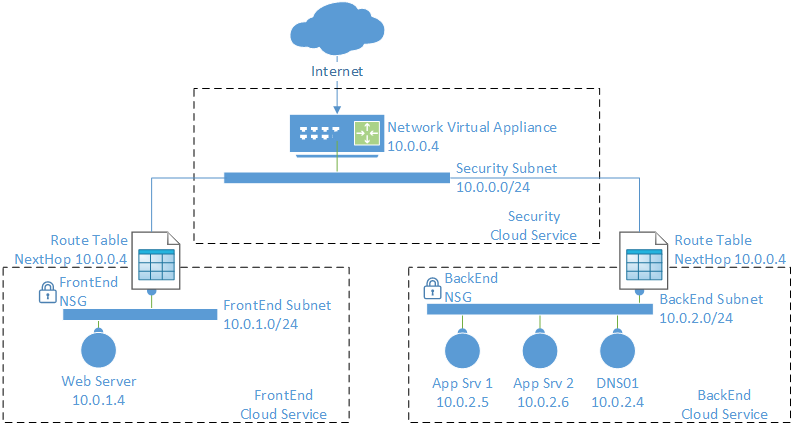
# Example 3 – Build a DMZ to Protect Networks with a Firewall, UDR, and NSG

[Return to the Security Boundary Best Practices Page](/documentation/articles/best-practices-network-security)

This example will create a DMZ with a firewall, four windows servers, User Defined Routing, IP Forwarding, and Network Security Groups. It will also walk through each of the relevant commands to provide a deeper understanding of each step. There is a also a Traffic Scenario section to provide a in-depth step-by-step how traffic proceeds through the layers of defense in the DMZ. Finally, in the references section is the complete code and instruction to build this environment to test and experiment with various scenarios.



Bi-directional DMZ with NVA, NSG, and UDR

## Environment Setup

In this example there is a subscription that contains the following:

* Three cloud services: “SecSvc001”, “FrontEnd001”, and “BackEnd001”
* A Virtual Network, “CorpNetwork”, with three subnets; “SecNet”, “FrontEnd”, and “BackEnd”
* A network virtual appliance, in this example a firewall, connected to the SecNet subnet
* A Windows Server that represents an application web server (“IIS01”)
* Two windows servers that represent application back end servers (“AppVM01”, “AppVM02”)
* A Windows server that represents a DNS server (“DNS01”)

In the references section below there is a PowerShell script that will build most of the environment described above. Building the VMs and Virtual Networks, although are done by the example script, are not described in detail in this document.

To build the environment;

1. Save the network config xml file included in the references section (updated with names, location, and IP addresses to match the given scenario)
2. Update the user variables in the script to match the environment the script is to be run against (subscriptions, service names, etc)
3. Execute the script in PowerShell

**Note**: The region signified in the PowerShell script must match the region signified in the network configuration xml file.

Once the script runs successfully the following post-script steps may be taken;

1. Set up the firewall rules, this is covered in the section below titled: Firewall Rule Description.
2. Optionally in the references section are two scripts to set up the web server and app server with a simple web application to allow testing with this DMZ configuration.

Once the script runs successfully the firewall rules will need to be completed, this is covered in the section titled: Firewall Rules.

## User Defined Routing (UDR)

By default, the following system routes are defined as:

Effective routes :   
 Address Prefix Next hop type Next hop IP address Status Source   
 -------------- ------------- ------------------- ------ ------   
 {10.0.0.0/16} VNETLocal Active Default   
 {0.0.0.0/0} Internet Active Default   
 {10.0.0.0/8} Null Active Default   
 {100.64.0.0/10} Null Active Default   
 {172.16.0.0/12} Null Active Default   
 {192.168.0.0/16} Null Active Default

The VNETLocal is always the defined address prefix(es) of the VNet for that specific network (ie it will change from VNet to VNet depending on how each specific VNet is defined). The remaining system routes are static and default as above.

As for priority, routes are processed via the Longest Prefix Match (LPM) method, thus the most specific route in the table would apply to a given destination address.

Therefore, traffic (for example to the DNS01 server, 10.0.2.4) intended for the local network (10.0.0.0/16) would be routed across the VNet to its destination due to the 10.0.0.0/16 route. In other words, for 10.0.2.4, the 10.0.0.0/16 route is the most specific route, even though the 10.0.0.0/8 and 0.0.0.0/0 also could apply, but since they are less specific they don’t affect this traffic. Thus the traffic to 10.0.2.4 would have a next hop of the local VNet and simply route to the destination.

If traffic was intended for 10.1.1.1 for example, the 10.0.0.0/16 route wouldn’t apply, but the 10.0.0.0/8 would be the most specific, and this the traffic would be dropped (“black holed”) since the next hop is Null.

If the destination didn’t apply to any of the Null prefixes or the VNETLocal prefixes, then it would follow the least specific route, 0.0.0.0/0 and be routed out to the Internet as the next hop and thus out Azure’s internet edge.

If there are two identical prefixes in the route table, the following is the order of preference based on the routes “source” attribute:

1. = A User Defined Route manually added to the table
2. “VPNGateway” = A Dynamic Route (BGP when used with hybrid networks), added by a dynamic network protocol, these routes may change over time as the dynamic protocol automatically reflects changes in peered network
3. “Default” = The System Routes, the local VNet and the static entries as shown in the route table above.

[AZURE.NOTE] There is a limitation using User Defined Routing (UDR) and ExpressRoute due to the complexity of dynamic routing used on the Azure Virtual Gateway. Subnets communicating to the Azure Gateway providing the ExpressRoute connection should not have UDR applied. Also, the Azure Gateway cannot be the NextHop device for other UDR bound subnets. The ability to fully integrate UDR and ExpressRoute will be enabled in a future Azure release.

#### Creating the local routes

In this example, two routing tables are needed, one each for the Frontend and Backend subnets. Each table is loaded with static routes appropriate for the given subnet. For the purpose of this example, each table has three routes:

1. Local subnet traffic with no Next Hop defined to allow local subnet traffic to bypass the firewall
2. Virtual Network traffic with a Next Hop defined as firewall, this overrides the default rule that allows local VNet traffic to route directly
3. All remaining traffic (0/0) with a Next Hop defined as the firewall

Once the routing tables are created they are bound to their subnets. For the Frontend subnet routing table, once created and bound to the subnet should look like this:

Effective routes :   
 Address Prefix Next hop type Next hop IP address Status Source   
 -------------- ------------- ------------------- ------ ------   
 {10.0.1.0/24} VNETLocal Active   
 {10.0.0.0/16} VirtualAppliance 10.0.0.4 Active   
 {0.0.0.0/0} VirtualAppliance 10.0.0.4 Active

For this example, the following commands are used to build the route table, add a user defined route, and then bind the route table to a subnet (Note; any items below beginning with a dollar sign (e.g.: $BESubnet) are user defined variables from the script in the reference section of this document):

1. First the base routing table must be created. This snippet shows the creation of the table for the Backend subnet. In the script, a corresponding table is also created for the Frontend subnet.

* New-AzureRouteTable -Name $BERouteTableName `  
   -Location $DeploymentLocation `  
   -Label "Route table for $BESubnet subnet"

1. Once the route table is created, specific user defined routes can be added. In this snipped, all traffic (0.0.0.0/0) will be routed through the virtual appliance (a variable, $VMIP[0], is used to pass in the IP address assigned when the virtual appliance was created earlier in the script). In the script, a corresponding rule is also created in the Frontend table.

* Get-AzureRouteTable $BERouteTableName |`  
   Set-AzureRoute -RouteName "All traffic to FW" -AddressPrefix 0.0.0.0/0 `  
   -NextHopType VirtualAppliance `  
   -NextHopIpAddress $VMIP[0]

1. The above route entry will override the default “0.0.0.0/0” route, but the default 10.0.0.0/16 rule still existing which would allow traffic within the VNet to route directly to the destination and not to the Network Virtual Appliance. To correct this behavior the follow rule must be added.

* Get-AzureRouteTable $BERouteTableName `  
   |Set-AzureRoute -RouteName "Internal traffic to FW" -AddressPrefix $VNetPrefix `  
   -NextHopType VirtualAppliance `  
   -NextHopIpAddress $VMIP[0]

1. At this point there is a choice to be made. With the above two routes all traffic will route to the firewall for assessment, even traffic within a single subnet. This may be desired, however to allow traffic within a subnet to route locally without involvement from the firewall a third, very specific rule can be added. This route states that any address destine for the local subnet can just route there directly (NextHopType = VNETLocal).

* Get-AzureRouteTable $BERouteTableName `  
   |Set-AzureRoute -RouteName "Allow Intra-Subnet Traffic" -AddressPrefix $BEPrefix `  
   -NextHopType VNETLocal

1. Finally, with the routing table created and populated with a user defined routes, the table must now be bound to a subnet. In the script, the front end route table is also bound to the Frontend subnet. Here is the binding script for the back end subnet.

* Set-AzureSubnetRouteTable -VirtualNetworkName $VNetName `  
   -SubnetName $BESubnet `  
   -RouteTableName $BERouteTableName

## IP Forwarding

A companion feature to UDR, is IP Forwarding. This is a setting on a Virtual Appliance that allows it to receive traffic not specifically addressed to the appliance and then forward that traffic to its ultimate destination.

