

```
[root@F10A-KVM ddr_prbs_rtl]# ./reg_rw /dev/xdma0_user 0x1c w 1
argc = 5
device: /dev/xdma0_user
address: 0x0000001c
access type: write
access width given.
access width: word (32-bits)
character device /dev/xdma0_user opened.
Memory mapped at address 0x7efe4aee3000.
Write 32-bits value 0x00000001 to 0x0000001c (0x0x7efe4aee301c)
[root@F10A-KVM ddr_prbs_rtl]# ./reg_rw /dev/xdma0_user 0x20 w 1
argc = 5
device: /dev/xdma0_user
address: 0x00000020
access type: write
access width given.
access width: word (32-bits)
character device /dev/xdma0_user opened.
Memory mapped at address 0x7fc65e7a9000.
Write 32-bits value 0x00000001 to 0x00000020 (0x0x7fc65e7a9020)
```

运行命令 ./reg_rw /dev/xdma0_user 0x28 w：访问 0x28 地址。
以下是返回结果示例。如果返回 1，说明逻辑成功结束。

```
[root@F10A-KVM ddr_prbs_rtl]# ./reg_rw /dev/xdma0_user 0x28 w
argc = 4
device: /dev/xdma0_user
address: 0x00000028
access type: write
access width given.
access width: word (32-bits)
character device /dev/xdma0_user opened.
Memory mapped at address 0x7fce66ff8000.
Read 32-bit value at address 0x00000028 (0x7fce66ff8028): 0x00000001
```

运行命令 ./reg_rw /dev/xdma0_user 0x30 w：访问 0x30 地址。

以下是返回结果示例。如果返回 0，表示没有错误。

```
[root@F10A-KVM ddr_prbs_rtl]# ./reg_rw /dev/xdma0_user 0x30 w
argc = 4
device: /dev/xdma0_user
address: 0x00000030
access type: write
access width given.
access width: word (32-bits)
character device /dev/xdma0_user opened.
Memory mapped at address 0x7f65bd8d1000.
Read 32-bit value at address 0x00000030 (0x7f65bd8d1030): 0x00000000
```

迁云实践之镜像迁移

镜像迁移，是指通过将源主机上的操作系统和应用程序及数据**镜像**到一个虚拟磁盘文件，并上传到阿里云镜像中心，成为自定义镜像。然后，通过此镜像启动一个和源主机配置相同的ECS实例，从而达到应用上云迁移的目的。

镜像迁移与手工重新部署迁移的技术对比分析

迁移技术类型		实现手段	优点	缺点
手工重新部署迁移		和物理主机部署方式一致。	通用性强	效率低，操作复杂，需要较多人工干预。
镜像迁移	冷迁移	通过工具直接镜像被迁移服务器主机，无法保障数据一致性。	简单、效率高、成功率高。	适用范围有限。
	热迁移（阿里云暂不支持）	通过镜像迁移工具部署在被迁移服务主机或远程连接的方式迁移，迁移过程可以保持数据实时同步。	简单、效率高、业务不中断。	适用范围有限。

迁移场景

目前，镜像迁移的场景来源有以下 4 种：

- 线下IDC机房的物理主机迁移到阿里云ECS主机实例。
- 传统虚拟化平台的虚拟主机迁移到阿里云ECS主机实例。
- 其他公有云的虚拟主机实例迁移到阿里云ECS主机实例。
- 阿里云ECS主机实例在各Region、各VPC中间进行迁移。

迁移类型

镜像迁移到阿里云，根据迁移类型分为以下 2 种：

P2V迁移

P2V指迁移物理服务器上的操作系统及其上的应用软件和数据到阿里云平台管理的ECS服务器中。这种迁移方式，主要是使用各种工具软件，把物理服务器上的系统状态和数据**镜像**到一个虚拟磁盘文件中。阿里云启动的时候在虚拟磁盘文件中**注入**存储硬件与网卡驱动程序，使之能够启动并运行。

V2V迁移

V2V是指从其他云平台或传统虚拟化平台的虚拟主机迁移到阿里云的ECS虚拟主机，比如 VMware 迁移到阿里云，AWS 迁移到阿里云等。

参考文档

与镜像迁移相关的其它内容，请参考如下文档：

应用迁云之镜像迁移 - 可行性评估

应用迁云之镜像迁移 - 工具介绍

应用迁云之镜像迁移 - 迁移流程和实践方法

应用迁云之镜像迁移 - 阿里云上跨VPC和区域、账号镜像迁移实践

目前，无论是P2V还是V2V的方式，迁移到阿里云还存在一些限制。在选择镜像迁移时，您需要对被迁移的服务器主机和镜像迁移的工具进行评估：

- 被迁移服务器主机操作系统类型、文件系统类型、服务器已使用空间大小。
- 镜像迁移工具支持导出的虚拟磁盘镜像文件格式。
- 兼容性要求及限制。

1. 被迁移服务器主机操作系统支持类型

Windows（32 和 64 位）

- Microsoft Windows Server 2012 R2（标准版）
- Microsoft Windows Server 2012（标准版、数据中心版）
- Microsoft Windows Server 2008 R2（标准版、数据中心版、企业版）
- Microsoft Windows Server 2008（标准版、数据中心版、企业版）
- 不支持 WinXP，Windows 8，Windows 10

Linux（64 位)

- CentOS 5,6,7
- Ubuntu 12,14,16

同时，部分工具支持如下类型：

- Debian 6,7
- OpenSUSE 13.1
- SUSE Linux 10,11,12
- CoreOS 681.2.0+

2. 被迁移服务器主机的文件系统类

目前， Windows操作系统的文件系统类型支持NTFS，Linux操作系统的文件系统类型支持ext3，ext4。

3. 被迁移服务器磁盘及空间使用情况

如果被迁移的服务器来自传统IDC、传统虚拟化平台以及其他云平台，只支持系统盘迁移，不支持数据盘的迁移；并且系统盘大小不能超过500GB。

被迁移的服务器本身在阿里云上，只是需要迁移到不同的region或者不同VPC中，是可以支持系统盘和数据盘进行同时迁移，同样系统盘大小不能超过500GB。

4. 兼容性要求及限制

Windows限制

- 导入的 Windows 镜像是提供 Windows 激活服务。
- 关闭防火墙。不关闭防火墙无法远程登录，需要放开3389端口。
- 关闭 UAC。

Linux 限制不支持开启 SELinux

- 关闭防火墙，默认打开22端口。
- 关闭或删除Network Manager。
- 导入的 Red Hat Enterprise Linux (RHEL) 镜像必须使用 BYOL 许可。需要自己向厂商购买产品序列号和服务。
- 不支持跟分区使用LVM。

其他限制

- 不支持多个网络接口。
- 不支持 IPv6 地址。

5. 镜像迁移工具支持导出的虚拟磁盘镜像文件格式

阿里云支持上传的镜像文件格式为RAW和VHD。其他格式的镜像文件都不支持，需要通过镜像文件格式转换工具进行转换。

目前，在镜像迁移过程中，主要使用镜像制作工具及镜像文件格式转换工具。镜像制作工具主要是把被迁移服务器主机的操作系统及应用程序和数据制作成镜像文件。因为不同的虚拟化平台的镜像文件或虚拟磁盘文件使用的格式不同，所以需要镜像格式转换工具对镜像文件格式进行转换来适配不同虚拟化平台。

当前镜像迁移到阿里云使用较多的工具有很多，比如Disk2VHD、DD等镜像文件制作工具以及XenConvert、StarWindConverter、qemu-img等镜像格式转换工具。它们都可以互相搭配使用，具体介绍如下所示。

1. 阿里云迁云工具

阿里云迁云工具，简称 **迁云工具**，是一个阿里云自主研发的能将计算机磁盘中的操作系统、应用程序以及应用数据等迁移到虚拟环境或是虚拟磁盘分区的便捷迁云工具。

阿里云迁云工具是专为平衡阿里云用户的线上线下服务器负载，或者各种不同云平台之间的负载而研制的。以其轻巧便捷的特点，阿里云迁云工具支持在线迁移物理机服务器、虚拟机以及其他云平台云主机至 ECS 管理控制台，实现统一部署资源的目的。更多详情，请参阅 **什么是阿里云迁云工具**。

2. QEMU

目前，阿里云只支持导入 RAW 或 VHD 格式的镜像文件。如果您要导入其他格式的镜像，可以使用 QEMU 的磁盘管理工具 qemu-img 转换成 VHD 或 RAW 格式的文件后再导入。

您可以使用 qemu-img 将 RAW、Qcow2、VMDK、VDI、VHD (vpc)、VHDX、qcow1 或 QED 格式的镜像转换成 VHD 或 RAW 格式的镜像，也可以使用它完成 RAW 和 VHD 格式的互相转换。更多详情，请参阅 **转换镜像格式**。

3. Disk2VHD

可用于将逻辑磁盘转换为 VHD 格式虚拟磁盘的实用工具。利用该工具，您可以轻松地将当前Windows系统中的C盘生成为一个 VHD 文件，然后上传到阿里云。

Disk2VHD能够运行在 Windows XP SP2，Windows Server 2003 SP1 或更高版本的Windows系统之上，并且支持 64位系统。

下载地址：http://publicread081.oss-cn-hangzhou.aliyuncs.com/Disk2vhd.zip-hangzhou.aliyuncs.com/Disk2vhd.zip

