Firewall Port Handling in TENA Applications

The purpose of this report is to describe the manner in which TENA applications handle communications using TCP. This report will also present some insight for firewall administrators so that they can understand what is needed for TENA applications to run behind firewalls.

For the reliable communication mode, which is the TENA default mode, TENA uses TCP connections to handle communication between applications. These connections are bidirectional which means the data is sent and received in both directions. For this reason, the port numbers used for forming the TCP connection are important from a network/firewall administration point of view.

Ports are used as doors on a system where data can be sent out and/or received. These ports are logical entities defined by the TCP/IP stack software. Each TCP packet contains a source port and a destination port within the header of the packet along with other information needed for the reliable connection. A TCP based server application (which is what TENA publishing applications are) will 'listen' on a certain port for packets from a client trying to connect. An example of this is with a web server. A client would request a connection on port 80 (HTTP port) of the web server. The server hears this request because it is listening on port 80 for incoming TCP packets. The web server would then send a TCP acknowledgement back to the client system on the client source port to begin the process of the connection oriented communication.

As was said before, TENA uses TCP for its reliable communication mode. Some of the ports that are used during communication can be specified in the middleware. Any publishing application (or the NNS and EMS) can specify on which port the application is going to 'listen' in order to form connections with other TENA applications, just as described above with the HTTP example. This port is specified by the ORBlistenEndpoints command. This port would then be assigned to the destination port number by any application that is trying to subscribe to the published SDO. The subscribing application does not need to know this port ahead of time. It is assumed that this information is obtained from the execution manager at the time of execution. The source port that the subscribing application uses for the TCP connection is assigned by the OS, according to the TENA Middleware FAQ document. This source port is a randomly assigned port that is in the range of 1024 or above.

When a connection request (TCP SYN) is sent to the publisher listening port (specified by ORBlistenEndpoints) by an external TENA subscription application, a TCP acknowledgement (TCP ACK) is sent by the publishing system back to the external subscribing system that requested the connection. The port number to which the TCP ACK is sent to is the source port from the TCP SYN message. This source port is the dynamically assigned port described earlier. To explain this better here is a scenario that depicts this situation.

Here are the machines involved in the scenario and the commands for the applications that are running on each machine:

Machine 1: Machine 1 runs the network naming service using the command:
networkNamingService –ORBlistenEndpoints iiop://<ipaddress1>:5000

Machine 2: Machine 2 runs the execution manger using the command:
executionManager –ORBdefaultInitRef corbaloc:iiop:<ipaddress1>:5000 –ORBlistenEndpoints iiop://<ipaddress2>:6000 –executionName test

Machine 3: Machine 3 runs a publisher using the command:
publish –ORBdefaultInitRef corbaloc:iiop:<ipaddress1>:5000 –ORBlistenEndpoints iiop://<ipaddress3>:7000 –executionName test

The following figure shows the connections that are setup from running these three applications.

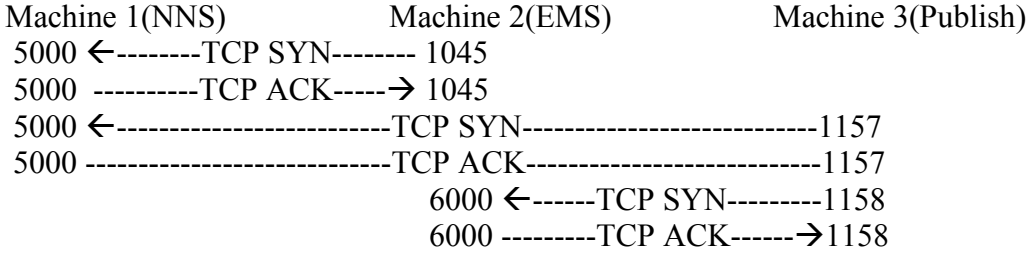

Figure 1: Connections established from scenario applications

The ports numbers that are shown here are a combination of the statically assigned ports (specified by the ORBlistenEndpoints command) and the dynamically assigned ports (assigned by the OS). The two connections with Machine 1 were formed by the other applications requesting a connection on port 5000 of Machine 1. As can be seen in the Machine 1 command above, port 5000 was specified as being the port on which this application should listen by the use of the ORBlistenEndpoints command. This is the port that the external machines send TCP SYN messages to begin communication with Machine 1. The acknowledgements were sent back to the requesting application at the source port number that was used by the external machine. These port numbers are not the port numbers that are assigned by the ORBlistenEndpoints command in the external machine but are the dynamic port numbers that are assigned at the beginning of the communication by the OS. So in the case of machine 2, it was sent to port 1045 because that was the source port included in the TCP SYN message. In the same manner, Machine 3 forms a connection with Machine 2 on port 6000, which was the static port that was assigned using ORBlistenEndpoints. The port used by Machine 3 to form the connection was a source port dynamically assigned by the OS, port 1158.