As an example, if traffic from AppVM01 makes a request to the DNS01 server, UDR would route this to the firewall. With IP Forwarding enabled, the traffic for the DNS01 destination (10.0.2.4) will be accepted by the appliance (10.0.0.4) and then forwarded to its ultimate destination (10.0.2.4). Without IP Forwarding enabled on the Firewall, traffic would not be accepted by the appliance even though the route table has the firewall as the next hop.

[AZURE.IMPORTANT] It’s critical to remember to enable IP Forwarding in conjunction with User Defined Routing.

Setting up IP Forwarding is a single command and can be done at VM creation time. For the flow of this example the code snippet is towards the end of the script and grouped with the UDR commands:

1. Call the VM instance that is your virtual appliance, the firewall in this case, and enable IP Forwarding (Note; any item in red beginning with a dollar sign (e.g.: $VMName[0]) is a user defined variable from the script in the reference section of this document. The zero in brackets, [0], represents the first VM in the array of VMs, for the example script to work without modification, the first VM (VM 0) must be the firewall):

* Get-AzureVM -Name $VMName[0] -ServiceName $ServiceName[0] `  
   |Set-AzureIPForwarding -Enable

## Network Security Groups (NSG)

In this example, a NSG group is built and then loaded with a single rule. This group is then bound only to the Frontend and Backend subnets (not the SecNet). Declaratively the following rule is being built:

1. Any traffic (all ports) from the Internet to the entire VNet (all subnets) is Denied

Although NSGs are used in this example, it’s main purpose is as a secondary layer of defense against manual misconfiguration. We want to block all inbound traffic from the internet to either the Frontend or Backend subnets, traffic should only flow through the SecNet subnet to the firewall (and then if appropriate on to the Frontend or Backend subnets). Plus, with the UDR rules in place, any traffic that did make it into the Frontend or Backend subnets would be directed out to the firewall (thanks to UDR). The firewall would see this as an asymmetric flow and would drop the outbound traffic. Thus there are three layers of security protecting the Frontend and Backend subnets; 1) no open endpoints on the FrontEnd001 and BackEnd001 cloud services, 2) NSGs denying traffic from the Internet, 3) the firewall dropping asymmetric traffic.

One interesting point regarding the Network Security Group in this example is that it contains only one rule, shown below, which is to deny internet traffic to the entire virtual network which would include the Security subnet.

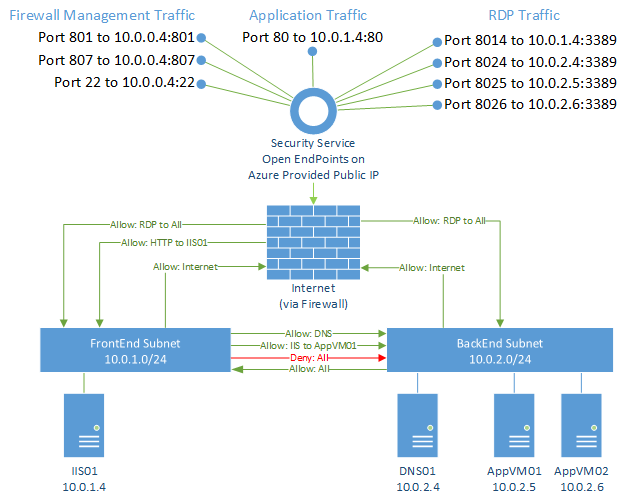
Get-AzureNetworkSecurityGroup -Name $NSGName | `  
 Set-AzureNetworkSecurityRule -Name "Isolate the $VNetName VNet `  
 from the Internet" `  
 -Type Inbound -Priority 100 -Action Deny `  
 -SourceAddressPrefix INTERNET -SourcePortRange '\*' `  
 -DestinationAddressPrefix VIRTUAL\_NETWORK `  
 -DestinationPortRange '\*' `  
 -Protocol \*

However, since the NSG is only bound to the Frontend and Backend subnets, the rule isn’t processed on traffic inbound to the Security subnet. As a result, even though the NSG rule says no Internet traffic to any address on the VNet, because the NSG was never bound to the Security subnet, traffic will flow to the Security subnet.

Set-AzureNetworkSecurityGroupToSubnet -Name $NSGName `  
 -SubnetName $FESubnet -VirtualNetworkName $VNetName  
  
Set-AzureNetworkSecurityGroupToSubnet -Name $NSGName `  
 -SubnetName $BESubnet -VirtualNetworkName $VNetName

## Firewall Rules

On the firewall, forwarding rules will need to be created. Since the firewall is blocking or forwarding all inbound, outbound, and intra-VNet traffic many firewall rules are needed. Also, all inbound traffic will hit the Security Service public IP address (on different ports), to be processed by the firewall. A best practice is to diagram the logical flows before setting up the subnets and firewall rules to avoid rework later. The following figure is a logical view of the firewall rules for this example:



Logical View of the Firewall Rules

[AZURE.NOTE] Based on the Network Virtual Appliance used, the management ports will vary. In this example a Barracuda NG Firewall is referenced which uses ports 22, 801, and 807. Please consult the appliance vendor documentation to find the exact ports used for management of the device being used.

### Logical Rule Description

In the logical diagram above, the security subnet is not shown since the firewall is the only resource on that subnet, and this diagram is showing the firewall rules and how they logically allow or deny traffic flows and not the actual routed path. Also, the external ports selected for the RDP traffic are higher ranged ports (8014 – 8026) and were selected to somewhat align with the last two octets of the local IP address for easier readability (e.g. local server address 10.0.1.4 is associated with external port 8014), however any higher non-conflicting ports could be used.

For this example, we need 7 types of rules, these rule types are described as follows:

* External Rules (for inbound traffic):

1. Firewall Management Rule: This App Redirect rule allows traffic to pass to the management ports of the network virtual appliance.
2. RDP Rules (for each windows server): These four rules (one for each server) will allow management of the individual servers via RDP. This could also be bundled into one rule depending on the capabilities of the Network Virtual Appliance being used.
3. Application Traffic Rules: There are two Application Traffic Rules, the first for the front end web traffic, and the second for the back end traffic (eg web server to data tier). The configuration of these rules will depend on the network architecture (where your servers are placed) and traffic flows (which direction the traffic flows, and which ports are used).
   * The first rule will allow the actual application traffic to reach the application server. While the other rules allow for security, management, etc., Application Rules are what allow external users or services to access the application(s). For this example, there is a single web server on port 80, thus a single firewall application rule will redirect inbound traffic to the external IP, to the web servers internal IP address. The redirected traffic session would be NAT’d to the internal server.
   * The second Application Traffic Rule is the back end rule to allow the Web Server to talk to the AppVM01 server (but not AppVM02) via any port.

* Internal Rules (for intra-VNet traffic)

1. Outbound to Internet Rule: This rule will allow traffic from any network to pass to the selected networks. This rule is usually a default rule already on the firewall, but in a disabled state. This rule should be enabled for this example.
2. DNS Rule: This rule allows only DNS (port 53) traffic to pass to the DNS server. For this environment most traffic from the Frontend to the Backend is blocked, this rule specifically allows DNS from any local subnet.
3. Subnet to Subnet Rule: This rule is to allow any server on the back end subnet to connect to any server on the front end subnet (but not the reverse).

* Fail-safe Rule (for traffic that doesn’t meet any of the above):

1. Deny All Traffic Rule: This should always be the final rule (in terms of priority), and as such if a traffic flows fails to match any of the preceding rules it will be dropped by this rule. This is a default rule and usually activated, no modifications are generally needed.

[AZURE.TIP] On the second application traffic rule, any port is allowed for easy of this example, in a real scenario the most specific port and address ranges should be used to reduce the attack surface of this rule.

[AZURE.IMPORTANT] Once all of the above rules are created, it’s important to review the priority of each rule to ensure traffic will be allowed or denied as desired. For this example, the rules are in priority order. It’s easy to be locked out of the firewall due to mis-ordered rules. At a minimum, ensure the management for the firewall itself is always the absolute highest priority rule.

### Rule Prerequisites

One prerequisite for the Virtual Machine running the firewall are public endpoints. For the firewall to process traffic the appropriate public endpoints must be open. There are three types of traffic in this example; 1) Management traffic to control the firewall and firewall rules, 2) RDP traffic to control the windows servers, and 3) Application Traffic. These are the three columns of traffic types in the upper half of logical view of the firewall rules above.

[AZURE.IMPORTANT] A key takeway here is to remember that **all** traffic will come through the firewall. So to remote desktop to the IIS01 server, even though it’s in the Front End Cloud Service and on the Front End subnet, to access this server we will need to RDP to the firewall on port 8014, and then allow the firewall to route the RDP request internally to the IIS01 RDP Port. The Azure Management Portal’s “Connect” button won’t work because there is no direct RDP path to IIS01 (as far as the portal can see). This means all connections from the internet will be to the Security Service and a Port, e.g. secscv001.chinacloudapp.cn:xxxx (reference the above diagram for the mapping of External Port and Internal IP and Port).

