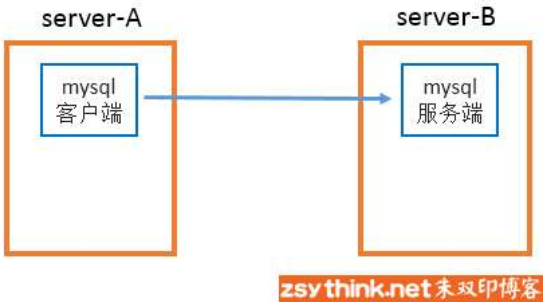
A+

ssh端口转发：ssh隧道

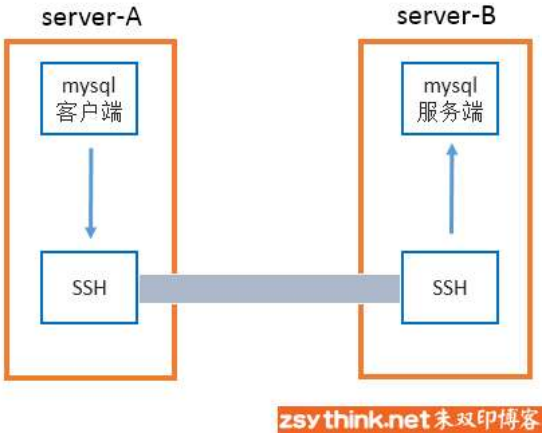
在之前的文章中，我们总结了"ssh代理转发"的相关知识点，"代理转发"是针对ssh认证过程的一种转发，而在这片文章中，我们将会总结ssh中的另外一种转发："端

"ssh端口转发"还有一个更加形象的名字，叫做"ssh隧道"，当然，只是纯粹的通过"ssh隧道"这几个字去理解它可能不太容易，我们来描述一些实际的场景，在这些场
会遇到一些问题，而这些问题可以通过"ssh隧道"解决，通过这样的方式，我们反而更加容易理解"ssh隧道"是什么以及它的作用。

假如我们现在有两个台主机，主机A与主机B，主机A上安装有mysql客户端，主机B上安装有mysql服务端，现在，主机A中的mysql客户端需要与主机B中的mysql服
讯，则需要从mysql的客户端连接到mysql服务端。如下图所示



然而我们知道，mysql在传输数据时是进行明文传输的，如果主机A与主机B只能通过公网进行通讯，那么暴露在公网的mysql通讯是非常不安全的，所以，我们需要
段，提高访问mysql服务时的安全性，比如，我们可以使用SSL证书为数据加密，或者使用stunnel加密隧道，我们还可以使用VPN，当然，这些方法都不是这篇文章
点，我们此处要总结的是"ssh隧道"这种方法，我们可以利用ssh，搭建出一条"通道"，然后将mysq的客户端与服务端通过这条"ssh通道"连接起来，如下图所示



mysql的客户端与服务端的连接方式从原来直连的方式变成了如上图所示的连接方式，它们之间并不直接进行通讯，而是借助ssh隧道将通讯数据转发，虽然仍然跨越
由于ssh本身的安全特性，所以别人无法看到明文传输的数据，数据依靠ssh隧道实现了加密的效果，达到了保护数据安全的作用，提升了mysql的客户端与服务端通

