Cogent Real-Time Systems Inc

OPC Tunnelling – Know Your Options

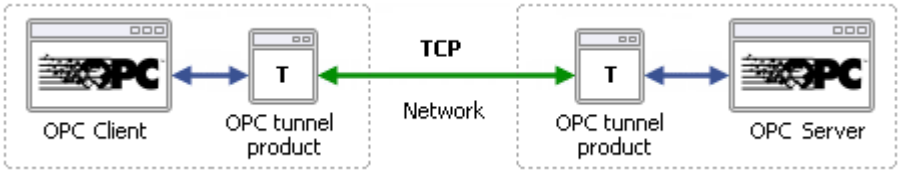
by Bob McIlvride and Andrew Thomas
Cogent Real-Time Systems

Since OPC was introduced over ten years ago, it has seen a steady rise in popularity within the process control industry. Using OPC, automation professionals can now select from a wide range of client applications to connect to their PLCs and hardware devices. The freedom to choose the most suitable OPC client application for the job has created an interest in drawing data from more places in the plant. Industry-wide, we are seeing a growing need to connect OPC clients on one computer to OPC servers on other, networked computers. As OPC has grown, so has the need to network OPC.

At the same time, anyone who has attempted to network OPC knows that it is challenging, at best. The networking protocol for OPC is DCOM, which was not designed for real-time data transfer. DCOM is difficult to configure, responds poorly to network breaks, and has serious security flaws. Using DCOM between different LANs, such as connecting between manufacturing and corporate LANs, is sometimes impossible to configure. Using OPC over DCOM also requires more network traffic than some networks can handle because of bandwidth limitations, or due to the high traffic already on the system. To overcome these limitations, there are various "tunnelling" solutions on the market. This article will look at how tunnelling solves the issues associated with DCOM, and show you what to look for in an OPC tunnelling product.

Eliminating DCOM

The goal of OPC tunnelling is to eliminate DCOM, which is commonly done by replacing the DCOM networking protocol with TCP. Instead of connecting the OPC client to a networked OPC server, the client program connects to a local OPC tunnelling application, which acts as a local OPC server. The tunnelling application accepts requests from the OPC client and converts them to TCP messages, which are then sent across the network to a companion tunnelling application on the OPC server computer. There the request is converted back to OPC and is sent to the OPC server application for processing. Any response from the server is sent back across the tunnel to the OPC client application in the same manner.

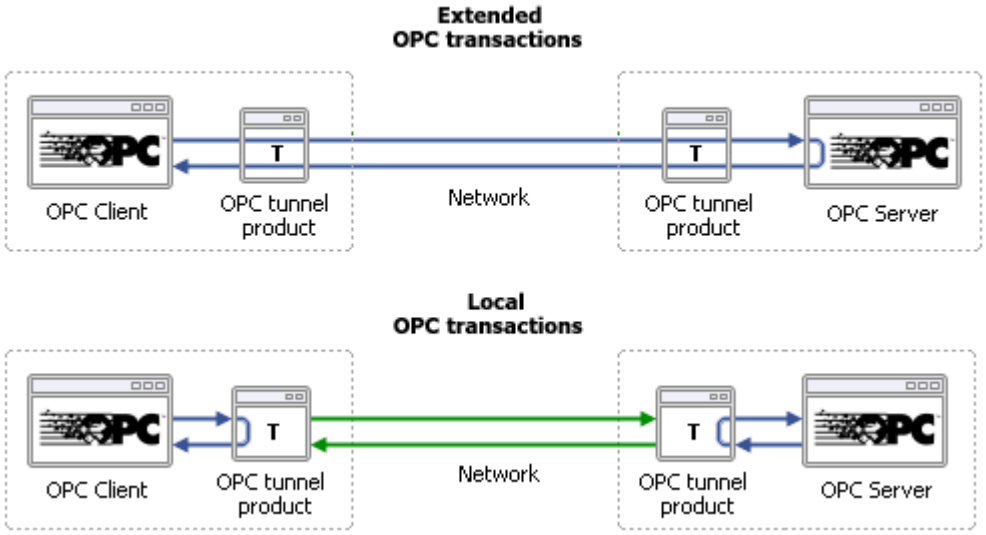


This is how most OPC tunnellers work, in principle. A closer look will show us that although all of them eliminate DCOM, there are some fundamentally different approaches to OPC tunnelling architecture that lead to distinctly different results in practice. As you review tunnelling solutions, here are four things to look out for:

1. Does the tunnelling product extend OPC transactions across the network, or does it keep all OPC transactions local?
2. What happens to the OPC client and server during a network break?
3. How does the tunnel support multiple client-server connections?
4. Does the tunnelling product provide security, including data encryption, user authentication, and authorization?