An endpoint can be opened either at the time of VM creation or post build as is done in the example script and shown below in this code snippet (Note; any item beginning with a dollar sign (e.g.: i]) is a user defined variable from the script in the reference section of this document. The “i], represents the array number of a specific VM in an array of VMs):

Add-AzureEndpoint -Name "HTTP" -Protocol tcp -PublicPort 80 -LocalPort 80 `  
 -VM (Get-AzureVM -ServiceName $ServiceName[$i] -Name $VMName[$i]) | `  
 Update-AzureVM

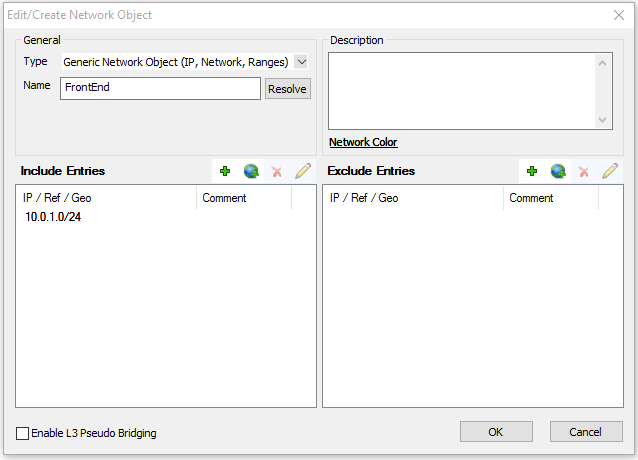
Although not clearly shown here due to the use of variables, but endpoints are **only** opened on the Security Cloud Service. This is to ensure that all inbound traffic is handled (routed, NAT’d, dropped) by the firewall.

A management client will need to be installed on a PC to manage the firewall and create the configurations needed. See the documentation from your firewall (or other NVA) vendor on how to manage the device. The remainder of this section and the next section, Firewall Rules Creation, will describe the configuration of the firewall itself, through the vendors management client (i.e. not the Azure Management Portal or PowerShell).

Instructions for client download and connecting to the Barracuda used in this example can be found here: [Barracuda NG Admin](https://techlib.barracuda.com/NG61/NGAdmin)

Once logged onto the firewall but before creating firewall rules, there are two prerequisite object classes that can make creating the rules easier; Network & Service objects.

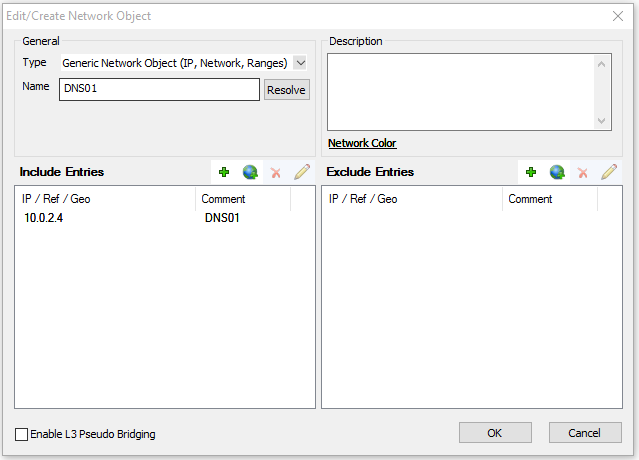
For this example, three named network objects should be defined (one for the Frontend subnet and the Backend subnet, also a network object for the IP address of the DNS server). To create a named network; starting from the Barracuda NG Admin client dashboard, navigate to the configuration tab, in the Operational Configuration section click Ruleset, then click “Networks” under the Firewall Objects menu, then click New in the Edit Networks menu. The network object can now be created, by adding the name and the prefix:



Create a FrontEnd Network Object

This will create a named network for the FrontEnd subnet, a similar object should be created for the BackEnd subnet as well. Now the subnets can be more easily referenced by name in the firewall rules.

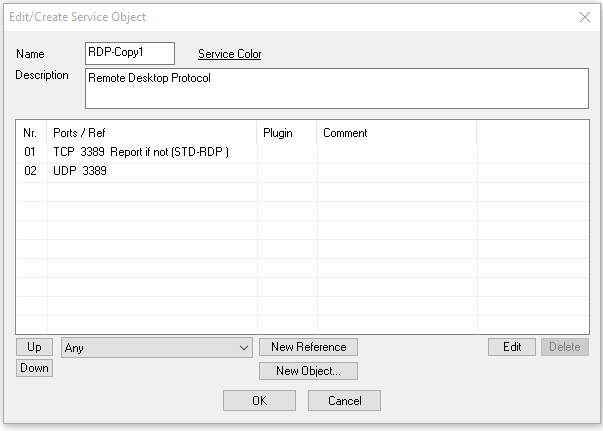
For the DNS Server Object:



Create a DNS Server Object

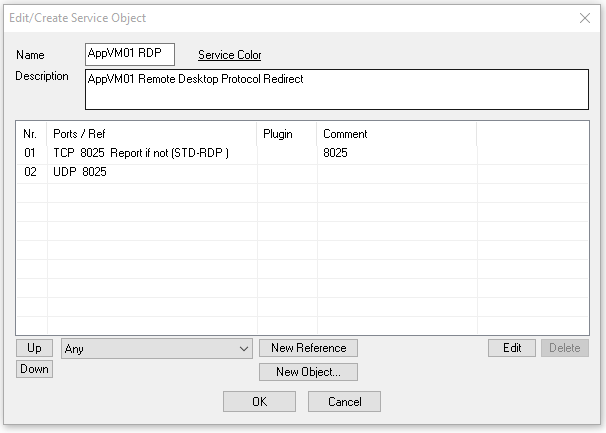
This single IP address reference will be utilized in a DNS rule later in the document.

The second prerequisite objects are Services objects. These will represent the RDP connection ports for each server. Since the existing RDP service object is bound to the default RDP port, 3389, new Services can be created to allow traffic from the external ports (8014-8026). The new ports could also be added to the existing RDP service, but for ease of demonstration, an individual rule for each server can be created. To create a new RDP rule for a server; starting from the Barracuda NG Admin client dashboard, navigate to the configuration tab, in the Operational Configuration section click Ruleset, then click “Services” under the Firewall Objects menu, scroll down the list of services and select the “RDP” service. Right-click and select copy, then right-click and select Paste. There is now a RDP-Copy1 Service Object that can be edited. Right-click RDP-Copy1 and select Edit, the Edit Service Object window will pop up as shown here:



Copy of Default RDP Rule

The values can then be edited to represent the RDP service for a specific server. For AppVM01 the above default RDP rule should be modified to reflect a new Service Name, Description, and external RDP Port used in the Figure 8 diagram (Note: the ports are changed from the RDP default of 3389 to the external port being used for this specific server, in the case of AppVM01 the external Port is 8025) the modified service is shown below:



AppVM01 Rule

This process must be repeated to create RDP Services for the remaining servers; AppVM02, DNS01, and IIS01. The creation of these Services will make the Rule creation simpler and more obvious in the next section.

[AZURE.NOTE] An RDP service for the Firewall is not needed for two reasons; 1) first the firewall VM is a Linux based image so SSH would be used on port 22 for VM management instead of RDP, and 2) port 22, and two other management ports are allowed in the first management rule described below to allow for management connectivity.

### Firewall Rules Creation

There are three types of firewall rules used in this example, they all have distinct icons:

The Application Redirect rule: Application Redirect Icon

The Destination NAT rule: Destination NAT Icon

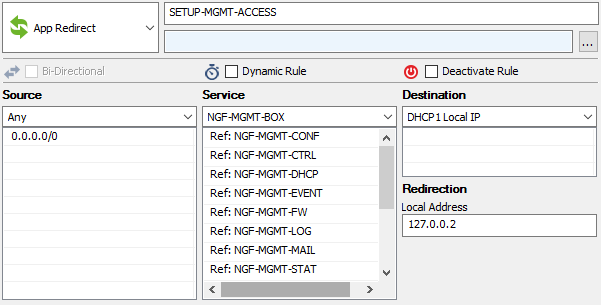
The Pass rule: Pass Icon

More information on these rules can be found at the Barracuda web site.

To create the following rules (or verify existing default rules), starting from the Barracuda NG Admin client dashboard, navigate to the configuration tab, in the Operational Configuration section click Ruleset. A grid called, “Main Rules” will show the existing active and deactivated rules on this firewall. In the upper right corner of this grid is a small, green “+” button, click this to create a new rule (Note: your firewall may be “locked” for changes, if you see a button marked “Lock” and you are unable to create or edit rules, click this button to “unlock” the rule set and allow editing). If you wish to edit an existing rule, select that rule, right-click and select Edit Rule.

Once your rules are created and/or modified, they must be pushed to the firewall and then activated, if this is not done the rule changes will not take effect. The push and activation process is described below the details rule descriptions.

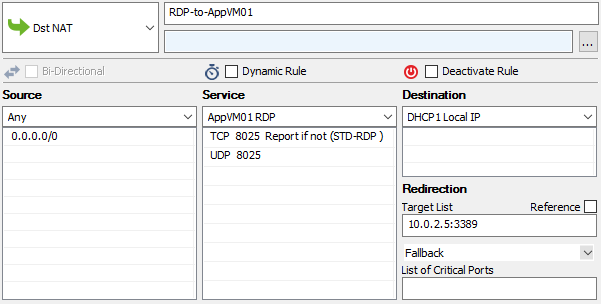
The specifics of each rule required to complete this example are described as follows:

* **Firewall Management Rule**: This App Redirect rule allows traffic to pass to the management ports of the network virtual appliance, in this example a Barracuda NG firewall. The management ports are 801, 807 and optionally 22. The external and internal ports are the same (i.e. no port translation). This rule, SETUP-MGMT-ACCESS, is a default rule and enabled by default (in Barracuda NG Firewall version 6.1).
* 
* Firewall Management Rule

[AZURE.TIP] The source address space in this rule is Any, if the management IP address ranges are known, reducing this scope would also reduce the attack surface to the management ports.