本地转发

经过上述描述，我想你对"ssh隧道"应该已经有了初步的理解，那么现在我们来实际动手配置一下。

首选，将实验环境准备好，两台主机的信息如下

ServerA：10.1.0.1

ServerB：10.1.0.2

ServerA中并不存在mysql服务。

ServerB中已经安装了mysql服务，mysql服务已经启动并监听了3306端口。

现在，我们只要在ServerA中执行如下命令，即可在ServerA与ServerB之间建立一条ssh隧道，执行如下命令时会提示输入ServerB的密码

如上图所示，执行上图中的命令后，我们直接从主机A连接到了主机B，这条连接就是我们创建的"ssh隧道"。

我们先来简单的解释一下上图中命令的含义，为了方便解释，我们把命令分成3部分理解，如下图所示。

第1部分为-L选项，-L 选项表示使用"本地转发"建立ssh隧道，本地转发是什么意思呢？

"本地转发"表示本地的某个端口上的通讯数据会被转发到目标主机的对应端口，你可以把它抽象的理解成一种"映射"，注意，我们把执行上述命令的主机称为"本地主

比如，访问本地(当前主机)的端口A，就相当于访问目标主机的端口B，因为当你访问本地的端口A时，通讯数据会被转发到目标主机的端口B，这就是本地转发，其实
发"是与"远程转发"相对应的，但是我们还没有介绍到远程转发，所以并不用在意那么多，我们只要先了解本地转发的作用就行了。

刚才说过，"本地转发"表示本地的某个端口上的通讯数据会被转发到目标主机的对应端口，那么你一定能够理解上述命令中第2部分的含义了

第2部分表示：通讯数据会从本地的9906端口上被转发，最终被转发到10.1.0.2的3306端口。

第3部分表示：我们创建的ssh隧道是连接到10.1.0.2上的root用户的，其实，第3部分可以与之前的ssh连在一起去理解，比如，ssh root@10.1.0.2，其实就是使用
verA中连接到ServerB的root用户，这就是为什么执行上述命令以后，会提示我们输入10.1.0.2中root用户的密码，当然，如果你已经在ServerB中配置好了Server
公钥，那么则可以省去输入密码的步骤直接连接，此时，ServerA的角色是ssh的客户端，ServerB的角色是ssh的服务端，而这条ssh隧道就是建立在ServerA与Ser

了解完上述命令的3个部分，我们来把它当做一个整体去理解一下

ssh -L 9906:10.1.0.2:3306 root@10.1.0.2

上述命令表示从本机(ServerA)建立一个到ServerB(10.1.0.2)的ssh隧道，使用本地端口转发模式，监听ServerA本地的9906端口，访问本机的9906端口时，通讯数
到ServerB(10.1.0.2)的3306端口。

好了，命令解释完了，现在我们来试试实际的使用效果，注意，此刻我们已经创建了ssh隧道，从serverA中已经连接到了ServerB，不要退出这个ssh连接，否则刚
道将会消失（稍后会介绍怎样后台建立连接），此刻，我们再打开一个新的ssh连接，连接到ServerA，如下图所示

在新链接中查看对应的端口号，本地回环地址的9906端口已经被监听了（稍后介绍怎样监听ServerA中指定的IP，即非本地回环地址）。

此时，我们直接在ServerA中通过mysql命令访问127.0.0.1的9906端口，就相当于访问ServerB的mysql服务了，我们来试试。

执行mysql命令时需要指定IP与端口号，因为我的ServerB中的mysql只是用于测试，所以没有为用户设置密码，如下图即可连接

如上图所示，已经可以正常在ServerA中连接到数据库，但是连接的数据库其实是ServerB中的mysql服务。

这就是通过ssh隧道访问远程主机的mysql服务的示例，这样做就是利用ssh的安全特性加密了mysql的通讯数据。

在没有使用ssh隧道时，直接从ServerA跨越公网访问ServerB的mysql服务时，如果在ServerB中通过抓包工具对通讯网卡进行抓包，可以直接从抓到的数据包中看
输数据。

但是如果使用了ssh隧道，并且在ServerB中仅对通讯网卡进行抓包时，则只能看到经过加密的ssh数据包，此时，如果对ServerB的本地回环网卡同时进行抓包，则
密的mysql传输数据，不过，这并不影响mysql通讯数据跨越公网时的安全性，因为这时已经是ServerB本机中的数据传输了，也就是说，mysql通讯数据在跨越公网
h隧道加密的，mysql通讯数据到达ServerB本机以后，是明文传输的。