4. 命令工具

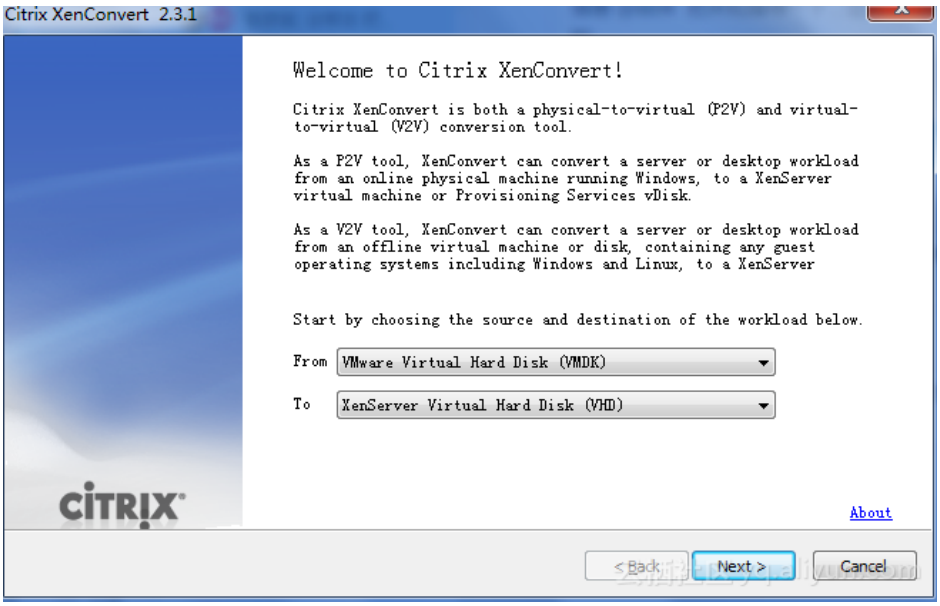
DD命令是Linux数据复制命令，通过DD可以将 Linux 及其系统盘映射打包为一个 RAW 格式的镜像文件。您可以使用DD制作镜像文件。

5. 镜像格式转换工具

5.1. XenConvert

XenConvert是用于实现物理到虚拟（P2V）转换的工具，另外此工具提供了镜像格式转换的功能，其中包括VMDK格式转换为VHD格式。

下载地址：http://publicread081.oss-cn-hangzhou.aliyuncs.com/XenConvert_Install_x64.exe

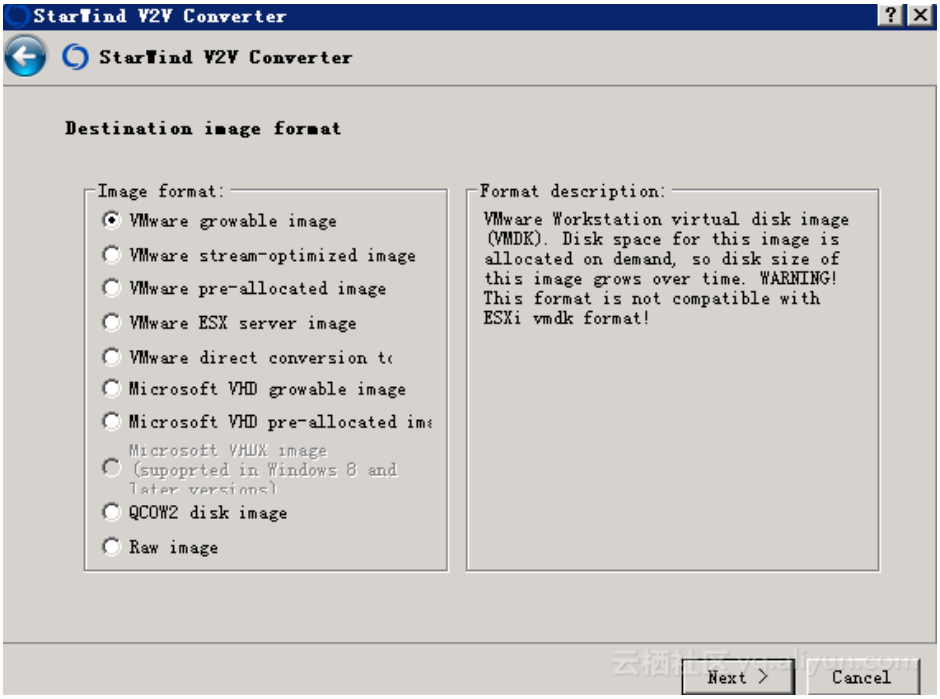


5.2. StarWind V2V Converter

StarWind V2V Converter 是一个格式转换软件，可以实现：

- 转换 VMDK 为 VHD
- 转换 VHD 为 VMDK

下载地址：http://publicread081.oss-cn-hangzhou.aliyuncs.com/starwindconverter.exe



迁移流程

1.镜像迁移可行性评估

当您选择镜像迁移前，需要对被迁移的服务器主机详细信息进行调研，并按照镜像迁移可行性评估小节中描述的要求及限制进行评估。评估是否可行及是否需要采用镜像迁移的方式来进行迁移。

如果被迁移服务器主机数量规模大、并且大多都带系统盘、网络条件不好，建议不要使用镜像迁移的方式。因为镜像文件都比较大，在此条件下进行镜像迁移反而会加大迁移的时间及人力成本。

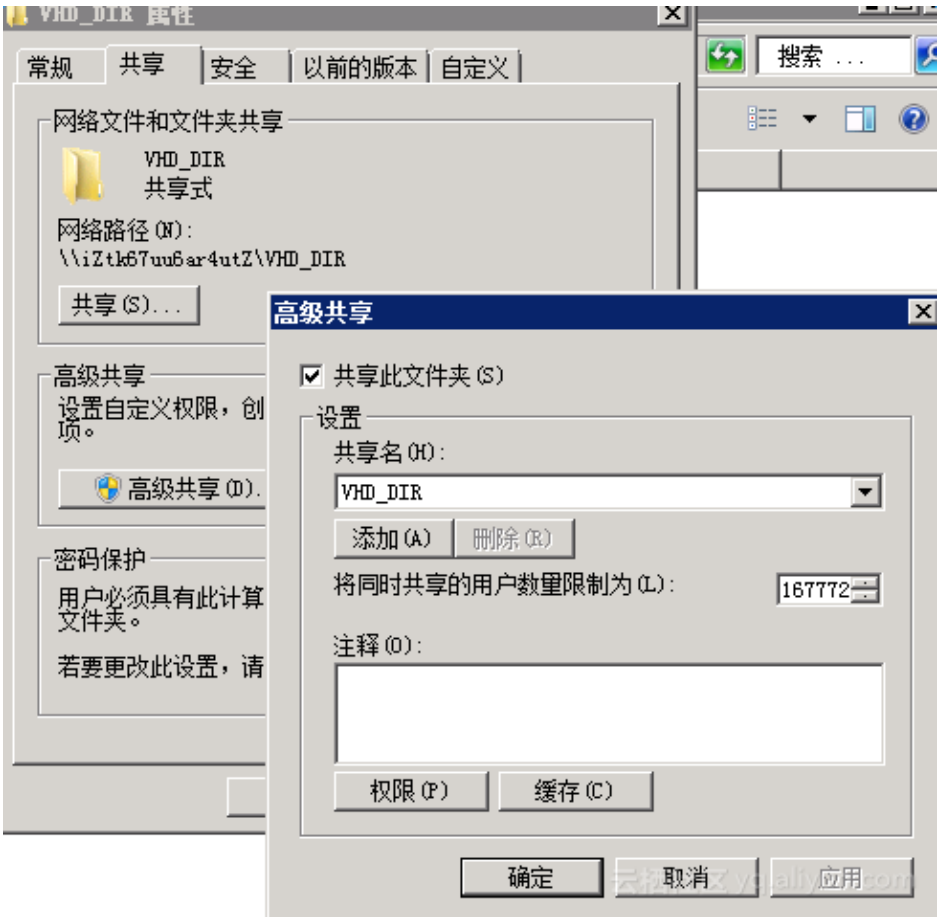
如果被迁移服务器主机中应用配置比较复杂、无人维护、网络条件好，建议您使用镜像迁移的方式。虽然数据盘不支持镜像迁移，但您可先把系统盘镜像迁移到阿里云，再采用文件同步的方式将数据盘数据同步到阿里云的数据盘中。

通常镜像迁移前需要一些准备工作，具体如下所示。

1.1 镜像文件存放公共目录准备

- Windows类

通过DISK2VHD工具对Windows操作系统的系统盘进行镜像文件制作。您可以把镜像文件存放地址输入公共目录地址，比如某台大容量空间的windows系统共享目录。



然后，在DISK2VHD的镜像文件保存地址中输入网络路径，比如\iZtk67uu6ar4utZ\VHD_DIR可以将镜像文件写入共享目录中进行统一管理。

- Linux类

通过DD工具对Linux操作系统的系统盘进行镜像文件制作的时候，可以把输出路径设置为一些挂载NFS的共享的目录，把镜像文件输出到统一的共享目录中。共享目录通常部署到镜像文件格式转换工具平台上。

环境搭建方法示例

一、环境示例

- 共享目录服务器端 CentOS6.5 192.168.0.10。
- 被迁移服务器端 CentOS6.5 192.168.0.11。

二、共享目录服务器端安装配置

先用rpm -qa命令查看所需安装包nfs-utils、rpcbind是否已经安装。

```
[root@local /]# rpm -qa | grep "rpcbind"
```

```
rpcbind-0.2.0-11.el6.x86_64
[root@local /]# rpm -qa | grep "nfs"
nfs-utils-1.2.3-39.el6.x86_64
nfs4-acl-tools-0.3.3-6.el6.x86_64
nfs-utils-lib-1.1.5-6.el6.x86_64
```

如查询结果如上，说明服务器自身已经安装了NFS；如果没有安装则用yum命令来安装。

```
[root@local /]# yum -y install nfs-utils rpcbind
```

创建共享目录。

```
[root@local /]# mkdir /sharestore
```

NFS共享文件路径配置。编辑/etc/exports添加下面一行，添加后保存退出。

```
 [root@local /]# vi /etc/exports
/sharestore *(rw,sync,no_root_squash)
```

启动NFS服务。先启动rpcbind，再启动nfs。如果服务器自身已经安装过NFS，就用restart重启两个服务。

```
[root@local /]# service rpcbind start</span> </pre>
Starting rpcbind: [ OK ]
[root@local /]# service nfs start
Starting NFS services: [ OK ]
Starting NFS quotas: [ OK ]
Starting NFS mountd: [ OK ]
Stopping RPC idmapd: [ OK ]
Starting RPC idmapd: [ OK ]
Starting NFS daemon: [ OK ]
[root@local /]
```

设置NFS服务开机自启动。

```
  [root@local /]# chkconfig rpcbind on
  [root@local /]# chkconfig nfs on
```

三、被迁移服务器端挂载配置

创建一个挂载点。

```
[root@localhost ~]# mkdir /mnt/store
```

挂载。

```
[root@localhost ~]# mount -t nfs 192.168.0.10:/sharestore /mnt/store
```