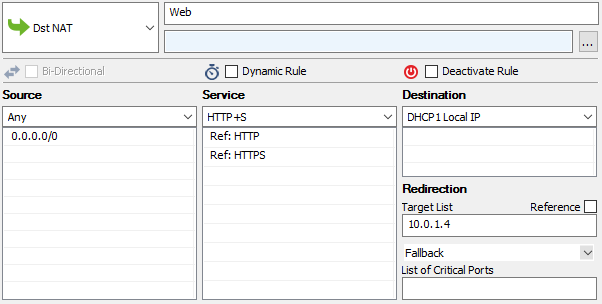
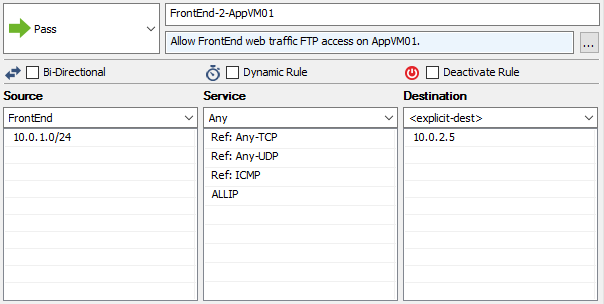
* **RDP Rules**: These Destination NAT rules will allow management of the individual servers via RDP. There are four critical fields needed to create this rule:

1. Source – to allow RDP from anywhere, the reference “Any” is used in the Source field.
2. Service – use the appropriate Service Object created earlier, in this case “AppVM01 RDP”, the external ports redirect to the servers local IP address and to port 3386 (the default RDP port). This specific rule is for RDP access to AppVM01.
3. Destination – should be the \*local port on the firewall\*, “DCHP 1 Local IP” or eth0 if using static IPs. The ordinal number (eth0, eth1, etc) may be different if your network appliance has multiple local interfaces. This is the port the firewall is sending out from (may be the same as the receiving port), the actual routed destination is in the Target List field.
4. Redirection – this section tells the virtual appliance where to ultimately redirect this traffic. The simplest redirection is to place the IP and Port (optional) in the Target List field. If no port is used the destination port on the inbound request will be used (ie no translation), if a port is designated the port will also be NAT’d along with the IP address.

* 
* Firewall RDP Rule
* A total of four RDP rules will need to be created:

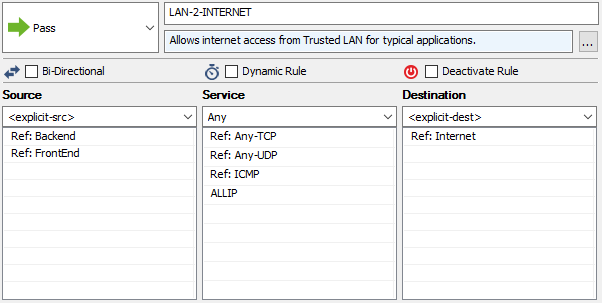
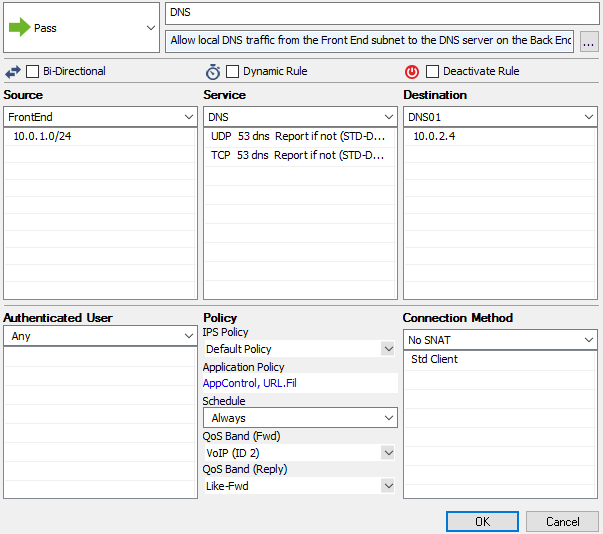
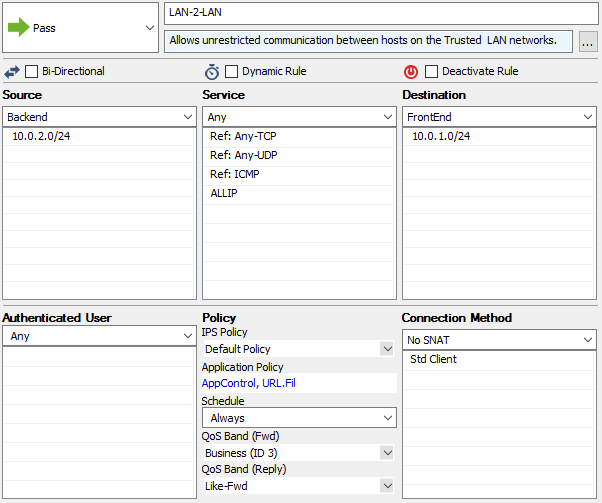
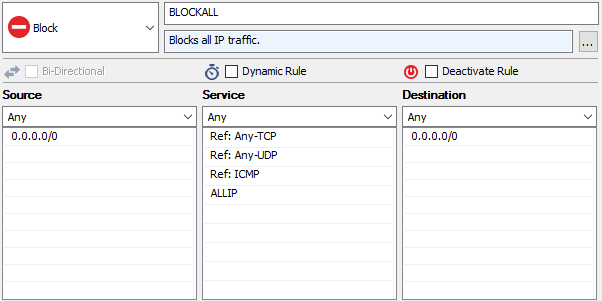
|  |  |  |  |
| --- | --- | --- | --- |
| * Rule Name | * Server | * Service | * Target List |
| * RDP-to-IIS01 | * IIS01 | * IIS01 RDP | * 10.0.1.4:3389 |
| * RDP-to-DNS01 | * DNS01 | * DNS01 RDP | * 10.0.2.4:3389 |
| * RDP-to-AppVM01 | * AppVM01 | * AppVM01 RDP | * 10.0.2.5:3389 |
| * RDP-to-AppVM02 | * AppVM02 | * AppVm02 RDP | * 10.0.2.6:3389 |

[AZURE.TIP] Narrowing down the scope of the Source and Service fields will reduce the attack surface. The most limited scope that will allow functionality should be used.

* **Application Traffic Rules**: There are two Application Traffic Rules, the first for the front end web traffic, and the second for the back end traffic (eg web server to data tier). These rules will depend on the network architecture (where your servers are placed) and traffic flows (which direction the traffic flows, and which ports are used).
* First discussed is the front end rule for web traffic:
* 
* Firewall Web Rule
* This Destination NAT rule allows the actual application traffic to reach the application server. While the other rules allow for security, management, etc., Application Rules are what allow external users or services to access the application(s). For this example, there is a single web server on port 80, thus the single firewall application rule will redirect inbound traffic to the external IP, to the web servers internal IP address.
* **Note**: that there is no port assigned in the Target List field, thus the inbound port 80 (or 443 for the Service selected) will be used in the redirection of the web server. If the web server is listening on a different port, for example port 8080, the Target List field could be updated to 10.0.1.4:8080 to allow for the Port redirection as well.
* The next Application Traffic Rule is the back end rule to allow the Web Server to talk to the AppVM01 server (but not AppVM02) via Any service:
* 
* Firewall AppVM01 Rule
* This Pass rule allows any IIS server on the Frontend subnet to reach the AppVM01 (IP Address 10.0.2.5) on Any port, using any Protocol to access data needed by the web application.
* In this screen shot an “<explicit-dest>” is used in the Destination field to signify 10.0.2.5 as the destination. This could be either explicit as shown or a named Network Object (as was done in the prerequisites for the DNS server). This is up to the administrator of the firewall as to which method will be used. To add 10.0.2.5 as an Explict Desitnation, double-click on the first blank row under <explicit-dest> and enter the address in the window that pops up.
* With this Pass Rule, no NAT is needed since this is internal traffic, so the Connection Method can be set to “No SNAT”.
* **Note**: The Source network in this rule is any resource on the FrontEnd subnet, if there will only be one, or a known specific number of web servers, a Network Object resource could be created to be more specific to those exact IP addresses instead of the entire Frontend subnet.

[AZURE.TIP] This rule uses the service “Any” to make the sample application easier to setup and use, this will also allow ICMPv4 (ping) in a single rule. However, this is not a recommended practice. The ports and protocols (“Services”) should be narrowed to the minimum possible that allows application operation to reduce the attack surface across this boundary.

[AZURE.TIP] Although this rule shows an explicit-dest reference being used, a consistent approach should be used throughout the firewall configuration. It is recommended that the named Network Object be used throughout for easier readability and supportability. The explicit-dest is used here only to show an alternative reference method and is not generally recommended (especially for complex configurations).