不过，当我们执行上述命令创建ssh隧道时，总会从ServerA中连接到ServerB中，而通常，我们只希望建立ssh隧道，并不会使用到这个新建立的ssh连接，而且在实
们往往会在建立隧道以后，退出当前的ssh会话，所以，上述命令并不能满足我们的需求，因为，我们一旦退出对应的ssh会话，相应的ssh隧道也会消失，所以，我
另外两个选项，"-N选项"与"-f选项"，我们——道来。

首先来试试"-N选项"，当配合此选项创建ssh隧道时，并不会打开远程shell连接到目标主机，我们来试试，如下图所示，配合-N选项创建隧道，输入ServerB的密码
连接到ServerB，而是停留在了如下图的位置

此时，再打开一个新的ssh会话连接到ServerA，可以看到，9906端口已经被监听。

但是，这样仍然不能满足我们的要求，虽然建立隧道时并没有连接到ServerB，但是，我们仍然不能关闭创建ssh隧道时所使用的ssh会话。

这时，只要配合"-f"选项即可，"-f"选项表示后台运行ssh隧道，即使我们关闭了创建隧道时所使用的ssh会话，对应的ssh隧道也不会消失，"-f"选项需要跟"-N"选项
以通常，我们会使用如下命令创建ssh隧道

ssh -f -N -L 9906:10.1.0.2:3306 root@10.1.0.2

配合上述选项创建ssh隧道时，即使我们完全关闭了执行命令时的ssh会话，对应创建的隧道也可以完全正常运行。

不过，当我们使用上述命令建立隧道时，只有127.0.0.1这个回环地址的9906端口会被监听，这样就会出现一个小问题，也就是说，我们只能在ServerA本机上访问
不能通过其他主机访问ServerA的9906端口，因为ServerA其他IP的9906端口并未被监听，那么怎么办呢？很简单，使用如下命令，即可让9906端口监听在Server
上

ssh -f -N -L 10.1.0.1:9906:10.1.0.2:3306 root@10.1.0.2

在ServerA中执行上述命令时，ServerA的10.1.0.1的9906端口会被监听，此刻，我们可以通过其他主机访问10.1.0.1的9906端口，即可访问到ServerB中的mysql
与之前的命令相比，只是在9906前增加了ServerA中对应的IP地址罢了，很简单吧。

如果你觉得这还不够，希望ServerA中的所有IP地址的9906端口都被监听，那么可以在建立隧道时开启"网关功能"，使用"-g"选项可以开启"网关功能"，开启网关功
erA中的所有IP都会监听对应端口，示例如下

好了，说了这么多，终于把ssh隧道(本地转发)给解释明白了，不过，我们也只是说明了本地转发，现在，我们来聊聊远程转发。

远程转发

在了解远程转发之前，请先确定你已经理解了"本地转发"。

老规矩，为了方便理解，我们先来描述一个场景。

公司有一台服务器ServerB，ServerB处于公司的内网中，公司内网中的所有主机都通过路由器访问互联网（典型的NAT网络），ServerB中有提供mysql服务，如果
要通过外网访问到ServerB中的mysql服务，该怎么办呢？通常的做法是，通过路由器或者防火墙，将公司的固定外网IP上的某个端口映射到ServerB内网IP的3306端

样，我们只要访问公司外网IP的对应端口，即可访问到内网ServerB中的mysql服务了，但是，如果你没有权限控制公司的防火墙或者路由器呢，这时该怎么办呢？

假设，你无法控制防火墙去进行端口映射，但是，公司在公网上有另外一台服务器ServerA，ServerA有自己的公网IP，你有权控制ServerA，这时，我们就可以利用
我们的目的，聪明如你，一定想到了解决方案，没错，我们可以在ServerA与ServerB之间创建一条SSH隧道，利用这条隧道将ServerA中的某个端口(假设仍然使用
erverB中的3306端口连接起来，这样，当我们访问ServerA的9906端口时，就相当于访问到内网ServerB中的mysql服务了，那么，我们能不能使用之前的"本地转
ServerA中创建SSH隧道呢？我们来模拟一下，看看会不会遇到什么问题，如果想要使用之前的命令创建SSH隧道，那么我们则需要在ServerA中执行如下命令。

```
ssh -f -N -L AIP:9906:BIP:3306 root@BIP
```