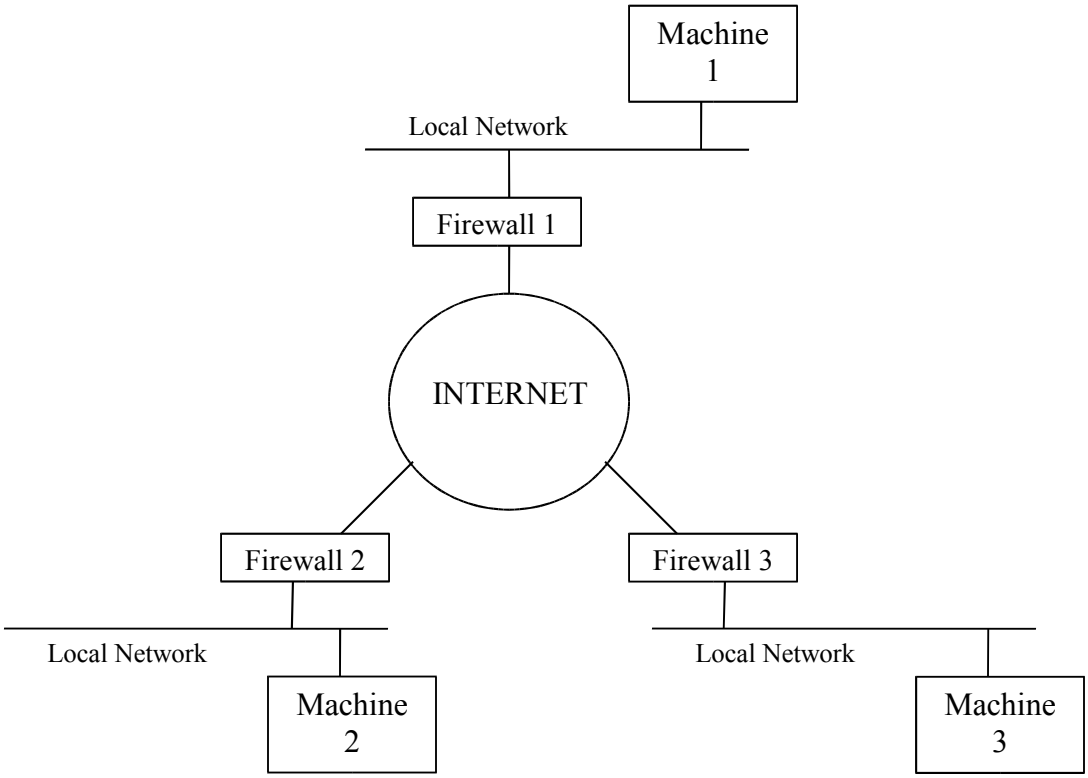

Figure 2: Network diagram of scenario network with firewalls

If there was a firewall in front of each machine (as shown in figure 2) then the firewall would need to open the respective ports in order for these two machines to communicate. Working from the previous scenario, the first issue would be the TCP connection request being sent to Machine 1 port 5000. For this connection to be established, port 5000 for the server machine (Machine 1) would need to be opened in Firewall 1. This should be handled by the Firewall 1 administrator. Firewall 1 needs to be configured for the internal machine to receive TCP connections at port 5000, which was the statically assigned port specified in the ORBlistenEndpoints command. The next step is the TCP acknowledgement. Since the port of Machine 2 was dynamically assigned by the OS, the Firewall 2 administrator cannot open this port dynamically to allow this connection. According to Steve Bachinsky with the FI2010 helpdesk, "Firewalls are set up to know that if a machine inside the firewall establishes a connection to an external machine, that socket connection will allow the remote machine to communicate back to the internal machine. It is the same TCP socket connection that is used for these TCP ACK messages." (TENA-243) This means that the firewall is smart enough to understand that the connection being established with the TCP acknowledgement message was produced from a TCP SYN message originating from within the local network so the message will be sent through to the requesting machine, Machine 2. This is the same for all of the connections that are made in this scenario. As can be seen in figure 1, every TCP SYN message is sent to a port that was specified by the ORBlistenEndpoints command line

parameter. This allows for the firewall administrators to open up this port to allow incoming connections. Beyond this, the firewalls should be smart enough to recognize the incoming TCP ACK message as an acknowledgement to a TCP SYN message that was sent from an internal machine.