1.2 镜像文件格式转换工具平台准备

镜像文件格式转换平台搭建，主要是安装镜像格式转换工具并且需要保证平台磁盘空间有较大容量来保存镜像文件，对镜像文件进行统一存储和管理。具体容量空间大小需根据迁移镜像规模而定。在格式转换平台上，需要安装OSS工具。在镜像文件完成格式转换后，上传到用户具体账号下阿里云OSS对象存储中。

Windows类操作系统可以安装StarWindConverter工具来作为镜像文件格式转换平台的基础工具。

Linux类操作系统需安装qemu-img工具来作为镜像文件格式转换平台的基础工具。安装方法如下:

以CentOS为例:

```
yum install qemu-img
```

1.3 镜像导出前操作系统检查准备工作

Windows 系统关闭防火墙UAC、启用远程桌面

关闭防火墙。操作方法：选择 **开始>控制面板>Windows防火墙>打开和关闭防火墙**，选择关闭防火墙。

关闭UAC用户帐户控制。选择 **开始>运行**，输入MSCONFIG，打开 **系统配置>工具Tab**，更改UAC设置最低，重启系统后生效。

启用远程桌面。选择 **开始** > **计算机** > **属性** > **远程设置** > **启用远程桌面。**

系统关闭防火墙、Selinux、Network Manager

关闭Linux系统防火墙执行命令chkconfig iptables off重启生效。

关闭Selinux 修改/etc/selinux/config文件中的SELINUX=”” 为 disabled 重启生效。

关闭或删除Network Manager。

在/etc/fstab文件中去掉mount配置。

2. 镜像文件制作或导出

对于传统IDC的物理服务器主机或者其他云平台服务器主机，若为Windows类型，您可以使用DISK2VHD工具进行Windows系统C盘的镜像文件制作。

对于传统IDC的物理服务器主机或者其他云平台服务器主机的Linux类型，您可以使用DD工具进行Linux系统盘的导出。该工具导出的是RAW格式，镜像文件RAW文件一般都比较大和系统盘size一样大。RAW虽然可以直接上传到阿里云，但是建议使用qemu-img转换为VHD后上传，以节约网络传输时间。

3. 镜像格式转换。

对于有的云平台可以导出镜像文件而且基本是VHD的格式。这种情况下， 您可以省去镜像制作和格式转换的步骤。

在传统虚拟化平台，VMware类型的虚拟主机迁移不用镜像制作。目前，VMware虚拟主机底层虚拟磁盘文件为VMDK格式。您可以到ESX Server中把VMDK文件拷贝到镜像格式转换平台后直接转换。

VMDK转VHD

```
qemu-img convert –f vmdk vmdkfile.vmdk –O vpc vhdfile.vhd
```

RAW转VHD

```
qemu-img convert –f raw centos65.raw  –O vpc centos65.vhd
```

qemu-img convert 说明

```
qemu-img convert [-c] [-e] [-f format] filename [-O output_format] output_filename
```

当然，您也可以在windows系统中部署Xenconvert或者StarWindConverter工具来进行格式转换。镜像格式转换阶段主要是正对VMDK转VHDRAW转VHD。

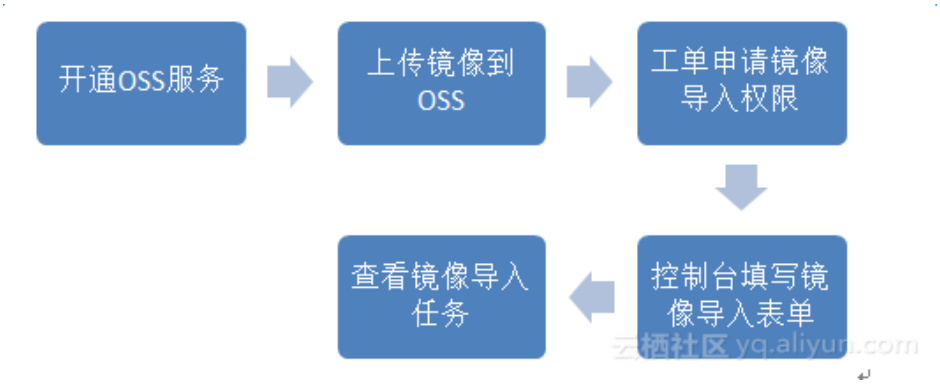
注意:

VMware的虚拟磁盘vmdk文件在创建的时候可以选择分割的方式，这样会导致一个虚拟机有N个虚拟磁盘文件。使用XenConvert转成VHD格式只能输入一个需要使用vmware-vdiskmanager.exe合并多个虚拟磁盘vmdk文件为一个vmdk文件。

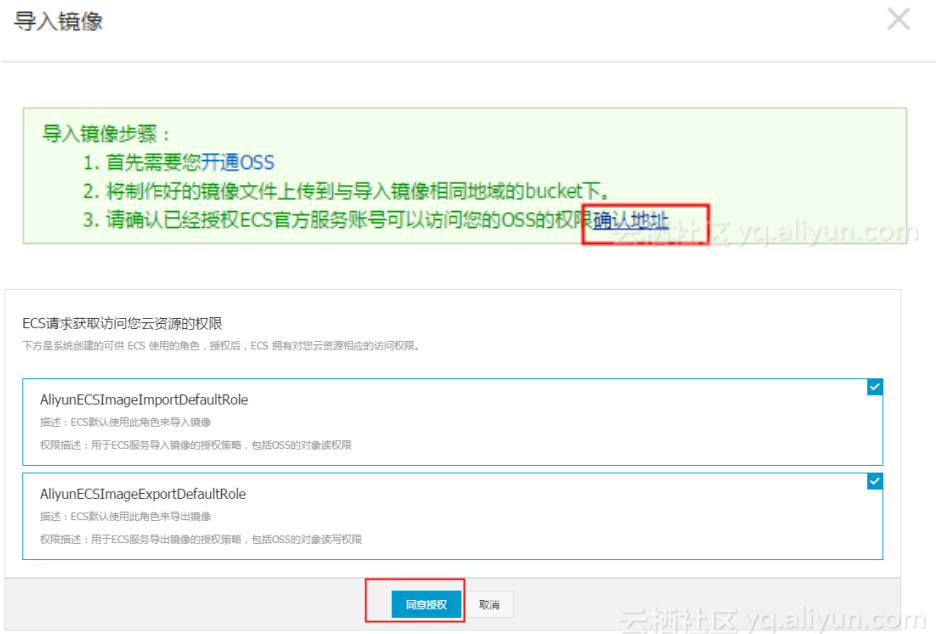
4. 镜像文件上传并设置为自定义镜像

在云下导出或制作好镜像后，需要上传的阿里云的镜像中心，上传过程中需要使用OSS服务。如果使用的阿里

云账号还没有开通OSS服务，请先开通OSS服务。使用OSS的第三方工具客户端OSS API 或者OSS SDK把制作好的文件上传到，和导入ECS用户自定义镜像相同地域的bucket里面，如对上传文件到OSS不熟悉，请参考 https://help.aliyun.com/document_detail/32185.html?spm=5176.doc32184.6.951.c6Ckyf。



镜像上传到OSS后，您可以在阿里云控制台发起工单申请ECS。导入镜像的权限并且主动把OSS的访问权限授权给ECS官方的服务账号。



授权完成后，进入阿里云ECS控制台。导入镜像前需要填写导入镜像信息表单。

表单属性	属性解释
地域	请选择您即将要部署应用的地域
镜像文件OSS地址	直接复制从OSS的控制台的Object对象的获取地址的内容。
镜像名称	长度为2-128个字符以大小写字母或中文开头可包含数字“.”“_”或“-“
系统盘大小	Windows系统盘大小取值40-500GB，Linux系统盘大小 20-500G。
系统架构	64位操作系统选择x86_64，32位操作系统选择i386
操作系统类型	Windows 或者 Linux
系统发行版	暂时支持的操作系统发行版。Windows支持 Windows Server 2003，2008，2012 和 Windows 7；Linux支持 CentOS，redhat，SUSE，Ubuntu Debian，Gentoo，FreeBSD，CoreOS。Other Linux请提交工单确认是否支持。如果您镜像的操作系统是根据Linux内核定制开发的，请发工单联系阿里云。
镜像格式	支持RAW和VHD两种格式，建议客户使用RAW格式，成功率会高很多。不支持使用qemu-image创建vhd格式的镜像。
镜像描述	填写镜像描述信息

在镜像导入过程中，通过任务管理找到该导入的镜像，您可以对导入的镜像进行取消。导入镜像需要耐心等待，一般需要数小时才能完成。完成的时间取决于镜像文件的大小和当前导入任务繁忙程度，您可以在导入地域

的镜像列表中看到这个镜像进度。

5. 根据镜像启动ECS实例

镜像导入到阿里云后，您可以进入阿里云ECS控制台，通过上传的镜像进行实例创建。在镜像选择的时候，镜像来源需要选择自定义镜像，您可以在自定义镜像列表看到导入的镜像。

启动完成后，您可以根据以下检查项列表来进入ECS实例进行相关检查。

Windows镜像实例检查列表

检查内容	说明
IP内网IP/外网ip 掩码 网关	1. 内网ip校验能通过另外一台vm ping通 2. 外网ip外网ping通
路由	正常访问外网
密码	administrator密码登录
hostname	计算机-属性-高级系统设置-计算机名 修改后重启计算机
DNS	ping DNS服务是否能ping通/是否能正常访问外网
默认网关	正常访问外网
host文件	位于:C:\Windows\System32\drivers\etc 测试域名绑定
挂载数据磁盘	挂载磁盘是否成功，格式化磁盘是否成功 是否能正确写入文件check，是否存在写保护
ntp	校验机器时间
KMS	1. 运行输入框中输入“Slmgr.vbs -dlv”命令并回车

	2. 查看批量激活过期时间
注入启动AliyunService进程以及XEN或KVM模块	任务管理器查看是否存在以下进程shutdownmon老版本叫shutdownmon/AliyunService

Linux镜像实例检查列表

检查内容	说明
ip 掩码 网关公私网卡	1. 内网ip校验能通过另外一台vm ping通 2. 外网ip外网ping通
路由	正常访问外网
密码	root密码
hostname	修改hostname
dns	ping DNS服务是否能ping通/是否能正常访问外网
默认网关	正常访问外网
hos文件	/etc/sysconfig/network修改hostname需要重启reboot
ssh key	/etc/ssh/ssh_host_key(一般不会修改)
挂载数据磁盘	mount磁盘是否成功格式化磁盘是否成功 是否能正确写入文件check是否存在写保护
ntp	查看服务器时间
yum/apt源	自动安装yum或apt软件
注入启动gshell进程以及XEN或KVM模块	ps -ef | grep gshell | grep -v grep | wc -l