问题来了，ServerA有自己的公网IP，我们只要把上述命令中的AIP替换成ServerA的公网IP即可，但是ServerB是内网主机，虽然ServerB能够通过公司内的路由器
网，但是ServerB并不持有任何公网IP，ServerB只有内网IP，所以，我们并不可能把上述命令中的BIP替换成B主机的内网IP，所以，使用上述命令是无法在ServerA
道连接到ServerB的，那么该怎么办呢？

虽然我们无法从ServerA中使用ssh命令连接到ServerB，但是，我们可以从ServerB中使用ssh命令连接到ServerA啊，虽然ServerB是没有公网IP的内网主机，但是
靠公司的路由器访问互联网，所以，我们只要在ServerB中执行如下命令，即可从ServerB中连接到ServerA中。

```
ssh root@AIP
```

那么，按照这个思路，我们似乎找到了方向，我们现在需要一种方法，能够从ServerB中创建SSH隧道连接到ServerA，并且，隧道创建后，ServerA中会监听9906
人能够通过外网访问，也就是说，我们需要一种方法，能够满足如下两个条件

条件1：从ServerB中主动连接到ServerA，即在ServerB中执行创建隧道的命令，连接到ServerA。

条件2：隧道创建后，转发端口需要监听在ServerA中，以便利用ServerA访问到内网的ServerB。

这种方法就是"远程转发"。

你可能还是不太明白，没有关系，我们先来实际动手操作一下，稍后，我们会对比本地转发与远程转发的具体区别。

为了方便，我们仍然使用之前的实验环境，假设ServerA是外网主机，ServerB是内网主机，ServerA的IP为10.1.0.1（假设此IP为公网IP），ServerB的IP为10.1.0.
之前本地转发的进程关闭，相当于一个没有任何隧道的新的实验环境。

使用"-R选项"，可以创建一个"远程转发"模式的ssh隧道，我们在ServerB中，执行如下命令即可

上述命令在ServerB中执行，执行后，即可在ServerA与ServerB之间建立ssh隧道，此时，ServerB是ssh客户端，ServerA是ssh服务端，隧道建立后，ServerA中的
被监听，在ServerA中查看对应端口，如下图所示

从图中可以看出，ServerA中的9906端口已经被监听，此刻，我们通过外网IP登录到ServerA，在ServerA中访问本地回环地址的9906端口，即可访问到内网Serve
服务，如下图所示。

不过你肯定注意到了，当使用远程转发的命令时，我并没有指定监听ServerA的外网IP，也没有使用"-g选项"开启网关功能，这是因为，即使你在命令中指定了IP地
verA中还是会只监听127.0.0.1的9906端口，你可以在ServerB中尝试一下如下命令

```
ssh -f -N -R 10.1.0.1:9906:10.1.0.2:3306 root@10.1.0.1
```

即使在ServerB中执行上述命令时指定了IP或者开启了网关功能，ServerA的9906端口仍然只监听在127.0.0.1上，当然，如果你一心想要通过别的主机访问ServerA
口，也可以使用其他程序去反代ServerA的9906端口，还有，我在实际的使用过程中，如果使用远程转发穿透到内网，ssh隧道将会非常不稳定，隧道会莫名其妙的
效，特别是在没有固定IP的网络内，网上有些朋友提供了autossh的解决方案，不过我并没有尝试过，如果你有兴趣，可以试一试。

