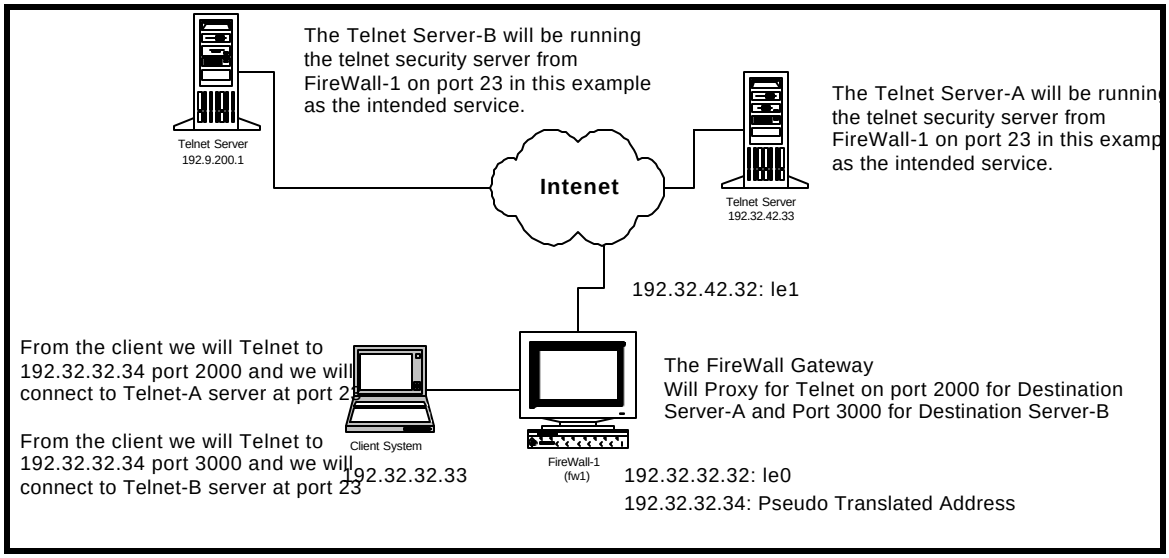
Check Point Software Technologies LTD.

Creating A Generic Service Proxy (GSP) Using Network Address Translation (NAT)

This document will outline the basic procedure for achieving the equivalent of a Generic Service Proxy (GSP) for any service or port at the IP level using IP Address Translation. The goal is to be able to connect to the FireWall for a given service and port on the inside, and have that connection forwarded or proxied directly to the target host on the outside. The net gain here is that the client host need not know from a routing perspective how to get to the target host, just to the FireWall service as a generic Proxy.

First lets outline our objects:

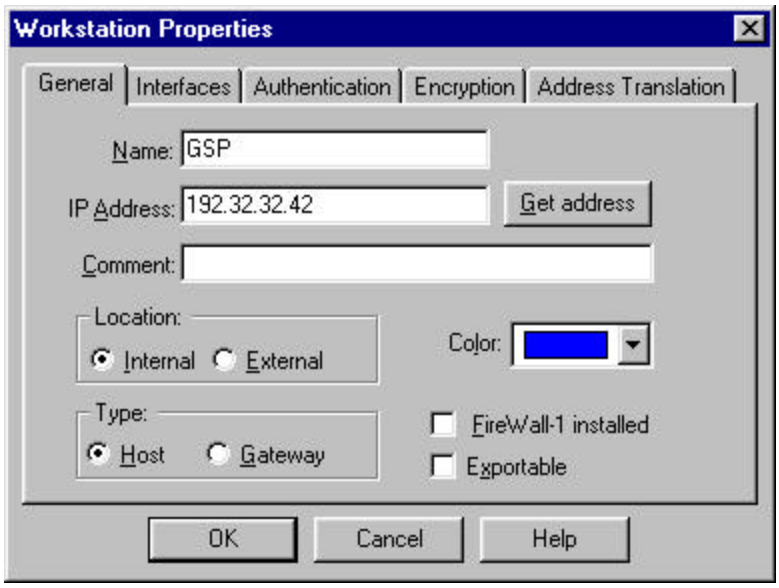
- fw1: This is out Getaway object.
- GSP: This is the pseudo address that the FireWall will publish Internally
- GenServ-A: This is a Generic TCP Service that we created on port 2000.
- GenServ-B: This is a Generic TCP Service that we created on port 3000.
- Local-Net: This is our internal network object.
- Target-A: This is Server-A that we will connect to using port 2000 in this example.
- Target-B: This is Server-B that we will connect to using port 3000 in this example.