目前，阿里云上的镜像迁移主要需求场景如下:

跨VPC迁移ECS实例比如从VPC A迁移到VPC B环境中。

跨区域迁移ECS实例比如从上海区域迁移到杭州区域。

跨账号迁移ECS实例比如从账号A迁移到账户B。

阿里云提供ECS实例快照和自定义镜像，支持系统盘和数据盘的功能，并且自定义镜像可以跨区域复制和共享给其他账号使用。基于这些功能特性，您可以实现跨VPC、跨区域、跨账号的镜像迁移。

跨VPC镜像迁移流程如下:

跨区域镜像迁移流程如下:

跨账号镜像迁移流程如下:

1. ECS 快照生成

所谓快照，就是某一个时间点上某一个磁盘的数据备份。需要注意的是，如果要保持数据的一致性，需要通过停机或停止服务的方式进行快照。

ECS快照操作流程如下：

登录云服务器管理控制台。

单击实例所在的地域，然后单击左侧导航的实例。单击实例的名称或在实例右侧单击**管理**。

单击左侧的本实例磁盘，对系统盘和数据盘进行创建快照。

2. 创建自定义镜像

镜像是云服务器 ECS 实例运行环境的模板。一般包括操作系统和预装的软件。

自定义镜像的来源渠道：

- 根据现有的云服务器 ECS 实例的快照创建自定义镜像。

- 把线下环境的镜像文件导入到ECS的集群中生成一个自定义镜像。

操作步骤：

进入ECS实例，单击**管理**。

单击左侧本实例快照，确定快照的磁盘属性是系统盘，数据盘不能单独用于创建镜像。

单击**创建自定义镜像**。

3.镜像跨区域复制

当前跨地域复制镜像处于公测状态，如需使用，可以提交工单申请白名单。工单中注明需复制镜像的总大小信息。自定义镜像是不能跨地域使用的。但是如果需要跨地域使用自定义镜像，可以通过复制镜像的方式，把当前地域的自定义镜像复制到其他地域进行镜像迁移复制。

复制镜像需要通过网络把源地域的镜像文件传输到目标地域。复制的时间取决于网络传输速度和任务队列的排队数量。

复制自动义镜像的步骤如下：

登录云服务器管理控制台。

单击左侧导航中的镜像可以看到镜像列表。

选择页面顶部的地域。

选中需要复制的镜像，镜像类型必须是自定义镜像单击复制镜像。在弹出的对话框中，您可以看到选中镜像的ID。

选择需要复制镜像的目标地域。

输入目标镜像的名称和描述。

单击确定镜像复制任务就创建成功了。

4. 镜像共享

在阿里云，您可以把自己的自定义镜像共享给其他用户。该用户可以通过管理控制台或 ECS API 查询到其他账号共享到本账号的共享镜像列表。被共享用户可以使用其他账号共享的镜像创建 ECS 实例和更换系统盘。

分享镜像的步骤如下:

登录云服务器管理控制台。

单击左侧导航中的镜像，您可以看到镜像列表。

选择页面顶部的地域。

选中需要复制的镜像。镜像类型必须是自定义镜像，单击**共享镜像**。

在弹出的对话框中，选择账号类型，并输入阿里云账号。有以下2种账号类型：

阿里云账号输入要共享给其他用户的阿里云账号登录账号。

AliyunID 输入要共享给其他的阿里云账号ID。 AliyunID 可以从阿里云官网的用户中心获取。选择 **账号管理** > **安全设置** > **账号ID**。可通过下面链接直接登录访问 https://account.console.aliyun.com/#/secure。

单击**共享镜像**完成自定义镜像的共享。

迁云实践之使用工具迁云

阿里云迁云工具，简称 **迁云工具**，是一个阿里云自主研发的能将计算机磁盘中的操作系统、应用程序以及应用数据等迁移到虚拟环境或是虚拟磁盘分区的便捷迁云工具。

阿里云迁云工具是专为平衡阿里云用户的线上线下服务器负载，或者各种不同云平台之间的负载而研制的。以其轻巧便捷的特点，阿里云迁云工具支持在线迁移物理机服务器、虚拟机、以及其他云平台云主机至 ECS 管理控制台，实现统一部署资源的目的。

应用场景

阿里云迁云工具可适用于以下场景：

迁移线下物理机服务器至 ECS 控制台。

迁移线下虚拟机服务器至 ECS 控制台。

迁移其他云平台的云主机至 ECS 控制台，包括 AWS（Amazon Web Services）、Microsoft Azure、腾讯云、华为云等云平台。

适用的操作系统

阿里云迁云工具支持迁移下列 64 位的 Windows 和 Linux 操作系统，包括：

- Windows
  - Windows Server 2003
  - Windows Server 2008
  - Windows Server 2012
  - Windows Server 2016
- Linux
  - CentOS 5/6/7
  - Ubuntu 12/14/16
  - Debian 7/8/9
  - Red Hat 6/7
  - SUSE 11.4
  - OpenSUSE 13.1
  - Gentoo 13.0

当您迁移其他没有包含在上述列表中的操作系统时，请谨慎操作并认真阅读 使用迁云工具迁移服务器至阿里云。

费用详情

阿里云迁云工具是免费工具，不收取额外的费用。但是，在迁云过程中会涉及少量费用：

迁云过程中，会创建快照以生成自定义镜像，该快照会按照实际占用容量收取少部分费用。详情请参阅 快照服务费用细则。

迁云过程中，系统默认在您的阿里云账号下创建一个 ECS 实例做中转站。该中转实例付费类型为按量付费，按量付费实例产生的资源耗费及计费说明请参阅 *产品定价* 按量付费。

使用限制

迁云之前，您需要确保待迁云的源服务器满足如下要求：

源服务器必须能够访问公网，便于传输数据至 ECS 控制台。

阿里云迁云工具当前版本暂不支持迁移增量数据。对于源服务器上需要保持数据完整的业务，您可以选择一个业务空闲时段，暂时停止这些业务，再迁移数据。

更多详情，请参阅 使用迁云工具迁移服务器至阿里云 的部分内容。

使用阿里云迁云工具

关于如何使用阿里云迁云工具，请参阅文档 使用迁云工具迁移服务器至阿里云。

参考链接

目前，阿里云支持的应用迁云的方式有：

- 使用 阿里云迁云工具 迁移入云。
- 通过 导入镜像 迁移入云。
- 使用 Packer 创建并导入本地镜像。
- 使用 Disk2VHD、DD、等镜像文件制作工具，转换镜像格式后导入镜像，参阅 工具介绍。

注意事项

使用阿里云迁云工具时，您需要注意：

阿里云迁云工具当前版本暂不支持迁移增量数据。对于源服务器上需要保持数据完整的业务，您可以选择一个业务空闲时段，暂时停止这些业务，再迁移数据。

迁云时，系统默认在您的阿里云账号下创建一个默认名为 INSTANCE_FOR_GOTOALIYUN 的 ECS 实例做中转站。该中转实例付费类型为按量付费，按量付费实例产生的资源耗费及计费说明请参阅 *产品定价* 按量付费。

注意：

- 为避免迁云失败，请勿停止、重启或者释放该中转实例。迁云完成后，该中转实例会自动释放。
- 迁云失败后，该实例保留在 ECS 控制台，以便于重新迁云。此时，您需要前往 ECS 控制台手动 释放实例，以免造成不必要的扣费。

中转实例的默认付费模式为 按量付费（Pay-As-You-Go），您需要确保账号余额大于等于 100 元。

每成功迁云一次，配置文件 client_data 会自动记录迁云成功后在 ECS 控制台创建的 ECS 实例的相关数据。再次迁云时，您需要使用初始下载的客户端配置文件。

注意：
为避免迁云失败，如无特殊需求，您无需自行修改配置文件 client_data。

阿里云迁云工具需要使用 AccessKeyID 以及 AccessKeySecret， AccessKeyID 以及 AccessKeySecret 是您的重要凭证，请妥善保管，防止泄露。

前提条件

使用阿里云迁云工具前，您需要注意：

- 迁云之前，待迁云的源服务器必须能够访问公网，便于传输数据至阿里云管理控制台。
- 确保系统本地时间与实际时间一致，否则日志文件会提示 TimeStamp 异常。

源服务器为 Windows 系统

- 确保防火墙没有限制 go2aliyun_client.exe 及 Rsync\bin 文件夹中的 rsync.exe。
- 确保系统开机启动引导正常。
- 您需要以管理员身份运行迁云工具。

源服务器为 Linux 系统

- 确保 go2aliyun_client 没有被防火墙限制。
- 确保您已经安装了 Rsync 库：
    - CentOS：运行 yum install rsync –y。
    - Ubuntu：运行 apt-get install rsync –y。
    - Debian：运行 apt-get install rsync –y。
    - 其他发行版：参考发行版官网安装相关的文档。
- 确保系统安装了 Xen 或者 KVM（Kernerl Virtual Machine）驱动，您可以参阅 安装 virtio 驱动 配置虚拟化环境。
- 确保系统已关闭 SELinux。您可以通过将文件 /etc/selinux/config 中的配置修改为 SELINUX=disabled 关闭 SELinux。
- 您需要以 root 身份运行迁云工具。
- 如果您的源服务器系统是 CentOS 5 或者 Debian 7 等内核版本比较低并且自带 GRUB 程序版本低于 1.99 时。您可以预先 安装 1.9 以上版本的系统引导程序 GRUB。

步骤一. 下载迁云工具

登录阿里云管理控制台 提交迁云申请。

注意：
提交迁云申请后，如果您长时间未收到迁云回复，您可以同时检查您的收件箱与垃圾邮件。

审核完成后，根据邮件、站内信或者短信提示下载阿里云迁云工具压缩包。解压后包含的文件列表如下：

Windows 服务器

文件（夹）名	描述
Rsync 文件夹	迁云所需要依赖的应用，除筛选机制文件 Rsync\etc\rsync_excludes_win.txt 外，其余文件请勿手动修改。
client_data	迁云过程中的数据文件
user_config.json	源服务器信息配置文件
go2aliyun_client.exe	迁云工具主程序