* **Outbound to Internet Rule**: This Pass rule will allow traffic from any Source network to pass to the selected Destination networks. This rule is a default rule usually already on the Barracuda NG firewall, but is in a disabled state. Right-clicking on this rule can access the Activate Rule command. The rule shown here has been modified to add the two local subnets that were created as references in the prerequisite section of this document to the Source attribute of this rule.
* 
* Firewall Outbound Rule
* **DNS Rule**: This Pass rule allows only DNS (port 53) traffic to pass to the DNS server. For this environment most traffic from the FrontEnd to the BackEnd is blocked, this rule specifically allows DNS.
* 
* Firewall DNS Rule
* **Note**: In this screen shot the Connection Method is included. Because this rule is for internal IP to internal IP address traffic, no NATing is required, this the Connection Method is set to “No SNAT” for this Pass rule.
* **Subnet to Subnet Rule**: This Pass rule is a default rule that has been activated and modified to allow any server on the back end subnet to connect to any server on the front end subnet. This rule is all internal traffic so the Connection Method can be set to No SNAT.
* 
* Firewall Intra-VNet Rule
* **Note**: The Bi-directional checkbox is not checked (nor is it checked in most rules), this is significant for this rule in that it makes this rule “one directional”, a connection can be initiated from the back end subnet to the front end network, but not the reverse. If that checkbox was checked, this rule would enable bi-directional traffic, which from our logical diagram is not desired.
* **Deny All Traffic Rule**: This should always be the final rule (in terms of priority), and as such if a traffic flows fails to match any of the preceding rules it will be dropped by this rule. This is a default rule and usually activated, no modifications are generally needed.
* 
* Firewall Deny Rule

[AZURE.IMPORTANT] Once all of the above rules are created, it’s important to review the priority of each rule to ensure traffic will be allowed or denied as desired. For this example, the rules are in the order they should appear in the Main Grid of forwarding rules in the Barracuda Management Client.

## Rule Activation

With the ruleset modified to the specification of the logic diagram, the ruleset must be uploaded to the firewall and then activated.



Firewall Rule Activation

In the upper right hand corner of the management client are a cluster of buttons. Click the “Send Changes” button to send the modified rules to the firewall, then click the “Activate” button.

With the activation of the firewall ruleset this example environment build is complete.

## Traffic Scenarios

[AZURE.IMPORTANT] A key takeway is to remember that **all** traffic will come through the firewall. So to remote desktop to the IIS01 server, even though it’s in the Front End Cloud Service and on the Front End subnet, to access this server we will need to RDP to the firewall on port 8014, and then allow the firewall to route the RDP request internally to the IIS01 RDP Port. The Azure Management Portal’s “Connect” button won’t work because there is no direct RDP path to IIS01 (as far as the portal can see). This means all connections from the internet will be to the Security Service and a Port, e.g. secscv001.chinacloudapp.cn:xxxx.

For these scenarios, the following firewall rules should be in place:

1. Firewall Management
2. RDP to IIS01
3. RDP to DNS01
4. RDP to AppVM01
5. RDP to AppVM02
6. App Traffic to the Web
7. App Traffic to AppVM01
8. Outbound to the Internet
9. Frontend to DNS01
10. Intra-Subnet Traffic (back end to front end only)
11. Deny All

The actual firewall ruleset will most likely have many other rules in addition to these, the rules on any given firewall will also have different priority numbers than the ones listed here. This list and associated numbers are to provide relevance between just these eleven rules and the relative priority amongst them. In other words; on the actual firewall, the “RDP to IIS01” may be rule number 5, but as long as it’s below the “Firewall Management” rule and above the “RDP to DNS01” rule it would align with the intention of this list. The list will also aid in the below scenarios allowing brevity; e.g. “FW Rule 9 (DNS)”. Also for brevity, the four RDP rules will be collectively called, “the RDP rules” when the traffic scenario is unrelated to RDP.

Also recall that Network Security Groups are in-place for inbound internet traffic on the Frontend and Backend subnets.

#### (Allowed) Internet to Web Server

1. Internet user requests HTTP page from SecSvc001.CloudApp.Net (Internet Facing Cloud Service)
2. Cloud service passes traffic through open endpoint on port 80 to firewall interface on 10.0.0.4:80
3. No NSG assigned to Security subnet, so system NSG rules allow traffic to firewall
4. Traffic hits internal IP address of the firewall (10.0.1.4)
5. Firewall begins rule processing:
6. FW Rule 1 (FW Mgmt) doesn’t apply, move to next rule
7. FW Rules 2 - 5 (RDP Rules) don’t apply, move to next rule
8. FW Rule 6 (App: Web) does apply, traffic is allowed, firewall NATs it to 10.0.1.4 (IIS01)
9. The Frontend subnet begins inbound rule processing:
10. NSG Rule 1 (Block Internet) doesn’t apply (this traffic was NAT’d by the firewall, thus the source address is now the firewall which is on the Security subnet and seen by the Frontend subnet NSG to be “local” traffic and is thus allowed), move to next rule
11. Default NSG Rules allow subnet to subnet traffic, traffic is allowed, stop NSG rule processing
12. IIS01 is listening for web traffic, receives this request and starts processing the request
13. IIS01 attempts to initiates an FTP session to AppVM01 on Backend subnet
14. The UDR route on Frontend subnet makes the firewall the next hop
15. No outbound rules on Frontend subnet, traffic is allowed
16. Firewall begins rule processing:
17. FW Rule 1 (FW Mgmt) doesn’t apply, move to next rule
18. FW Rule 2 - 5 (RDP Rules) don’t apply, move to next rule
19. FW Rule 6 (App: Web) doesn’t apply, move to next rule
20. FW Rule 7 (App: Backend) does apply, traffic is allowed, firewall forwards traffic to 10.0.2.5 (AppVM01)
21. The Backend subnet begins inbound rule processing:
22. NSG Rule 1 (Block Internet) doesn’t apply, move to next rule
23. Default NSG Rules allow subnet to subnet traffic, traffic is allowed, stop NSG rule processing
24. AppVM01 receives the request and initiates the session and responds
25. The UDR route on Backend subnet makes the firewall the next hop
26. Since there are no outbound NSG rules on the Backend subnet the response is allowed
27. Because this is returning traffic on an established session the firewall passes the response back to the web server (IIS01)
28. Frontend subnet begins inbound rule processing:
29. NSG Rule 1 (Block Internet) doesn’t apply, move to next rule
30. Default NSG Rules allow subnet to subnet traffic, traffic is allowed, stop NSG rule processing
31. The IIS server receives the response, completes the transaction with AppVM01, and then completes building the HTTP response, this HTTP response is sent to the requestor
32. Since there are no outbound NSG rules on the Frontend subnet the response is allowed
33. The HTTP response hits the firewall, and because this is the response to an established NAT session is accepted by the firewall
34. The firewall then redirects the response back to the Internet User
35. Since there are no outbound NSG rules or UDR hops on the Frontend subnet the response is allowed, and the Internet User receives the web page requested.

#### (Allowed) Internet RDP to Backend

1. Server Admin on internet requests RDP session to AppVM01 via SecSvc001.CloudApp.Net:8025, where 8025 is the user assigned port number for the “RDP to AppVM01” firewall rule
2. The cloud service passes traffic through the open endpoint on port 8025 to firewall interface on 10.0.0.4:8025
3. No NSG assigned to Security subnet, so system NSG rules allow traffic to firewall
4. Firewall begins rule processing:
5. FW Rule 1 (FW Mgmt) doesn’t apply, move to next rule
6. FW Rule 2 (RDP IIS) doesn’t apply, move to next rule
7. FW Rule 3 (RDP DNS01) doesn’t apply, move to next rule
8. FW Rule 4 (RDP AppVM01) does apply, traffic is allowed, firewall NATs it to 10.0.2.5:3386 (RDP port on AppVM01)
9. The Backend subnet begins inbound rule processing:
10. NSG Rule 1 (Block Internet) doesn’t apply (this traffic was NAT’d by the firewall, thus the source address is now the firewall which is on the Security subnet and seen by the Backend subnet NSG to be “local” traffic and is thus allowed), move to next rule
11. Default NSG Rules allow subnet to subnet traffic, traffic is allowed, stop NSG rule processing
12. AppVM01 is listening for RDP traffic and responds
13. With no outbound NSG rules, default rules apply and return traffic is allowed
14. UDR routes outbound traffic to the firewall as the next hop
15. Because this is returning traffic on an established session the firewall passes the response back to the internet user
16. RDP session is enabled
17. AppVM01 prompts for user name password