本地转发与远程转发的区别

读到此处，你可能会有些蒙圈，"远程转发"与"本地转发"到底有什么不一样，我们来对比一下

在对比之前，再强调一点，我们把执行创建隧道命令的主机称为本地主机(本地)。

"本地转发"

在本机执行创建隧道的命令时，本地是ssh客户端，隧道的另一头是远程主机(ssh服务端)，本地主机(也就是ssh客户端)会监听一个端口，当访问本地主机的这个端口
会通过ssh隧道转发到ssh服务端(即远程主机)，远程主机再将通讯数据发往应用服务所监听端口，在本地转发中，本地主机不仅扮演了ssh客户端的角色，也扮演了应
端(比如mysql客户端)，远程主机不仅扮演了ssh服务端，也扮演了应用程序服务端(比如mysql服务端)，那么我们可以总结一下，本地转发的特性如下

本地主机：隧道的一头，本地主机既是ssh客户端，又是应用客户端

远程主机：隧道的另一头，远程主机既是ssh服务端，又是应用服务端

隧道创建以后，转发端口监听在本地主机中，即监听在ssh客户端主机中。

"远程转发"

在本机执行创建隧道的命令时，本地是ssh客户端，隧道的另一头是远程主机(ssh服务端)，远程主机(也就是ssh服务端)会监听一个端口，当访问远程主机的这个端口
会通过ssh隧道转发到ssh客户端(即本地主机)，本地主机再将通讯数据发往应用服务所监听端口，在远程转发中，本地主机不仅扮演了ssh客户端的角色，也扮演了应
端(比如mysql服务端)，远程主机不仅扮演了ssh服务端，也扮演了应用程序客户端(比如mysql客户端)，那么我们可以总结一下，远程转发的特性如下

本地主机：隧道的一头，本地主机既是ssh客户端，又是应用服务端

远程主机：隧道的另一头，远程主机既是ssh服务端，又是应用客户端

隧道创建以后，转发端口监听在远程主机中，即监听在ssh服务端主机中。

"本地转发"与"远程转发"都属于ssh端口转发，也可以称呼它们为"ssh隧道"，只不过，有的朋友喜欢将"远程转发"称呼为为"ssh反向隧道"或者"ssh逆向隧道"

经过上述描述，我想你应该已经明白了它们之间的区别。

一些扩展

在之前的示例中，ServerB是ssh隧道的一头，同时，ServerB也是应用的服务端，也就是说，应用程序的服务端与ssh隧道的连接端在同一台服务器上，那么，当应用
端处于其他主机时（比如ServerC），我们还能够通过ServerB去转发通讯数据吗？我们来动手试试，不过在动手之前，先来描述一下实验场景，实验场景如下图所示

如上图所示，我们想要在A与B之间创建隧道，最终通过隧道访问到ServerC中的mysql服务。

ServerAIP：10.1.0.1

ServerBIP：10.1.0.2

ServerCIP：10.1.0.3

ServerA与ServerB上没有开启任何mysql服务。

ServerC中开启了mysql服务，监听了3306端口。

之前用于示例所创建的ssh隧道已经全部关闭，相当于一个全新的实验环境。

好了，实验环境描述完毕，现在开始实际操作，就以本地转发为例，在ServerA中执行如下命令，即可创建一条隧道并满足上图中的应用场景。

如上图所示，ServerA的9906端口已经被监听，细心如你一定发现了，上图中的命令与之前创建隧道时所使用的命令在结构上并没有什么不同，只是目标端口所对应了ServerC的IP，是不是很简单，我再来啰嗦一遍，上述命令表示，从本机（ServerA）建立一条ssh隧道连接到10.1.0.2（ServerB），隧道使用本地转发模式建立，监听在本地的9906端口上，访问本机的9906端口时，数据会被ssh隧道转发到10.1.0.3（ServerC)的3306端口。