So what does this mean for the DTE ARCH project? The suggestion would be that each application has a set port number to include in an ORBlistenEndpoints command line parameters for any publishing applications. These port numbers would be statically assigned and would be given to the firewall administrator ahead of time so that he/she may open that port for the internal TENA application server. If what is described above is correct, then this should take care of any inconsistencies in the connections. The reason for this is that each application that needs to form a connection with a published SDO will have the port number for that SDO application since it will be known by the TENA middleware. This port number will be previously opened by the firewall administrator so that incoming connections can be established. The firewalls will then be smart enough to understand that the TCP acknowledgements going to randomly assigned ports are okay to let through since the connection originated within the local network.

A couple of things need to happen before this can be guaranteed to work The firewall administrators at each location needs to verify this behavior to ensure that this is not a policy restricted behavior that has been set by administration. The firewall administrator will also need to know the following information for each internal machine running TENA applications (including the machine running the NNS and EMS): Internal Machine Name/IP Address and ORBlistenEndpoints Port Number(s) for the application(s) running on that internal machine. In some cases, the firewall administrator may want to know the IP addresses of all of the remote machines that may attempt to connect to the internal machine (through the ORBlistenEndpoints port). This allows them to be slightly more restrictive and avoid opening up the firewall for that port for all machines on the WAN. DTE-ARCH will coordinate with firewall administrators in providing IP addresses and port addresses that will be used in the exercise.

The next two sections discuss some other options and features of the ORBlistenEndpoints command that have been presented as solutions to the firewall issues. A description of the option and why it will or will not help for Milestone 4.0 is given.

Port Span Command

The port span command is added to the ORBlistenEndpoints command in order to give the TENA application a range of port numbers in which the application can use to listen. The command looks like

networkNamingService –ORBlistenEndpoints iiop://<ipaddress>:5000/portspan=5

In the situation where the machine on which the TENA application is running has more than one application running using various ports, the port selection process can be complicated. This command would allow for a range of ports that the TENA application can use to listen for connections, which in this case the range is 5 (5000, 5001, …, 5004). The application will try the first port, port 5000, and if that one is unavailable it will try 5001. This will continue until it either finds an open port or reaches the end of the list, which in this case is 5004. This command is useful in that it would allow for the application to connect to the execution and prevent the situation where an error would be thrown because the port that was specified in the ORBlistenEndpoints command was currently in use. Since the application may use any of the ports in the port span, each port would then need to be manually opened in the firewall for incoming connections.

This feature of the ORBlistenEndpoints command could be useful in a situation where one server would be running multiple applications and a preconfigured capability is desired. This command will allow for these applications to be run off of a block of set aside ports so that each one can have a port to listen on without the user having to manual find an available port. This is an ability that may be desired now and in the future. If this ability is not necessary or the application is only going to be run a small number of times, one answer would be to make sure that the assigned port number for each application is not in use on the system in which the TENA application will be running. If the port number is not in use then the application should have no problem starting and can listen on the single endpoint specified in the ORBlistenEndpoints command. Whether port span is used or not, does not make a difference for firewall configuration except for the fact that the firewall administrator would need to know the entire range of port numbers that the application could use.

Firewall Detection

The firewall detection feature, which is an enhancement included in release 4.0.4.1, is a feature that allows applications to know when communications have not been established between it and the execution manager. The middleware has been changed so that the execution manager displays an error message in its console window if an application is unable to receive network traffic from the execution manager. An error message is also displayed in the applications console window. These error messages could be helpful in the situation where an application is unable to join the connection, but it does nothing to repair the problem. These error messages are only meant to be a diagnostic tool. This enhancement also seems to only address the communication between an application and the execution manager and not the communication between applications. For this reason the tool may not be completely helpful and a more permanent solution should be used. While updating to release 4.0.4.1 for the diagnostic tool could be helpful, it may be more trouble than it is worth for Milestone 4.0. The firewall settings discussed earlier should side step this issue and allow for the systems to communicate through firewalls.