#### (Allowed) Web Server DNS lookup on DNS server

1. Web Server, IIS01, needs a data feed at www.data.gov, but needs to resolve the address.
2. The network configuration for the VNet lists DNS01 (10.0.2.4 on the Backend subnet) as the primary DNS server, IIS01 sends the DNS request to DNS01
3. UDR routes outbound traffic to the firewall as the next hop
4. No outbound NSG rules are bound to the Frontend subnet, traffic is allowed
5. Firewall begins rule processing:
6. FW Rule 1 (FW Mgmt) doesn’t apply, move to next rule
7. FW Rule 2 - 5 (RDP Rules) don’t apply, move to next rule
8. FW Rules 6 & 7 (App Rules) don’t apply, move to next rule
9. FW Rule 8 (To Internet) doesn’t apply, move to next rule
10. FW Rule 9 (DNS) does apply, traffic is allowed, firewall forwards traffic to 10.0.2.4 (DNS01)
11. The Backend subnet begins inbound rule processing:
12. NSG Rule 1 (Block Internet) doesn’t apply, move to next rule
13. Default NSG Rules allow subnet to subnet traffic, traffic is allowed, stop NSG rule processing
14. DNS server receives the request
15. DNS server doesn’t have the address cached and asks a root DNS server on the internet
16. UDR routes outbound traffic to the firewall as the next hop
17. No outbound NSG rules on Backend subnet, traffic is allowed
18. Firewall begins rule processing:
19. FW Rule 1 (FW Mgmt) doesn’t apply, move to next rule
20. FW Rule 2 - 5 (RDP Rules) don’t apply, move to next rule
21. FW Rules 6 & 7 (App Rules) don’t apply, move to next rule
22. FW Rule 8 (To Internet) does apply, traffic is allowed, session is SNAT out to root DNS server on the Internet
23. Internet DNS server responds, since this session was initiated from the firewall, the response is accepted by the firewall
24. As this is an established session, the firewall forwards the response to the initiating server, DNS01
25. The Backend subnet begins inbound rule processing:
26. NSG Rule 1 (Block Internet) doesn’t apply, move to next rule
27. Default NSG Rules allow subnet to subnet traffic, traffic is allowed, stop NSG rule processing
28. The DNS server receives and caches the response, and then responds to the initial request back to IIS01
29. The UDR route on backend subnet makes the firewall the next hop
30. No outbound NSG rules exist on the Backend subnet, traffic is allowed
31. This is an established session on the firewall, the response is forwarded by the firewall back to the IIS server
32. Frontend subnet begins inbound rule processing:
33. There is no NSG rule that applies to Inbound traffic from the Backend subnet to the Frontend subnet, so none of the NSG rules apply
34. The default system rule allowing traffic between subnets would allow this traffic so the traffic is allowed
35. IIS01 receives the response from DNS01

#### (Allowed) Backend server to Frontend server

1. An administrator logged on to AppVM02 via RDP requests a file directly from the IIS01 server via windows file explorer
2. The UDR route on Backend subnet makes the firewall the next hop
3. Since there are no outbound NSG rules on the Backend subnet the response is allowed
4. Firewall begins rule processing:
5. FW Rule 1 (FW Mgmt) doesn’t apply, move to next rule
6. FW Rule 2 - 5 (RDP Rules) don’t apply, move to next rule
7. FW Rules 6 & 7 (App Rules) don’t apply, move to next rule
8. FW Rule 8 (To Internet) doesn’t apply, move to next rule
9. FW Rule 9 (DNS) doesn’t apply, move to next rule
10. FW Rule 10 (Intra-Subnet) does apply, traffic is allowed, firewall passes traffic to 10.0.1.4 (IIS01)
11. Frontend subnet begins inbound rule processing:
12. NSG Rule 1 (Block Internet) doesn’t apply, move to next rule
13. Default NSG Rules allow subnet to subnet traffic, traffic is allowed, stop NSG rule processing
14. Assuming proper authentication and authorization, IIS01 accepts the request and responds
15. The UDR route on Frontend subnet makes the firewall the next hop
16. Since there are no outbound NSG rules on the Frontend subnet the response is allowed
17. As this is an existing session on the firewall this response is allowed and the firewall returns the response to AppVM02
18. Backend subnet begins inbound rule processing:
19. NSG Rule 1 (Block Internet) doesn’t apply, move to next rule
20. Default NSG Rules allow subnet to subnet traffic, traffic is allowed, stop NSG rule processing
21. AppVM02 receives the response

#### (Denied) Internet direct to Web Server

1. Internet user tries to access the web server, IIS01, through the FrontEnd001.CloudApp.Net service
2. Since there are no endpoints open for HTTP traffic, this would not pass through the Cloud Service and wouldn’t reach the server
3. If the endpoints were open for some reason, the NSG (Block Internet) on the Frontend subnet would block this traffic
4. Finally, the Frontend subnet UDR route would send any outbound traffic from IIS01 to the firewall as the next hop, and the firewall would see this as asymmetric traffic and drop the outbound response Thus there are at least three independent layers of defense between the internet and IIS01 via its cloud service preventing unauthorized/inappropriate access.

#### (Denied) Internet to Backend Server

1. Internet user tries to access a file on AppVM01 through the BackEnd001.CloudApp.Net service
2. Since there are no endpoints open for file share, this would not pass the Cloud Service and wouldn’t reach the server
3. If the endpoints were open for some reason, the NSG (Block Internet) would block this traffic
4. Finally, the UDR route would send any outbound traffic from AppVM01 to the firewall as the next hop, and the firewall would see this as asymmetric traffic and drop the outbound response Thus there are at least three independent layers of defense between the internet and AppVM01 via its cloud service preventing unauthorized/inappropriate access.

#### (Denied) Frontend server to Backend Server

1. Assume IIS01 was compromised and is running malicious code trying to scan the Backend subnet servers.
2. The Frontend subnet UDR route would send any outbound traffic from IIS01 to the firewall as the next hop. This is not something that can be altered by the compromised VM.
3. The firewall would process the traffic, if the request was to AppVM01, or to the DNS server for DNS lookups that traffic could potentially be allowed by the firewall (due to FW Rules 7 and 9). All other traffic would be blocked by FW Rule 11 (Deny All).
4. If advanced threat detection was enabled on the firewall (which is not covered in this document, see the vendor documentation for your specific network appliance advanced threat capabilities), even traffic that would be allowed by the basic forwarding rules discussed in this document could be prevented if the traffic contained known signatures or patterns that flag an advanced threat rule.

#### (Denied) Internet DNS lookup on DNS server

1. Internet user tries to lookup an internal DNS record on DNS01 through BackEnd001.CloudApp.Net service
2. Since there are no endpoints open for DNS traffic, this would not pass through the Cloud Service and wouldn’t reach the server
3. If the endpoints were open for some reason, the NSG rule (Block Internet) on the Frontend subnet would block this traffic
4. Finally, the Backend subnet UDR route would send any outbound traffic from DNS01 to the firewall as the next hop, and the firewall would see this as asymmetric traffic and drop the outbound response Thus there are at least three independent layers of defense between the internet and DNS01 via its cloud service preventing unauthorized/inappropriate access.

#### (Denied) Internet to SQL access through Firewall

1. Internet user requests SQL data from SecSvc001.CloudApp.Net (Internet Facing Cloud Service)
2. Since there are no endpoints open for SQL, this would not pass the Cloud Service and wouldn’t reach the firewall
3. If SQL endpoints were open for some reason, the firewall would begin rule processing:
4. FW Rule 1 (FW Mgmt) doesn’t apply, move to next rule
5. FW Rules 2 - 5 (RDP Rules) don’t apply, move to next rule
6. FW Rule 6 & 7 (Application Rules) don’t apply, move to next rule
7. FW Rule 8 (To Internet) doesn’t apply, move to next rule
8. FW Rule 9 (DNS) doesn’t apply, move to next rule
9. FW Rule 10 (Intra-Subnet) doesn’t apply, move to next rule
10. FW Rule 11 (Deny All) does apply, traffic is blocked, stop rule processing

## References

### Main Script and Network Config

Save the Full Script in a PowerShell script file. Save the Network Config into a file named “NetworkConf2.xml”. Modify the user defined variables as needed. Run the script, then follow the Firewall rule setup instruction above.

#### Full Script

This script will, based on the user defined variables;

1. Connect to an Azure subscription
2. Create a new storage account
3. Create a new VNet and three subnets as defined in the Network Config file
4. Build five virtual machines; 1 firewall and 4 windows server VMs
5. Configure UDR including:
6. Creating two new route tables
7. Add routes to the tables
8. Bind tables to appropriate subnets
9. Enable IP Forwarding on the NVA
10. Configure NSG including:
11. Creating a NSG
12. Adding a rule
13. Binding the NSG to the appropriate subnets

This PowerShell script should be run locally on an internet connected PC or server.

[AZURE.IMPORTANT] When this script is run, there may be warnings or other informational messages that pop in PowerShell. Only error messages in red are cause for concern.

<#   
 .SYNOPSIS  
 Example of DMZ and User Defined Routing in an isolated network (Azure only, no hybrid connections)  
  
 .DESCRIPTION  
 This script will build out a sample DMZ setup containing:  
 - A default storage account for VM disks  
 - Three new cloud services  
 - Three Subnets (SecNet, FrontEnd, and BackEnd subnets)  
 - A Network Virtual Appliance (NVA), in this case a Barracuda NG Firewall  
 - One server on the FrontEnd Subnet  
 - Three Servers on the BackEnd Subnet  
 - IP Forwading from the FireWall out to the internet  
 - User Defined Routing FrontEnd and BackEnd Subnets to the NVA  
  
 Before running script, ensure the network configuration file is created in  
 the directory referenced by $NetworkConfigFile variable (or update the  
 variable to reflect the path and file name of the config file being used).  
  
 .Notes  
 Everyone's security requirements are different and can be addressed in a myriad of ways.  
 Please be sure that any sensitive data or applications are behind the appropriate  
 layer(s) of protection. This script serves as an example of some of the techniques  
 that can be used, but should not be used for all scenarios. You are responsible to  
 assess your security needs and the appropriate protections needed, and then effectively  
 implement those protections.  
  
