# Create User Defined Routes (UDR) in PowerShell

This article covers the Resource Manager deployment model. You can also [create UDRs in the classic deployment model](/documentation/articles/virtual-network-create-udr-classic-ps).

The sample PowerShell commands below expect a simple environment already created based on the scenario above. If you want to run the commands as they are displayed in this document, first build the test environment by deploying [this template](http://github.com/telmosampaio/azure-templates/tree/master/IaaS-NSG-UDR-Before), click **Deploy to Azure**, replace the default parameter values if necessary, and follow the instructions in the portal.

## Create the UDR for the front end subnet

To create the route table and route needed for the front end subnet based on the scenario above, follow the steps below.

1. Create a route used to send all traffic destined to the back end subnet (192.168.2.0/24) to be routed to the **FW1** virtual appliance (192.168.0.4).

* $route = New-AzureRMRouteConfig -Name RouteToBackEnd `  
   -AddressPrefix 192.168.2.0/24 -NextHopType VirtualAppliance `  
   -NextHopIpAddress 192.168.0.4

1. Create a route table named **UDR-FrontEnd** in the **westus** region that contains the route created above.

* $routeTable = New-AzureRMRouteTable -ResourceGroupName TestRG -Location westus `  
   -Name UDR-FrontEnd -Route $route

1. Create a variable that contains the VNet where the subnet is. In our scenario, the VNet is named **TestVNet**.

* $vnet = Get-AzureRMVirtualNetwork -ResourceGroupName TestRG -Name TestVNet

1. Associate the route table created above to the **FrontEnd** subnet.

* Set-AzureRMVirtualNetworkSubnetConfig -VirtualNetwork $vnet -Name FrontEnd `  
   -AddressPrefix 192.168.1.0/24 -RouteTable $routeTable
* Expected output:
* Name : TestVNet  
  ResourceGroupName : TestRG  
  Location : westus  
  Id : /subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourceGroups/TestRG/providers/Microsoft.Network/virtualNetworks/TestVNet  
  Etag : W/"7df26c0e-652f-4754-bc4e-733fef7d5b2b"  
  ProvisioningState : Succeeded  
  Tags :   
   Name Value  
   =========== =====  
   displayName VNet   
    
  AddressSpace : {  
   "AddressPrefixes": [  
   "192.168.0.0/16"  
   ]  
   }  
  DhcpOptions : {  
   "DnsServers": null  
   }  
  NetworkInterfaces : null  
  Subnets : [  
   ...,  
   {  
   "Name": "FrontEnd",  
   "Etag": "W/\"xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx\"",  
   "Id": "/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourceGroups/TestRG/providers/Microsoft.Network/virtualNetworks/TestVNet/subnets/FrontEnd",  
   "AddressPrefix": "192.168.1.0/24",  
   "IpConfigurations": [  
   {  
   "Id": "/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourceGroups/TestRG/providers/Microsoft.Network/networkInterfaces/NICWEB2/ipConfigurations/ipconfig1"  
   },  
   {  
   "Id": "/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourceGroups/TestRG/providers/Microsoft.Network/networkInterfaces/NICWEB1/ipConfigurations/ipconfig1"  
   }  
   ],  
   "NetworkSecurityGroup": {  
   "Id": "/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourceGroups/TestRG/providers/Microsoft.Network/networkSecurityGroups/NSG-BackEnd"  
   },  
   "RouteTable": {  
   "Id": "/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourceGroups/TestRG/providers/Microsoft.Network/routeTables/UDR-FrontEnd"  
   },  
   "ProvisioningState": "Succeeded"  
   },  
   ...  
   ]

## Create the UDR for the back end subnet

To create the route table and route needed for the back end subnet based on the scenario above, follow the steps below.

1. Create a route used to send all traffic destined to the front end subnet (192.168.1.0/24) to be routed to the **FW1** virtual appliance (192.168.0.4).