Linux 服务器

文件（夹）名	描述
client_check	辅助程序
client_data	迁云过程中的数据文件
user_config.json	源服务器信息配置文件
rsync_excludes_linux.txt	筛选机制文件，设置不迁云的路径。
go2aliyun_client	迁云工具主程序

步骤二. 使用迁云工具

登录待迁云的服务器、虚拟机或者云主机。

将下载的阿里云迁云工具压缩包解压到您指定的目录。

在控制台 创建 Access Key，用于输出到配置文件 user_config.json 里。

根据您的实际情况，自定义配置文件 user_config.json 和 自定义无需迁云的目录。

运行阿里云迁云工具：

Windows 服务器：右击 go2aliyun_client.exe，选择 **以管理员身份运行**。

Linux 服务器：

执行命令 chmod +x go2aliyun_client 赋予 go2aliyun_client 可执行权限。

执行命令 ./ go2aliyun_client 运行 go2aliyun_client。

等待运行结果：

当出现 Goto Aliyun Finished! 提示时，前往 **ECS 控制台** 镜像详情页查看结果。

当出现 Goto Aliyun Not Finished! 提示时，检查同一目录下 Logs 文件夹下的日志文件 **排查故障**。修复问题后，重新运行迁云工具，迁云工具会从上一次执行的进度中继续迁云。

自定义 user_config.json

user_config.json 是一个以 JSON 语言编写的配置文件，主要包含源服务器需要迁移至阿里云云平台的一些必要配置信息，其中包括您的 AccessKey 信息、生成的目标自定义镜像的配置信息等。您需要手动配置部分参数，修改后，仔细检查 JSON 语言格式的规范性，关于 JSON 的语法标准请参阅 RFC 7159。

配置文件模板

以下是配置文件 user_config.json 的模板：

```
{
"access_id": "",
"secret_key": "",
"region_id": "",
"image_name": "",
"system_disk_size": 40,
"platform": "",
"architecture": "",
"data_disks": []
}
```

模板参数说明

表一. 服务器配置参数说明

参数名	类型	是否必填	说明
access_id	String	Yes	您的阿里云账号的 API 访问密钥 AccessKeyID。更多详情，参阅 创建 Access Key。
secret_key	String	Yes	您的阿里云账号的 API 访问密钥 AccessKeySecret。更多详情，参阅 创建 Access Key。
region_id	String	Yes	您的服务器迁移入阿里云的地域 ID，如 cn-hangzhou（华东 1），取值参阅 地域与可用区。
image_name	String	Yes	为您的服务器镜像设定一个镜像名称，该名称不能与同一地域下现有镜像名重复。 - 长度为 [2, 128] 个英文或中文字符，必须以大小字母或中文开头，可包含数字，点号（.），下划线（_）或短横线（-）。 - 镜像名称会显示在 ECS 控制台。 - 不能以 http:// 和 https:// 开头。
system_disk_size	int	Yes	为系统盘指定大小，单位为 GB。取值范围： - [40, 500] - 该参数取值需要大于源服务器系统盘实际占用大小，例如，源系统盘大小为 500

			GB，实际占用 100 GB，那该参数取值只要大于 100 GB 即可。
platform	String	No	源服务器的操作系统。取值范围： - CentOS - Ubuntu - SUSE - OpenSUSE - Debian - RedHat - Others Linux - Windows Server 2003 - Windows Server 2008 - Windows Server 2012 - Windows Server 2016 参数 platform 的取值需要与以上列表保持一致，必须区分大小写，并保持空格一致。
architecture	String	No	系统架构。取值范围： - i386：32 位系统架构 - x86_64：64 位系统架构
data_disks	Array	No	数据盘列表，最多支持 16 块数据盘。具体参数参阅 数据盘配置参数说明。

表二. 数据盘配置参数说明

参数名	类型	是否需要手动配置	说明
data_disk_index	int	Yes	数据盘序号。取值范围：[1, 16] 初始值：1

data_disk_size	int	Yes	数据盘大小。单位为 GB。取值范围： - [20, 32768] - 该参数取值需要大于源服务器数据盘实际占用大小，例如，源数据盘大小为 500 GB，实际占用 100 GB，那该参数取值只要大于 100 GB 即可。
src_path	String	Yes	数据盘源目录。取值范围： - Windows 指定盘符，例如，D、E 或者 F。 - Linux 指定目录，例如，/mnt/disk1、/mnt/disk2 或者 /mnt/disk3。

自定义配置示例

此处以四种场景为例，为您示范如何根据场景和配置文件模板自定义配置 user_config.json 文件：

场景一. 迁移一台无数据盘的 Windows 服务器

- 假设您的服务器配置信息为：
  - 操作系统：Windows Server 2008
  - 系统盘：30 GB
  - 系统架构：64 位
- 您的迁云目标为：
  - 目标地域：阿里云华东 1 地域（cn-hangzhou）
  - 镜像名称：CLIENT_IMAGE_WIN08_01
  - 系统盘设置：50 GB

那么您可以根据如下信息配置 user_config.json 文件：

```
{
"access_id": "YourAccessKeyID",
"secret_key": "YourAccessKeySecret",
"region_id": "cn-hangzhou",
"image_name": "CLIENT_IMAGE_WIN08_01",
"system_disk_size": 50,
"platform": "Windows Server 2008",
"architecture": "x86_64",
"data_disks": []
}
```

场景二. 迁移一台带数据盘的 Windows 服务器

如果您的 Windows 服务器在场景一的基础上加入了 3 块数据盘，源目录和数据盘大小分别为：

- D：100 GB
- E：150 GB
- F：200 GB

那么您可以根据如下信息配置 user_config.json 文件：

```
{
"access_id": "YourAccessKeyID",
"secret_key": "YourAccessKeySecret",
"region_id": "cn-hangzhou",
"image_name": "CLIENT_IMAGE_WIN08_01",
"system_disk_size": 50,
"platform": "Windows Server 2008",
"architecture": "x86_64",
"data_disks": [ {
"data_disk_index": 1,
"data_disk_size": 100,
"src_path": "D:"
}, {
"data_disk_index": 2,
"data_disk_size": 150,
"src_path": "E:"
}, {
"data_disk_index": 3,
"data_disk_size": 200,
"src_path": "F:"
}
]
}
```

场景三. 迁移一台无数据盘的 Linux 服务器

- 假设您的服务器配置信息为：

- 发行版本：CentOS 7.2
- 系统盘：30 GB
- 系统架构：64 位

- 您的迁云目标为：
  - 目标地域：阿里云华东 1 地域（cn-hangzhou）
  - 镜像名称：CLIENT_IMAGE_CENTOS72_01
  - 系统盘设置：50 GB

那么您可以根据如下信息配置 user_config.json 文件：

```
{
"access_id": "YourAccessKeyID",
"secret_key": "YourAccessKeySecret",
"region_id": "cn-hangzhou",
"image_name": "CLIENT_IMAGE_CENTOS72_01",
"system_disk_size": 50,
"platform": "CentOS",
"architecture": "x86_64",
"data_disks": []
}
```

场景四. 迁移一台有数据盘的 Linux 服务器

如果您的 Linux 服务器在场景三的基础上加入了 3 块数据盘，源目录和数据盘大小分别为：

- /mnt/disk1：100 GB
- /mnt/disk2：150 GB
- /mnt/disk3：200 GB

那么您可以根据如下信息配置 user_config.json 文件：

```
{
"access_id": "YourAccessKeyID",
"secret_key": "YourAccessKeySecret",
"region_id": "cn-hangzhou",
"image_name": "CLIENT_IMAGE_CENTOS72_01",
"system_disk_size": 50,
"platform": "CentOS",
"architecture": "x86_64",
"data_disks": [ {
"data_disk_index": 1,
"data_disk_size": 100,
"src_path": "/mnt/disk1"
}, {
"data_disk_index": 2,
"data_disk_size": 150,
"src_path": "/mnt/disk2"
}, {
"data_disk_index": 3,
"data_disk_size": 200,
```

```
"src_path": "/mnt/disk3"
}
]
}
```

自定义无需迁云的目录

阿里云迁云工具同时具备过滤筛选功能，该功能过滤掉部分文件或者目录，这些被过滤的文件不会迁移到阿里云云端。其默认过滤文件为：

- Linux 服务器：
  - /dev/*
  - /sys/*
  - /proc/*
  - /media/*
  - /lost+found/*
  - /mnt/*
  - /var/lib/lxcfs/*
  - ...

    注意：
    /var/lib/lxcfs/* 目录仅针对部分系统版本，例如，无权访问 Ubuntu 的 Linux 容器服务缓存目录时，需要排除 Ubuntu 的 /var/lib/lxcfs/* 才能顺利迁云。

- Windows 服务器
  - pagefile.sys
  - $RECYCLE.BIN
  - System Volume Information

您可以通过配置 rsync 实现过滤不想迁云的文件或者目录。Linux 服务器通过配置文本文件 **rsync_excludes_linux.txt** 实现，Windows 文件通过配置文本文件 **Rsync/etc/rsync_excludes_win.txt** 实现。您可以参阅 rsync 相关文档，在文本文件里按需添加过滤的文件或目录。

排查故障

日志文件

阿里云迁云工具的日志记录保存在主程序目录下的 Logs 目录。日志文件记录迁云过程中出现异常中断，如提示 Go2aliyun Not Finished 时，您可查看日志详情 定位并解决问题。

FAQ

1. 迁云日志关键字里包含了 TimeStamp

请检查系统时间是否为正确时间。

2. 迁云日志关键字里包含了 OperationDenied

请确保配置文件 user_config.json 中参数 access_id 所属的阿里云账号已 申请 开通迁云功能白名单。

3. Linux 服务器日志关键字里包含了 check rsync failed

请检查系统是否已安装 rsync 组件。

4. Linux 服务器迁云日志关键字里包含了 check virtio failed

请检查系统是否安装 virtio 驱动。

5. Linux 服务器迁云日志关键字里包含了 check selinux failed

请检查是否已禁用 SElinux。

您可以通过将文件 /etc/selinux/config 中的配置修改为 SELINUX=disabled 关闭 SELinux。

6. Linux 服务器迁云日志错误提示关键字里包含了 Do Grub Failed

日志文件提示如 Do Grub Failed 的错误信息时，确保源服务器已经安装了系统引导程序 GRUB（GRand Unified Bootloader）。您可以 安装 1.9 以上版本的系统引导程序 GRUB 后重试。