 Security Service (SecNet subnet 10.0.0.0/24)  
 myFirewall - 10.0.0.4  
   
 FrontEnd Service (FrontEnd subnet 10.0.1.0/24)  
 IIS01 - 10.0.1.4  
   
 BackEnd Service (BackEnd subnet 10.0.2.0/24)  
 DNS01 - 10.0.2.4  
 AppVM01 - 10.0.2.5  
 AppVM02 - 10.0.2.6  
  
#>  
  
# Fixed Variables  
 $LocalAdminPwd = Read-Host -Prompt "Enter Local Admin Password to be used for all VMs"  
 $VMName = @()  
 $ServiceName = @()  
 $VMFamily = @()  
 $img = @()  
 $size = @()  
 $SubnetName = @()  
 $VMIP = @()  
  
# User Defined Global Variables  
 # These should be changes to reflect your subscription and services  
 # Invalid options will fail in the validation section  
  
 # Subscription Access Details  
 $subID = "xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"  
  
 # VM Account, Location, and Storage Details  
 $LocalAdmin = "theAdmin"  
 $DeploymentLocation = "China North"  
 $StorageAccountName = "vmstore02"  
  
 # Service Details  
 $SecureService = "SecSvc001"  
 $FrontEndService = "FrontEnd001"  
 $BackEndService = "BackEnd001"  
  
 # Network Details  
 $VNetName = "CorpNetwork"  
 $VNetPrefix = "10.0.0.0/16"  
 $SecNet = "SecNet"  
 $FESubnet = "FrontEnd"  
 $FEPrefix = "10.0.1.0/24"  
 $BESubnet = "BackEnd"  
 $BEPrefix = "10.0.2.0/24"  
 $NetworkConfigFile = "C:\Scripts\NetworkConf3.xml"  
  
 # VM Base Disk Image Details  
 $SrvImg = Get-AzureVMImage | Where {$\_.ImageFamily -match 'Windows Server 2012 R2 Datacenter'} | sort PublishedDate -Descending | Select ImageName -First 1 | ForEach {$\_.ImageName}  
 $FWImg = Get-AzureVMImage | Where {$\_.ImageFamily -match 'Barracuda NG Firewall'} | sort PublishedDate -Descending | Select ImageName -First 1 | ForEach {$\_.ImageName}  
  
 # UDR Details  
 $FERouteTableName = "FrontEndSubnetRouteTable"  
 $BERouteTableName = "BackEndSubnetRouteTable"  
  
 # NSG Details  
 $NSGName = "MyVNetSG"  
  
# User Defined VM Specific Config  
 # Note: To ensure UDR and IP forwarding is setup  
 # properly this script requires VM 0 be the NVA.  
  
 # VM 0 - The Network Virtual Appliance (NVA)  
 $VMName += "myFirewall"  
 $ServiceName += $SecureService  
 $VMFamily += "Firewall"  
 $img += $FWImg  
 $size += "Small"  
 $SubnetName += $SecNet  
 $VMIP += "10.0.0.4"  
  
 # VM 1 - The Web Server  
 $VMName += "IIS01"  
 $ServiceName += $FrontEndService  
 $VMFamily += "Windows"  
 $img += $SrvImg  
 $size += "Standard\_D3"  
 $SubnetName += $FESubnet  
 $VMIP += "10.0.1.4"  
  
 # VM 2 - The First Appliaction Server  
 $VMName += "AppVM01"  
 $ServiceName += $BackEndService  
 $VMFamily += "Windows"  
 $img += $SrvImg  
 $size += "Standard\_D3"  
 $SubnetName += $BESubnet  
 $VMIP += "10.0.2.5"  
  
 # VM 3 - The Second Appliaction Server  
 $VMName += "AppVM02"  
 $ServiceName += $BackEndService  
 $VMFamily += "Windows"  
 $img += $SrvImg  
 $size += "Standard\_D3"  
 $SubnetName += $BESubnet  
 $VMIP += "10.0.2.6"  
  
 # VM 4 - The DNS Server  
 $VMName += "DNS01"  
 $ServiceName += $BackEndService  
 $VMFamily += "Windows"  
 $img += $SrvImg  
 $size += "Standard\_D3"  
 $SubnetName += $BESubnet  
 $VMIP += "10.0.2.4"  
  
# ----------------------------- #  
# No User Defined Varibles or #  
# Configuration past this point #  
# ----------------------------- #  
  
 # Get your Azure accounts  
 Add-AzureAccount  
 Set-AzureSubscription –SubscriptionId $subID -ErrorAction Stop  
 Select-AzureSubscription -SubscriptionId $subID -Current -ErrorAction Stop  
  
 # Create Storage Account  
 If (Test-AzureName -Storage -Name $StorageAccountName) {   
 Write-Host "Fatal Error: This storage account name is already in use, please pick a diffrent name." -ForegroundColor Red  
 Return}  
 Else {Write-Host "Creating Storage Account" -ForegroundColor Cyan   
 New-AzureStorageAccount -Location $DeploymentLocation -StorageAccountName $StorageAccountName}  
  
 # Update Subscription Pointer to New Storage Account  
 Write-Host "Updating Subscription Pointer to New Storage Account" -ForegroundColor Cyan   
 Set-AzureSubscription –SubscriptionId $subID -CurrentStorageAccountName $StorageAccountName -ErrorAction Stop  
  
# Validation  
$FatalError = $false  
  
If (-Not (Get-AzureLocation | Where {$\_.DisplayName -eq $DeploymentLocation})) {  
 Write-Host "This Azure Location was not found or available for use" -ForegroundColor Yellow  
 $FatalError = $true}  
  
If (Test-AzureName -Service -Name $SecureService) {   
 Write-Host "The SecureService service name is already in use, please pick a different service name." -ForegroundColor Yellow  
 $FatalError = $true}  
Else { Write-Host "The FrontEndService service name is valid for use." -ForegroundColor Green}  
  
If (Test-AzureName -Service -Name $FrontEndService) {   
 Write-Host "The FrontEndService service name is already in use, please pick a different service name." -ForegroundColor Yellow  
 $FatalError = $true}  
Else { Write-Host "The FrontEndService service name is valid for use" -ForegroundColor Green}  
  
If (Test-AzureName -Service -Name $BackEndService) {   
 Write-Host "The BackEndService service name is already in use, please pick a different service name." -ForegroundColor Yellow  
 $FatalError = $true}  
Else { Write-Host "The BackEndService service name is valid for use." -ForegroundColor Green}  
  
If (-Not (Test-Path $NetworkConfigFile)) {   
 Write-Host 'The network config file was not found, please update the $NetworkConfigFile variable to point to the network config xml file.' -ForegroundColor Yellow  
 $FatalError = $true}  
Else { Write-Host "The network config file was found" -ForegroundColor Green  
 If (-Not (Select-String -Pattern $DeploymentLocation -Path $NetworkConfigFile)) {  
 Write-Host 'The deployment location was not found in the network config file, please check the network config file to ensure the $DeploymentLocation varible is correct and the netowrk config file matches.' -ForegroundColor Yellow  
 $FatalError = $true}  
 Else { Write-Host "The deployment location was found in the network config file." -ForegroundColor Green}}  
  
If ($FatalError) {  
 Write-Host "A fatal error has occured, please see the above messages for more information." -ForegroundColor Red  
 Return}  
Else { Write-Host "Validation passed, now building the environment." -ForegroundColor Green}  
  
# Create VNET  
 Write-Host "Creating VNET" -ForegroundColor Cyan   
 Set-AzureVNetConfig -ConfigurationPath $NetworkConfigFile -ErrorAction Stop  
  
# Create Services  
 Write-Host "Creating Services" -ForegroundColor Cyan  
 New-AzureService -Location $DeploymentLocation -ServiceName $SecureService -ErrorAction Stop  
 New-AzureService -Location $DeploymentLocation -ServiceName $FrontEndService -ErrorAction Stop  
 New-AzureService -Location $DeploymentLocation -ServiceName $BackEndService -ErrorAction Stop  
  