我们来测试一下实际的使用效果，如下图所示，一切正常。

上述场景中存在一个问题，就是数据安全性的问题，我们之所以使用ssh隧道，就是为了用它来保护明文传输的数据，从而提升安全性，不过，在上例的场景中，只有ServerB之间的传输是受ssh隧道保护的，ServerB与ServerC之间的传输，仍然是明文的，所以，如果想要在上述场景中使用ssh隧道进行数据转发，首先要考虑ServerB之间的网络是否可靠。

其实，当我们在创建隧道时如果开启了网关功能，那么应用客户端与ServerA之间的通讯也会面临同样的问题，如下图所示

既然上述场景中存在没有办法通过ssh隧道保护的连接，那么为什么还要使用上述方式进行转发呢？

这是因为，在某些实际的使用场景中，我们使用ssh隧道的目的并不是提升数据的安全性，而是为了"绕过防火墙"，比如如下场景

上图中，ServerC中提供了mysql服务，我们想要通过ServerA访问ServerC中的mysql服务，但是，ServerA与ServerC之间存在防火墙，阻断了它们的通讯，所以ServerA中直接访问ServerC中的服务，不过幸运的是，我们还有另外一台机器：ServerB，ServerA与ServerB之间可以自由通讯，同时，ServerB与ServerC之间讯，没错，你一定想到了，我们可以利用ServerB，在ServerA与ServerB之间建立ssh隧道，达到我们的最终目的：使得ServerA可以访问到ServerC中的mysql服务

当上图中的ssh隧道建立以后，访问ServerA中的转发端口，即可访问到ServerC中的mysql服务，因为对于ServerC来说，ServerA是透明的，ServerC并不知道Se它只能看到ServerB，当你在ServerA中使用ssh隧道访问ServerC的mysql服务时，如果你在ServerC中的网卡上进行抓包，只会看到ServerB的IP地址，因为数据经发了。

一些配置

其实，如果想要能够正常的使用ssh端口转发，我们还需要做出正确的配置才行，之前一直没有说明，是因为openssh默认的配置就是支持端口转发的。

如果想要ssh端口转发能够正常工作，需要在ssh服务端的配置文件中将AllowTcpForwarding的值设置为yes。

此处所指的ssh服务端即ssh隧道中的一头，扮演ssh服务端角色的那台主机。

小结

经过上述描述，我想你应该已经了解的ssh隧道的作用。

通常，ssh隧道可以帮助我们达到如下目的：

1、保护tcp会话，保护会话中明文传输的内容。

2、绕过防火墙或者穿透到内网，访问对应的服务。

为了以后方便回顾，我们将上文中使用到的命令及选项进行总结

创建隧道时的常用选项有：

"-L选项"：表示使用本地端口转发创建ssh隧道

"-R选项"：表示使用远程端口转发创建ssh隧道

"-N选项"： 表示创建隧道以后不连接到sshServer端，通常与"-f"选项连用

"-f选项"：表示在后台运行ssh隧道，通常与"-N"选项连用

"-g选项"：表示ssh隧道对应的转发端口将监听在主机的所有IP中，不使用"-g选项"时，转发端口默认只监听在主机的本地回环地址中，"-g"表示开启网关模式，远程
无法开启网关功能。

创建本地转发模式的ssh隧道，命令如下

```
1 ssh -g -f -N -L forwardingPort:targetIP:targetPort user@sshServerIP
```

本机上的forwardingPort将会被监听，访问本机的forwardingPort，就相当于访问targetIP的targetPort，ssh隧道建立在本机与sshServer之间。

创建远程转发模式的ssh隧道，命令如下

```
1 ssh -f -N -R forwardingPort:targetIP:targetPort user@sshServerIP
```

sshServer上的forwardingPort将会被监听，访问sshServer上的forwardingPort，就相当于访问targetIP的targetPort，ssh隧道建立在本机与sshServer之间。

关于ssh的端口转发就总结到这里，希望可以帮助到你。