7. 日志错误提示关键字里包含了 Unknow Error

请检查配置文件 user_config.json 中参数 platform 取值是否正确。

8. 日志错误提示关键字里包含了 Permission denied

日志文件提示如 rsync: send_files failed to open "...": Permission denied (13) 的错误信息时，表明阿里云迁云工具无权访问该目录或文件夹，导致 rsync 失败。此时您可以通过配置 rsync_excludes_linux.txt 或者 Rsync/etc/rsync_excludes_win.txt 过滤该目录或文件夹，然后重试。

9. 日志错误提示关键字里包含了 NotEnoughBalance

中转实例的默认付费模式为 按量付费（Pay-As-You-Go），您的账号余额不足时，无法顺利迁云。您需要更新账户状态后重试。

10. 为什么启动 Others Linux 实例后，网络服务不正常？

导入 Others Linux 类型镜像时，阿里云不会对该自定义镜像所创建的 ECS 实例做任何配置工作，包括相关的网络配置、SSH 配置等。此时，您需要参阅 配置 Customized Linux 自定义镜像 自行修改。

如果网络配置失败，您可以 提交工单 联系阿里云。

11. 迁移 Linux 服务器后，根据该自定义镜像创建的 ECS 实例为何不能启动？

检查驱动。创建 I/O 优化的 ECS 实例时，请确保源服务器已经安装 virtio 驱动。

检查一下源系统引导配置是否正确。

如果您的源服务器系统是 CentOS 5 或者 Debian 7 等内核版本比较低并且自带 GRUB 程序版本低于 1.99 时。同时在 ECS 控制台 远程连接 登录实例发现开机界面如下图所示。

您可以 安装 1.9 以上版本的系统引导程序 GRUB 后重试。

测试镜像

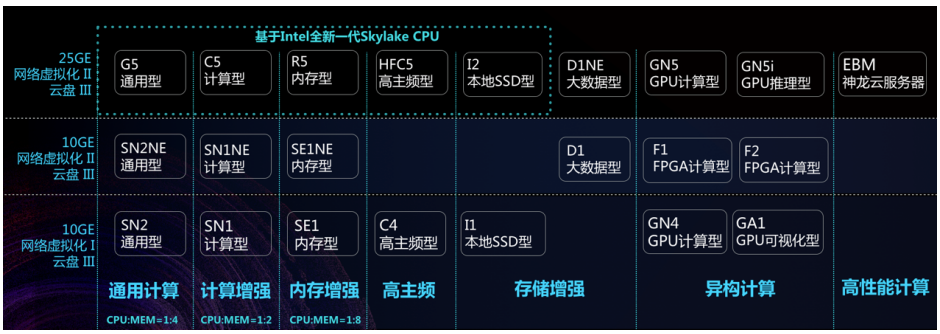
迁云完成后，您的源服务器中的操作系统、应用程序以及应用数据等以自定义镜像的形式出现在相应地域的 ECS 控制台上。您可以 使用该自定义镜像创建按量付费 ECS 实例，测试自定义镜像能否正常运行。

后续操作

- 您可以 使用自定义镜像创建 ECS 实例。
- 您可以 使用自定义镜像更换系统盘。

ECS性能测试

Elasticsearch 是一个分布式、可扩展、实时的搜索与数据分析引擎。下图显示阿里云的企业级家族（独享实例的SLA性能是有保证的）。



Elasticsearch 对 CPU 的要求不高，但要求比较大的内存（无需超过 64 GB）和 I/O 吞吐量，对网络要求高。推荐实例规格族：ecs.sn2ne.4xlarge 和 ecs.i2.4xlarge。

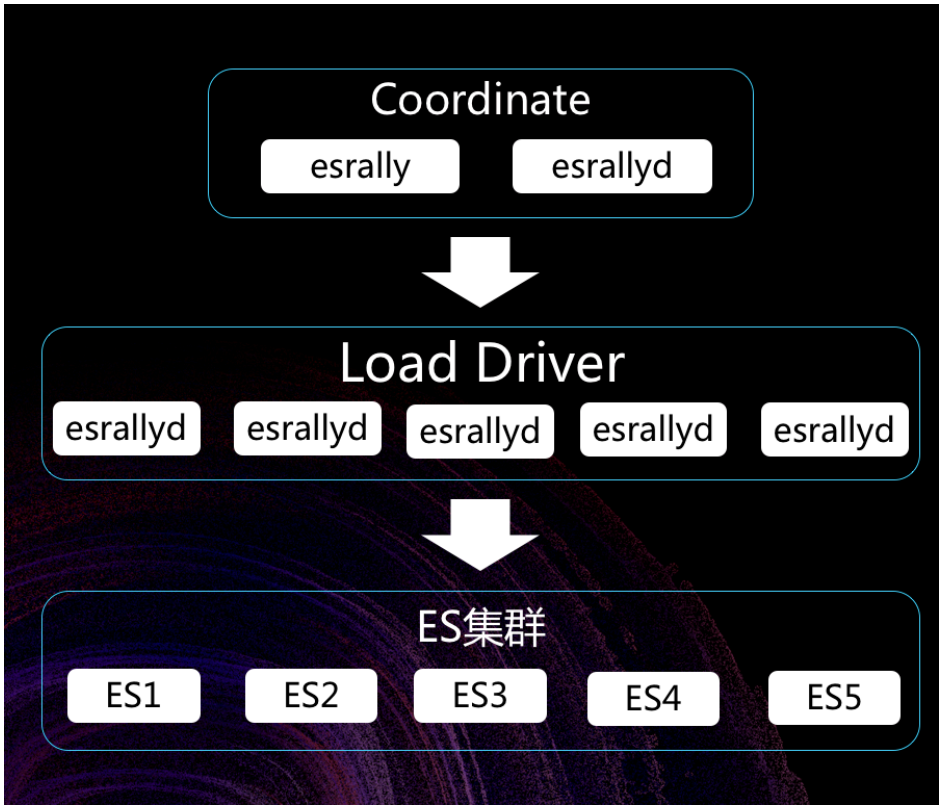
测试验证

测试方法

测试版本：Elasticsearch 5.5

压测软件：esrally

测试架构：



压测时，请执行如下命令：

```
esrally --offline --load-driver-hosts=172.31.189.171,172.31.189.169,172.31.189.170,172.31.189.167,172.31.189.168 --track=geonames --pipeline=benchmark-only --target-hosts=172.31.189.182:9200,172.31.189.181:9200,172.31.189.180:9200,172.31.189.179:9200,172.31.189.178:9200 --client-options="basic_auth_user:'elastic',basic_auth_password:'changeme'" --user-tag="version:i2_test_1100"
```

压测机型如下：

- ecs.sn2ne.4xlarge，4 块 1 TB SSD 云盘，并做了 RAID 0。
- ecs.i2.4xlarge 2 块 1.7 TB SSD 本盘，并做了 RAID 0。

参数调整

系统参数调整。

i. 打开多队列。
ii. 文件打开数增大。
iii. 禁用swap：vm.swappiness = 1。

Elasticsearch参数调整。

这次压测基本没有调整参数，简单调整了 2 个参数，配置项如下：

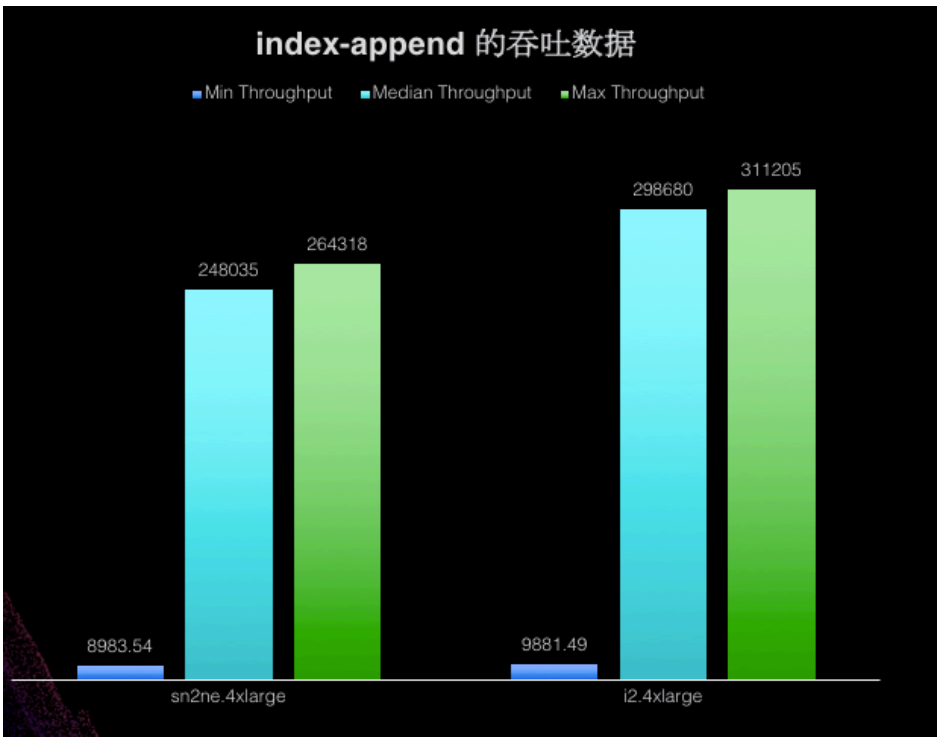
- bootstrap.memory_lock: true

  indices.query.bool.max_clause_count: 4096

  不推荐随便修改参数，认为如果性能不达标，更多的需要关注运营，在官网的中文介绍《Elasticsearch: 权威指南》里面有很多建议，如：

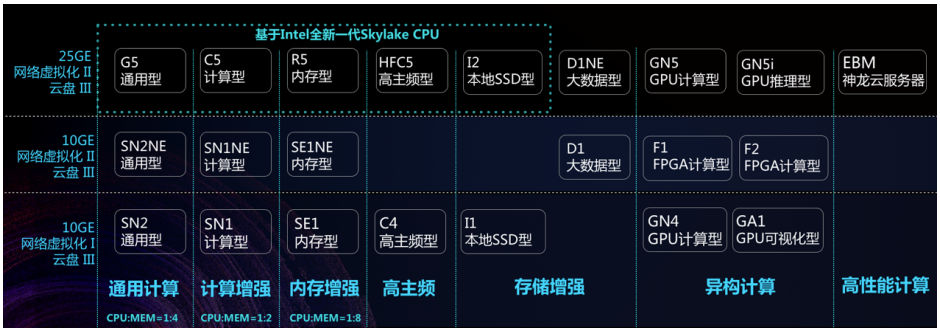
  - 使用别名而不是索引名，这样可以在任何时候重建索引。
  - 深度分页不可取，游标查询。
  - 利用好索引模板。