# Build VMs  
 $i=0  
 $VMName | Foreach {  
 Write-Host "Building $($VMName[$i])" -ForegroundColor Cyan  
 If ($VMFamily[$i] -eq "Firewall")   
 {   
 New-AzureVMConfig -Name $VMName[$i] -ImageName $img[$i] –InstanceSize $size[$i] | `  
 Add-AzureProvisioningConfig -Linux -LinuxUser $LocalAdmin -Password $LocalAdminPwd | `  
 Set-AzureSubnet –SubnetNames $SubnetName[$i] | `  
 Set-AzureStaticVNetIP -IPAddress $VMIP[$i] | `  
 New-AzureVM –ServiceName $ServiceName[$i] -VNetName $VNetName -Location $DeploymentLocation  
 # Set up all the EndPoints we'll need once we're up and running  
 # Note: All traffic goes through the firewall, so we'll need to set up all ports here.  
 # Also, the firewall will be redirecting traffic to a new IP and Port in a forwarding  
 # rule, so all of these endpoint have the same public and local port and the firewall  
 # will do the mapping, NATing, and/or redirection as declared in the firewall rules.  
 Add-AzureEndpoint -Name "MgmtPort1" -Protocol tcp -PublicPort 801 -LocalPort 801 -VM (Get-AzureVM -ServiceName $ServiceName[$i] -Name $VMName[$i]) | Update-AzureVM  
 Add-AzureEndpoint -Name "MgmtPort2" -Protocol tcp -PublicPort 807 -LocalPort 807 -VM (Get-AzureVM -ServiceName $ServiceName[$i] -Name $VMName[$i]) | Update-AzureVM  
 Add-AzureEndpoint -Name "HTTP" -Protocol tcp -PublicPort 80 -LocalPort 80 -VM (Get-AzureVM -ServiceName $ServiceName[$i] -Name $VMName[$i]) | Update-AzureVM  
 Add-AzureEndpoint -Name "RDPWeb" -Protocol tcp -PublicPort 8014 -LocalPort 8014 -VM (Get-AzureVM -ServiceName $ServiceName[$i] -Name $VMName[$i]) | Update-AzureVM  
 Add-AzureEndpoint -Name "RDPApp1" -Protocol tcp -PublicPort 8025 -LocalPort 8025 -VM (Get-AzureVM -ServiceName $ServiceName[$i] -Name $VMName[$i]) | Update-AzureVM  
 Add-AzureEndpoint -Name "RDPApp2" -Protocol tcp -PublicPort 8026 -LocalPort 8026 -VM (Get-AzureVM -ServiceName $ServiceName[$i] -Name $VMName[$i]) | Update-AzureVM  
 Add-AzureEndpoint -Name "RDPDNS01" -Protocol tcp -PublicPort 8024 -LocalPort 8024 -VM (Get-AzureVM -ServiceName $ServiceName[$i] -Name $VMName[$i]) | Update-AzureVM  
 # Note: A SSH endpoint is automatically created on port 22 when the appliance is created.  
 }  
 Else  
 {  
 New-AzureVMConfig -Name $VMName[$i] -ImageName $img[$i] –InstanceSize $size[$i] | `  
 Add-AzureProvisioningConfig -Windows -AdminUsername $LocalAdmin -Password $LocalAdminPwd | `  
 Set-AzureSubnet –SubnetNames $SubnetName[$i] | `  
 Set-AzureStaticVNetIP -IPAddress $VMIP[$i] | `  
 Set-AzureVMMicrosoftAntimalwareExtension -AntimalwareConfiguration '{"AntimalwareEnabled" : true}' | `  
 Remove-AzureEndpoint -Name "RemoteDesktop" | `  
 Remove-AzureEndpoint -Name "PowerShell" | `  
 New-AzureVM –ServiceName $ServiceName[$i] -VNetName $VNetName -Location $DeploymentLocation  
 }  
 $i++  
 }  
  
# Configure UDR and IP Forwarding  
 Write-Host "Configuring UDR" -ForegroundColor Cyan  
  
 # Create the Route Tables  
 Write-Host "Creating the Route Tables" -ForegroundColor Cyan   
 New-AzureRouteTable -Name $BERouteTableName `  
 -Location $DeploymentLocation `  
 -Label "Route table for $BESubnet subnet"  
 New-AzureRouteTable -Name $FERouteTableName `  
 -Location $DeploymentLocation `  
 -Label "Route table for $FESubnet subnet"  
  
 # Add Routes to Route Tables  
 Write-Host "Adding Routes to the Route Tables" -ForegroundColor Cyan   
 Get-AzureRouteTable $BERouteTableName `  
 |Set-AzureRoute -RouteName "All traffic to FW" -AddressPrefix 0.0.0.0/0 `  
 -NextHopType VirtualAppliance `  
 -NextHopIpAddress $VMIP[0]  
 Get-AzureRouteTable $BERouteTableName `  
 |Set-AzureRoute -RouteName "Internal traffic to FW" -AddressPrefix $VNetPrefix `  
 -NextHopType VirtualAppliance `  
 -NextHopIpAddress $VMIP[0]  
 Get-AzureRouteTable $BERouteTableName `  
 |Set-AzureRoute -RouteName "Allow Intra-Subnet Traffic" -AddressPrefix $BEPrefix `  
 -NextHopType VNETLocal  
 Get-AzureRouteTable $FERouteTableName `  
 |Set-AzureRoute -RouteName "All traffic to FW" -AddressPrefix 0.0.0.0/0 `  
 -NextHopType VirtualAppliance `  
 -NextHopIpAddress $VMIP[0]  
 Get-AzureRouteTable $FERouteTableName `  
 |Set-AzureRoute -RouteName "Internal traffic to FW" -AddressPrefix $VNetPrefix `  
 -NextHopType VirtualAppliance `  
 -NextHopIpAddress $VMIP[0]  
 Get-AzureRouteTable $FERouteTableName `  
 |Set-AzureRoute -RouteName "Allow Intra-Subnet Traffic" -AddressPrefix $FEPrefix `  
 -NextHopType VNETLocal  
  
 # Assoicate the Route Tables with the Subnets  
 Write-Host "Binding Route Tables to the Subnets" -ForegroundColor Cyan   
 Set-AzureSubnetRouteTable -VirtualNetworkName $VNetName `  
 -SubnetName $BESubnet `  
 -RouteTableName $BERouteTableName  
 Set-AzureSubnetRouteTable -VirtualNetworkName $VNetName `  
 -SubnetName $FESubnet `  
 -RouteTableName $FERouteTableName  
  
 # Enable IP Forwarding on the Virtual Appliance  
 Get-AzureVM -Name $VMName[0] -ServiceName $ServiceName[0] `  
 |Set-AzureIPForwarding -Enable  
  
# Configure NSG  
 Write-Host "Configuring the Network Security Group (NSG)" -ForegroundColor Cyan  
  
 # Build the NSG  
 Write-Host "Building the NSG" -ForegroundColor Cyan  
 New-AzureNetworkSecurityGroup -Name $NSGName -Location $DeploymentLocation -Label "Security group for $VNetName subnets in $DeploymentLocation"  
  
 # Add NSG Rule  
 Write-Host "Writing rules into the NSG" -ForegroundColor Cyan  
 Get-AzureNetworkSecurityGroup -Name $NSGName | Set-AzureNetworkSecurityRule -Name "Isolate the $VNetName VNet from the Internet" -Type Inbound -Priority 100 -Action Deny `  
 -SourceAddressPrefix INTERNET -SourcePortRange '\*' `  
 -DestinationAddressPrefix VIRTUAL\_NETWORK -DestinationPortRange '\*' `  
 -Protocol \*  
  
 # Assign the NSG to two Subnets  
 # The NSG is \*not\* bound to the Security Subnet. The result  
 # is that internet traffic flows only to the Security subnet  
 # since the NSG bound to the Frontend and Backback subnets  
 # will Deny internet traffic to those subnets.  
 Write-Host "Binding the NSG to two subnets" -ForegroundColor Cyan  
 Set-AzureNetworkSecurityGroupToSubnet -Name $NSGName -SubnetName $FESubnet -VirtualNetworkName $VNetName  
 Set-AzureNetworkSecurityGroupToSubnet -Name $NSGName -SubnetName $BESubnet -VirtualNetworkName $VNetName  
  
# Optional Post-script Manual Configuration  
 # Configure Firewall  
 # Install Test Web App (Run Post-Build Script on the IIS Server)  
 # Install Backend resource (Run Post-Build Script on the AppVM01)  
 Write-Host  
 Write-Host "Build Complete!" -ForegroundColor Green  
 Write-Host  
 Write-Host "Optional Post-script Manual Configuration Steps" -ForegroundColor Gray  
 Write-Host " - Configure Firewall" -ForegroundColor Gray  
 Write-Host " - Install Test Web App (Run Post-Build Script on the IIS Server)" -ForegroundColor Gray  
 Write-Host " - Install Backend resource (Run Post-Build Script on the AppVM01)" -ForegroundColor Gray  
 Write-Host

#### Network Config File

Save this xml file with updated location and add the link to this file to the $NetworkConfigFile variable in the script above.

<NetworkConfiguration xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://schemas.microsoft.com/ServiceHosting/2011/07/NetworkConfiguration">  
 <VirtualNetworkConfiguration>  
 <Dns>  
 <DnsServers>  
 <DnsServer name="DNS01" IPAddress="10.0.2.4" />  
 <DnsServer name="Level3" IPAddress="209.244.0.3" />  
 </DnsServers>  
 </Dns>  
 <VirtualNetworkSites>  
 <VirtualNetworkSite name="CorpNetwork" Location="China North">  
 <AddressSpace>  
 <AddressPrefix>10.0.0.0/16</AddressPrefix>  
 </AddressSpace>  
 <Subnets>  
 <Subnet name="SecNet">  
 <AddressPrefix>10.0.0.0/24</AddressPrefix>  
 </Subnet>  
 <Subnet name="FrontEnd">  
 <AddressPrefix>10.0.1.0/24</AddressPrefix>  
 </Subnet>  
 <Subnet name="BackEnd">  
 <AddressPrefix>10.0.2.0/24</AddressPrefix>  
 </Subnet>  
 </Subnets>  
 <DnsServersRef>  
 <DnsServerRef name="DNS01" />  
 <DnsServerRef name="Level3" />  
 </DnsServersRef>  
 </VirtualNetworkSite>  
 </VirtualNetworkSites>  
 </VirtualNetworkConfiguration>  
</NetworkConfiguration>

#### Sample Application Scripts

If you wish to install a sample application for this, and other DMZ Examples, one has been provided at the following link: [Sample Application Script](./virtual-networks-sample-app.md)