Given the above diagram we will configure address translation on the FireWall server to proxy a Telnet session on port 2000 and 3000 to the Telnet Server target hosts. We will use a common internal IP address but will translate the different service port numbers to determine which one of the target systems we connect to. The client system need not have a route to the target hosts. The goal is to connect to the FireWall at port 2000 or 3000, and communicate directly to the Target hosts at port 23 for Telnet.

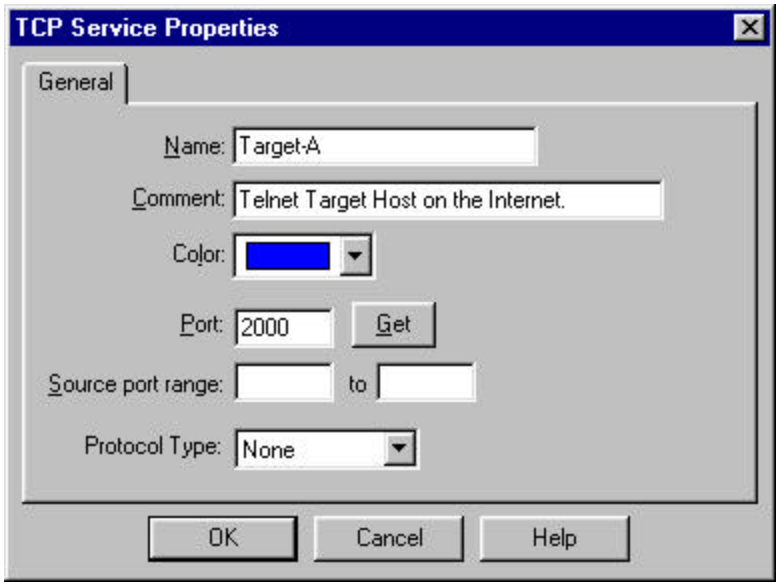
First we will create all of our test objects using the FireWall-1 GUI Interface. We must create the basic Workstation and Network objects for our FireWall, but we must also create 2 generic services for the port based addressing to work properly. We must also create objects for the target systems that we intend to connect to using the NAT based proxy. This approach has obvious limitations here in that we have the ability to do a “Many To One” translation , not a “One To Many” or “Many To Many”. What I mean is to accomplish the GSP functionality we are modifying the SRC, DST, and PORT information on the packet to define the target server. So, we must know the address of the target server or servers in advance, and place a NAT rule for each destination.

Object Configuration:



We must create an object that represents the pseudo address that our users will connect to on the inside.

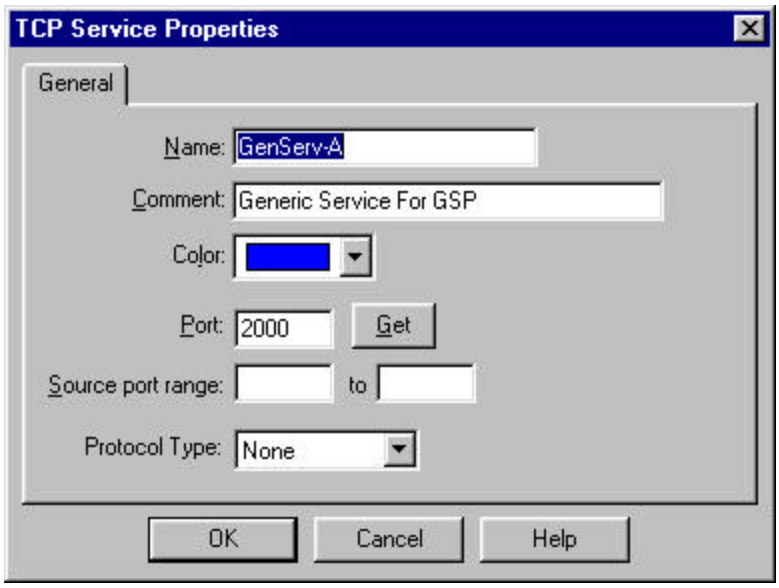
This object is called GSP.



We must also create objects for our target hosts on the Internet.

This object is called Target-A.

We will create a Target-B as well.