测试结论



从上图可以看出，用多个 SSD 云盘效果不错，而用 SSD 本地盘效果更佳！

阿里云的企业级家族（独享实例的SLA性能是有保证的）如下图所示。



MySQL 对 I/O 的低延迟非常敏感，同时对网络 PPS 要求也很高。基于 MySQL 这 2 个特性，推荐使用网络增强 + SSD 云盘的规格族 ecs.sn2ne以及本地 SSD 型的规格族 I1 和 I2。

官网介绍中，SSD 云盘的单盘最大 IOPS 是 20000，延时 ms 级，而 SSD 本地盘的单盘最大 IOPS 是 240000，延时 us 级。不同盘使得MySQL性能差别很大，下面的测试可以对此进行验证。

测试验证

测试方法

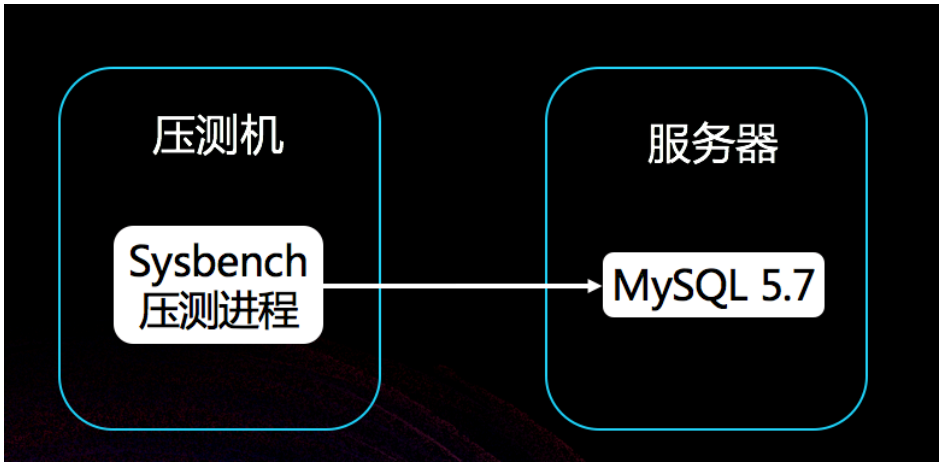
测试软件: percona-5.7.19-17

测试对象：

- ecs.sn2ne.8xlarge 32C/128G+1TB SSD云盘
- ecs.i1-c10d1.8xlarge 32C/128G+1456G SSD本地盘

测试工具：Sysbench 1.0.9

测试架构：



压测命令：

```
/usr/local/sysbench/bin/sysbench /usr/local/sysbench/share/sysbench/oltp_read_write.lua \
--mysql-host=目标IP \
--mysql-port=3306 \
--mysql-user=root \
--mysql-password='mysql密码' \
--mysql-db=dbtest1a \
--db-driver=mysql \
--tables=10 \
--table-size=10000000 \
--report-interval=10 \
--threads=64 \
--time=120 \
prepare/run/cleanup
```

测试步骤

i. 不做任何优化的压测。
ii. 做优化后的压测。

参数调优

系统参数调优。

i. 开启多队列。
ii. 文件打开数增大。

MySQL参数调优。

innodb_buffer_pool_size: 缓存 innodb 表的索引，数据，插入数据时的缓冲。MySQL 默认的值是 128M。官方推荐使用物理内存的 70% - 80%。现在设置是：100GB。

innodb_log_file_size: 表示在一个日志组每个日志文件的字节大小，默认 48MB，对于写很多尤其是大数据量时非常重要。要注意，大的文件提供更高的性能，但数据库恢复时会用更多的时间。一般用 64M-512M，具体取决于服务器的空间。

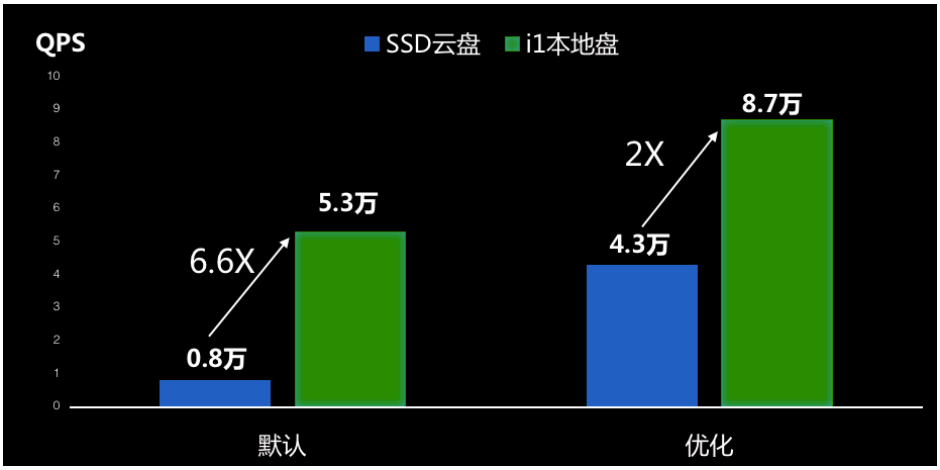
该参数决定了 recovery speed。太大的话 recovery 就会比较慢，太小了影响查询性能，一般取 256M 可以兼顾性能和 recovery 的速度。现在设置是512M。

innodb_flush_log_at_trx_commit：参数指定了 InnoDB 在事务提交后的日志写入频率。当取值为 1 时，每次事务提交时，log buffer 会被写入到日志文件并刷写到磁盘，这也是默认值，这是最安全的配置，但由于每次事务都需要进行磁盘I/O，所以也最慢。当取值为 2 时，每次事务提交会写入日志文件，但并不会立即刷写到磁盘，日志文件会每秒刷写一次到磁盘。取值为 0 的时候，log buffer 会 每秒写入到日志文件并刷写（flush）到磁盘。

对于一些数据一致性和完整性要求不高的应用，配置为 2 就足够了；如果为了最高性能，可以设置为 0。有些应用，如支付服务，对一致性和完整性要求很高，所以即使最慢，也最好设置为 1。现在设置是2。

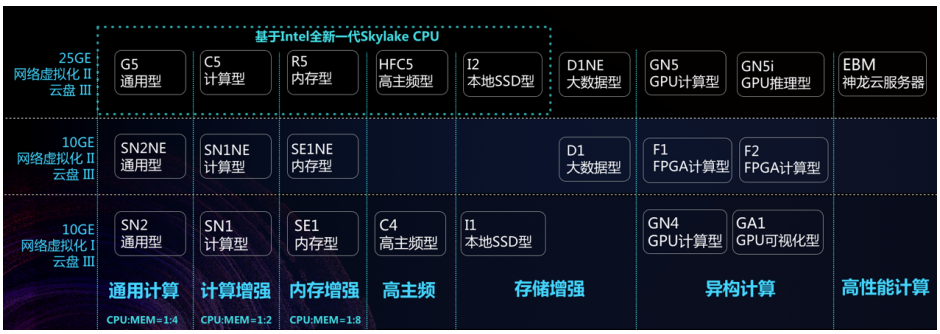
innodb_flush_method：推荐设置 O_DIRECT。

测试结论



规格族 I1 的 SSD 本地盘性能比 SSD 云盘性能好很多。推荐规格族 I1 或者 I2。

阿里云的企业级家族（独享实例的SLA性能是有保证的）如下图所示。



Nginx可以作为HTTP服务器和反向代理服务器。反向代理服务器取决于后端服务器的性能，这次只针对HTTP服务器做性能测试。Nginx作为服务器对于网络的性能必然是非常依赖的，尤其是PPS转发能力，那么网络增强型实例必然是首选。

在 10G 网络带宽下，推荐独享实例规格族如下：规格族 ecs.sn1ne（Nginx 对内存要求不高，不需要规格族 ecs.sn2ne）； 在 25G 网络带宽下，推荐实例规格族： 规格族 C5。