* $route = New-AzureRMRouteConfig -Name RouteToFrontEnd `  
   -AddressPrefix 192.168.1.0/24 -NextHopType VirtualAppliance `  
   -NextHopIpAddress 192.168.0.4

1. Create a route table named **UDR-BackEnd** in the **uswest** region that contains the route created above.

* $routeTable = New-AzureRMRouteTable -ResourceGroupName TestRG -Location westus `  
   -Name UDR-BackEnd -Route $route

1. Associate the route table created above to the **BackEnd** subnet.

* Set-AzureRMVirtualNetworkSubnetConfig -VirtualNetwork $vnet -Name BackEnd `  
   -AddressPrefix 192.168.2.0/24 -RouteTable $routeTable
* Expected output:
* Name : TestVNet  
  ResourceGroupName : TestRG  
  Location : westus  
  Id : /subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourceGroups/TestRG/providers/Microsoft.Network/virtualNetworks/TestVNet  
  Etag : W/"xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"  
  ProvisioningState : Succeeded  
  Tags :   
   Name Value  
   =========== =====  
   displayName VNet   
    
  AddressSpace : {  
   "AddressPrefixes": [  
   "192.168.0.0/16"  
   ]  
   }  
  DhcpOptions : {  
   "DnsServers": null  
   }  
  NetworkInterfaces : null  
  Subnets : [  
   ...,  
   {  
   "Name": "BackEnd",  
   "Etag": "W/\"xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx\"",  
   "Id": "/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourceGroups/TestRG/providers/Microsoft.Network/virtualNetworks/TestVNet/subnets/BackEnd",  
   "AddressPrefix": "192.168.2.0/24",  
   "IpConfigurations": [  
   {  
   "Id": "/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourceGroups/TestRG/providers/Microsoft.Network/networkInterfaces/NICSQL2/ipConfigurations/ipconfig1"  
   },  
   {  
   "Id": "/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourceGroups/TestRG/providers/Microsoft.Network/networkInterfaces/NICSQL1/ipConfigurations/ipconfig1"  
   }  
   ],  
   "NetworkSecurityGroup": {  
   "Id": "/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourceGroups/TestRG/providers/Microsoft.Network/networkSecurityGroups/NSG-FrontEnd"  
   },  
   "RouteTable": {  
   "Id": "/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourceGroups/TestRG/providers/Microsoft.Network/routeTables/UDR-BackEnd"  
   },  
   "ProvisioningState": "Succeeded"  
   }  
   ]

## Enable IP forwarding on FW1

To enable IP forwarding in the NIC used by **FW1**, follow the steps below.

1. Create a variable that contains the settings for the NIC used by FW1. In our scenario, the NIC is named **NICFW1**.

* $nicfw1 = Get-AzureRMNetworkInterface -ResourceGroupName TestRG -Name NICFW1

1. Enable IP forwarding, and save the NIC settings.

* $nicfw1.EnableIPForwarding = 1  
  Set-AzureRMNetworkInterface -NetworkInterface $nicfw1
* Expected output:
* Name : NICFW1  
  ResourceGroupName : TestRG  
  Location : westus  
  Id : /subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourceGroups/TestRG/providers/Microsoft.Network/networkInterfaces/NICFW1  
  Etag : W/"xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"  
  ProvisioningState : Succeeded  
  Tags :   
   Name Value   
   =========== =======================  
   displayName NetworkInterfaces - DMZ  
    
  VirtualMachine : {  
   "Id": "/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourceGroups/TestRG/providers/Microsoft.Compute/virtualMachines/FW1"  
   }  
  IpConfigurations : [  
   {  
   "Name": "ipconfig1",  
   "Etag": "W/\"xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx\"",  
   "Id": "/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourceGroups/TestRG/providers/Microsoft.Network/networkInterfaces/NICFW1/ipConfigurations/ipconfig1",  
   "PrivateIpAddress": "192.168.0.4",  
   "PrivateIpAllocationMethod": "Static",  
   "Subnet": {  
   "Id": "/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourceGroups/TestRG/providers/Microsoft.Network/virtualNetworks/TestVNet/subnets/DMZ"  
   },  
   "PublicIpAddress": {  
   "Id": "/subscriptions/xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx/resourceGroups/TestRG/providers/Microsoft.Network/publicIPAddresses/PIPFW1"  
   },  
   "LoadBalancerBackendAddressPools": [],  
   "LoadBalancerInboundNatRules": [],  
   "ProvisioningState": "Succeeded"  
   }  
   ]  
  DnsSettings : {  
   "DnsServers": [],  
   "AppliedDnsServers": [],  
   "InternalDnsNameLabel": null,  
   "InternalFqdn": null  
   }  
  EnableIPForwarding : True  
  NetworkSecurityGroup : null  
  Primary : True