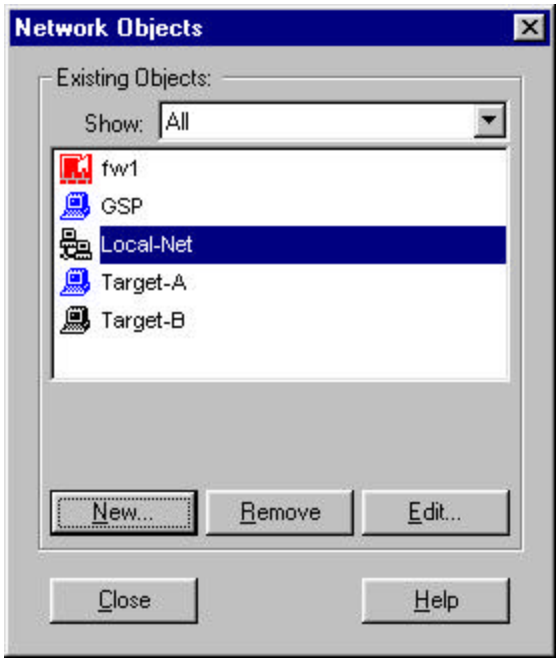


Next we must create generic services to define the port numbers that users will use on the inside to connect to the corresponding server on the Internet.

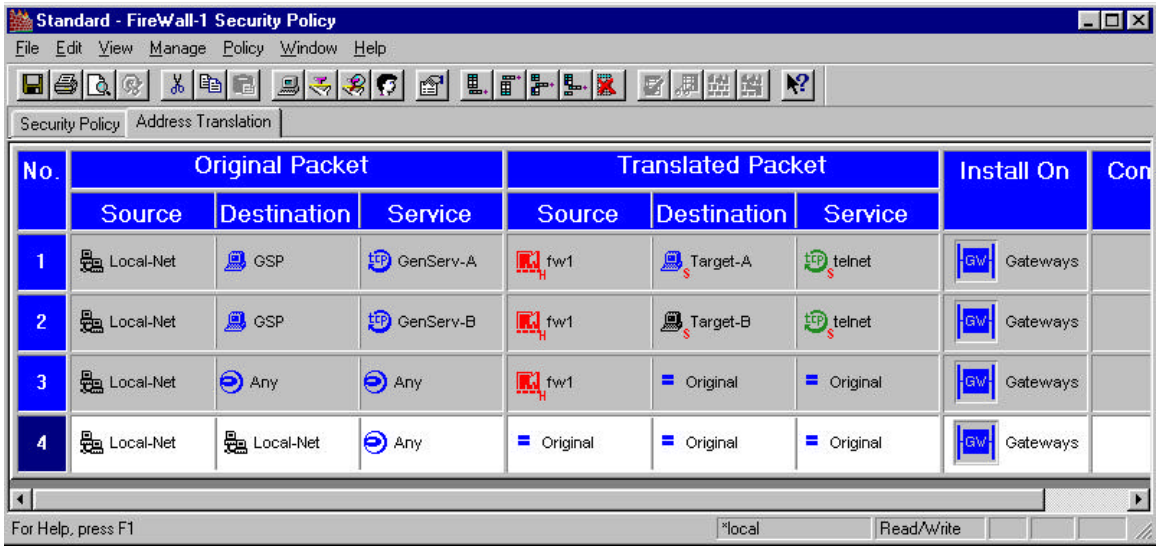
This is our Service object that will facilitate the PORT translation for the user. The user will Telnet to the pseudo address on the inside of the FireWall to port 2000, then the FireWall will use the PORT number to determine the destination target host on the Internet.

We will create a service for GenServ-A and GenServ-B.

Our completed Object Base:



Our Completed Services:



Now we are ready to build out NAT Rule Base.

Notice that we have a Single Rule for each destination. This is the limitation. We are translating SRC, DST, and PORT to effectively proxy this Telnet session from the Internal Network to the 2 Telnet servers on the Internet. In order for this to work we must have a Rule per destination, and we must know the destinations in advance.

NOTE: As per any static NAT configuration we must create a proxy arp for the 192.32.32.34 pseudo address that is being published on the inside of the gateway.

To test this we can remove the static route from the internal client system so that we only have IP connectivity to the internal interface of the gateway. To reach Target-A we can type the following:

Telnet 192.32.32.34 2000

We will end up connected to the Target-A server on the Internet. This may be useful in some circumstances where we are trying to offset some of the features like Generic Service Proxy offered by our competitors.