测试验证

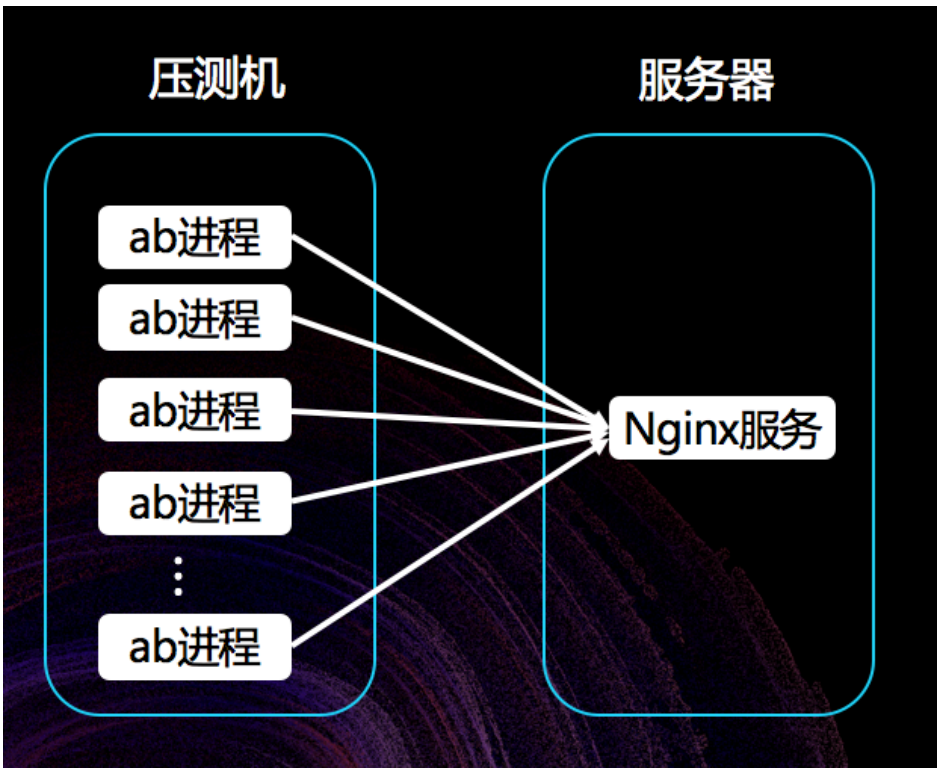
测试方法

- 操作系统：Centos 7.3 （默认打开irqbalance）
- 测试软件: Nginx 1.12.1
- 压测工具：ApacheBench 2.3

测试对象

- ecs.sn1ne.4xlarge 16C/32GB
- ecs.sn1ne.8xlarge 32C/64GB

测试架构



压测命令

32个并发命令： ab -n 100000000 -c 10 -k http://${server_ip}/

参数调整

系统参数调整

i. 打开多队列。

开启 RPS。

经过测试发现，16 核的时候，不需要开启RPS特性，就可以把所有 CPU 打满，网络达到极限；但是测试 32 核的时候，需要开启 RPS。

修改文件打开数。

Nginx 参数调整。

打开多进程。Nginx默认是单work进程。

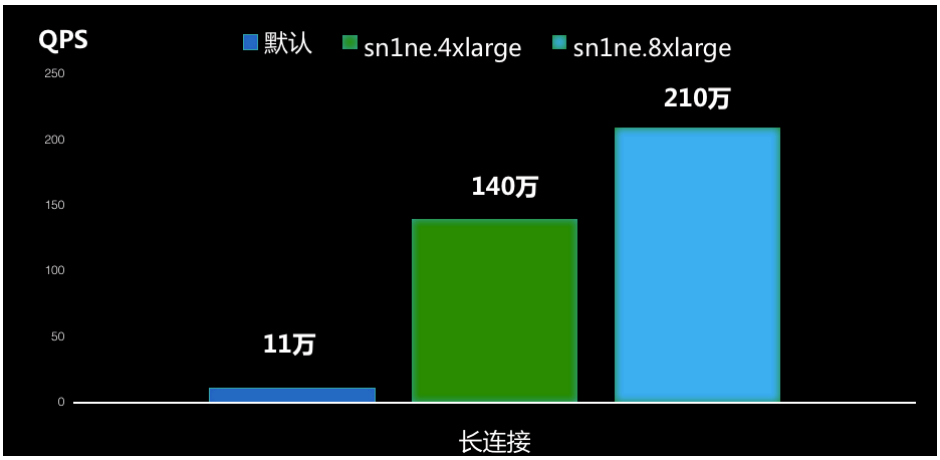
在 nginx.conf 文件中可以配置如下：

i. worker_processes 32;

ii. worker_cpu_affinity auto;

增大连接数：配置 worker_connections 102400。

测试结论

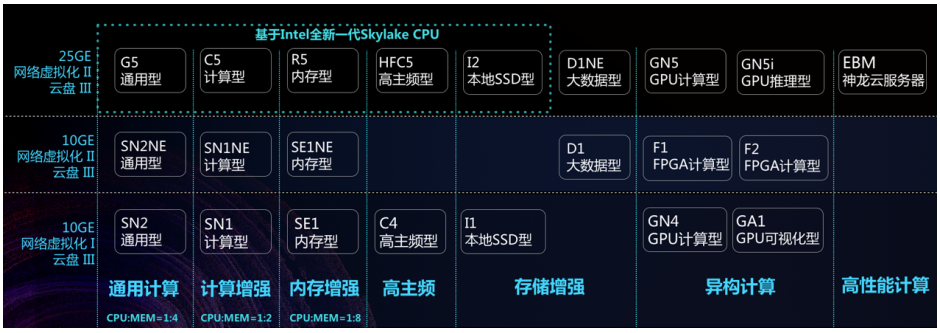


sn1ne.4xlarge 的 pps 最高是 150w，此次压测 QPS 达到了 140w。此时所有的 CPU 利用率都接近 100%。（此处的QPS是通过tsar统计的。）

sn2ne.8xlarge 的 PPS 最高是 250w，此次压测 QPS 达到了 210w。此时所有的 CPU 利用率都接近 100% 了。

Redis 是高性能的 key-value 数据库，被广泛使用。但 Redis 作为一个单进程应用，它需要被发挥作用，就需要集群部署，否则可用性将无法保障。本次采用的方案是 Redis-cluster 方案，这也是官方推荐的方案。

在给 Redis 集群选择机型之前，先看下阿里云的企业级VM大图：



这里推荐规格族 SE1NE 或者 R5，强烈推荐 R5，Redis 对 CPU 的利用比较高。

测试验证

- 机型：ecs.se1ne.4xlarge 16C128G，单机 PPS 能力是：160w。

- Redids 版本是：Redis-4.0.2
- 压测软件：memtier_benchmark-1.2.10
- 操作系统：Centos 7.3

测试方法

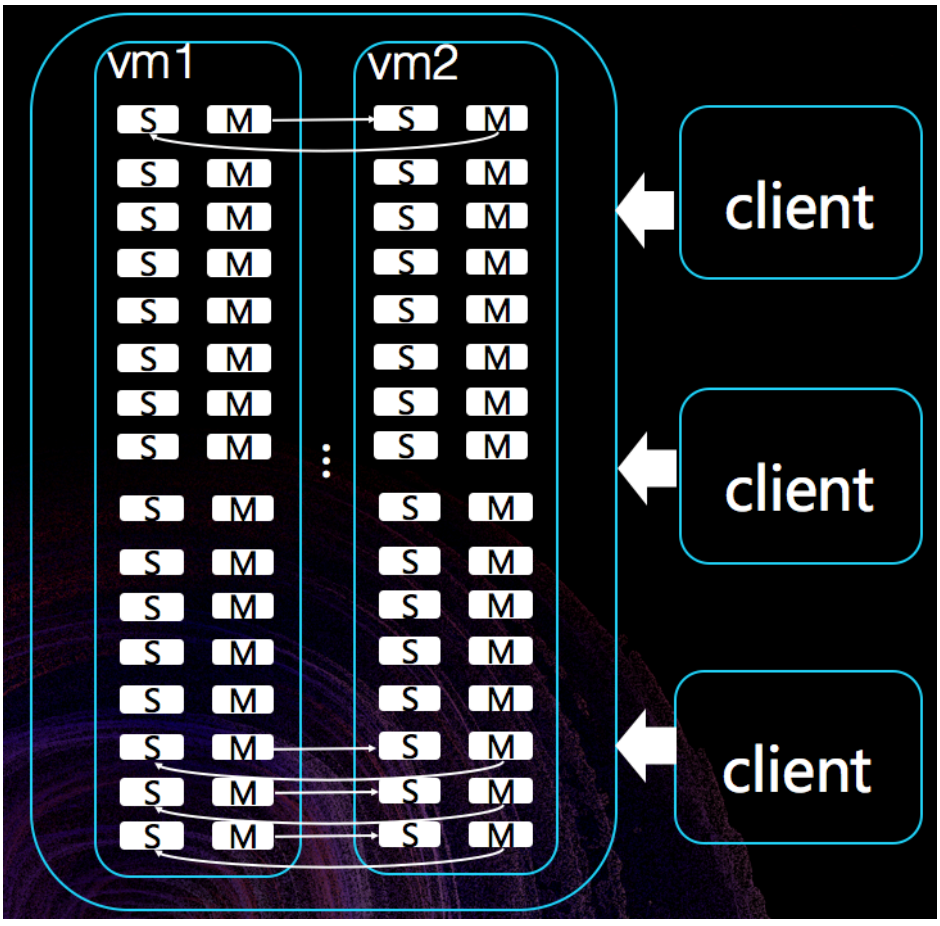
官方在 Redis cluster 方法上介绍的是单机版本，如果这台机器宕机，Redis 的整个服务将不可用。而测试采用的是 2 台实例互备。压测机器是 3 台 ecs.sn1ne.4xlarge 16C32G，3 台 client 递增压测。

由于 memtier_benchmark 不支持对集群压测，需要通过 hash tag 指定 key-prefix，以达到压测指定 Redis 进程。

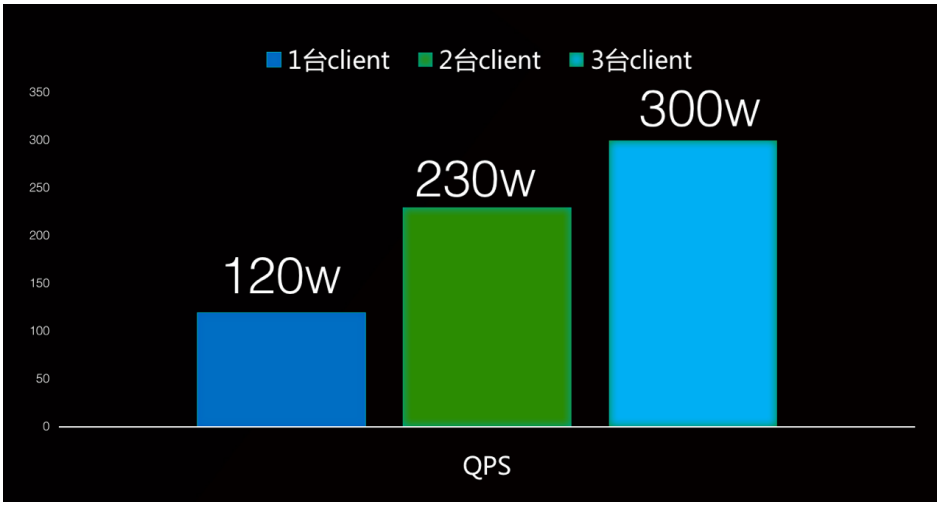
压测命令

```
memtier_benchmark -s ${ip} -p ${port} -t 2 -n 100000000 --key-prefix={prefix} --out-file=/tmp/${port}.out > /tmp/${port}.log 2>&1 &
```

压测拓扑图



压测结果



ecs.se1ne.4xlarge 单机最高 PPS 是160w，2 台实例加起来最高 PPS 是 320w，当 3 台 client 压测的时候，压测机的 CPU 使用率都已经接近 100%，已经完全把实例的性能发挥出来了。