1. Extended or Local OPC Transactions?

There are two basic types of OPC tunnelling products on the market today, each with a different approach to the problem. The first approach extends the OPC transaction across the network link, while the second approach keeps all OPC transactions local to the sending or receiving computer.



Extending the OPC transaction across the network means that a typical OPC client request is passed across the network to the OPC server, and the server's response is then passed all the way back to the client. Unfortunately, this approach preserves the synchronous nature of DCOM over the link, with all of its negative effects. It exposes every OPC client-server transaction to network issues like timeouts, delays, and blocking behaviour. Link monitoring can reduce these effects, but it doesn't eliminate them, as we shall see below.

On the other hand, the local OPC transaction approach limits the client and server OPC transactions to their respective local machines. For example, when the OPC tunnelling program receives an OPC client request, it responds immediately to the OPC client with data from a locally cached copy. At the other end, the same thing happens. The tunnelling program's job is then to maintain the two copies of the data (client side and server side) in constant synchronization. This can be done very efficiently without interfering with the function of the client and server. The result is that the data crosses the network as little as possible, and both OPC server and OPC client are protected from all network irregularities.

2. Handling Network Issues

There is a huge variety of network speeds and capabilities, ranging from robust LANs, to WANs running over T1 lines on multi-node internets, and on down to low-throughput satellite connections. The best tunnelling products give the best possible performance over any given kind of network.

To protect against network irregularities and breaks, any good tunnelling application will offer some kind of link monitoring. Typically this done with a "heartbeat" message, where the two tunnel programs send messages to one another on a timed interval, for example every few seconds. If a reply isn't received back within a user-specified time, the tunnelling application assumes that the network is down. The OPC client and server may then be informed that the network is broken.

In practice this sounds simple. The problem arises when you have to specify the timeout used to identify a network disconnection. If you set the timeout too long, the client may block for a long time waiting for a reply, only to discover that the network is down. On the other hand, setting the timeout too short will give you false indications of a network failure if for some reason the connection latency exceeds your expectations. The slower the network, the greater the timeout must be.

However, this balancing act is only necessary if the tunnelling product uses the extended OPC approach. A product that offers local OPC transactions still provides link monitoring, but the OPC client and server are decoupled from the network failure detection. Consequently, the timeout can be set appropriately for the network characteristics—from a few hundred milliseconds for highly robust networks to many seconds, even minutes for extremely slow networks—without the risk of blocking the OPC client or server.

How the tunnelling product informs your OPC client of the network break also varies according to the tunnel product design. Products that extend the OPC transactions generally do one of two things:

1. Synthesize an OPC server shutdown. The OPC client receives a shutdown message that appears to be coming from the server. Unaware of the network failure, the client instead operates under the assumption that the OPC server itself has stopped functioning.
2. Tell the client nothing, and generate a COM failure the next time the client initiates a transaction. This has two drawbacks. First the client must be able to deal with COM failures, the most likely event to crash a client. Worse yet, since OPC clients often operate in a "wait" state without initiating transactions, the client may think the last data values are valid and up-to-date, never realizing that there is any problem.

Products that provide local OPC transactions offer a third option:

3. Maintain the COM connection throughout the network failure, and alter the quality of the data items to "Not Connected" or something similar. This approach keeps the OPC connection open in a simple and robust way, and the client doesn't have to handle COM disconnects.

3. Support for Multiple Connections

Every tunnelling connection has an associated cost in network load. Tunnelling products that extend OPC transactions across the network may allow many clients to connect through the same tunnel, but each client sends and receives data independently. For each connected client the network bandwidth usage increases. Tunnelling products that satisfy OPC transactions locally can handle any number of clients and servers on either end of the tunnel, and the data flows across the network only once. Consequently, adding clients to the system will not add load to the network. In a resource-constrained system, this can be a crucial factor in the success of the control application.

If you are considering multiple tunnelling connections, be sure to test for cross-coupling between clients. Does a time-intensive request from a slow client block other requests from being handled? Some tunnelling applications serialize access to the OPC server when multiple clients are connected, handling the requests one by one. This may simplify the tunnel vendor's code, but it can produce unacceptable application behavior. If one client makes a time-consuming request via the tunnel, then other clients must line up and wait until that request completes before their own requests will be serviced. All clients block for the duration of the longest request by any client, reducing system performance and increasing latency dramatically.

On the other hand, if the tunnel satisfies OPC requests locally, this situation simply does not happen. The OPC transactions do not cross the network, so they are not subject to network effects nor to serialization across the tunnel.

4. What About Security?

Whenever you get involved in networking plant data, security is a key concern. In fact, security is a primary reason for choosing tunnelling over DCOM. DCOM was never intended for use over a wide area network, so its security model is primarily designed to be easily configured only on a centrally administered LAN. Even making DCOM security work between two different segments of the same LAN can be extremely difficult. One approach to DCOM security is to firewall the whole system, so that nothing gets in or out, then relax the security settings on the computers inside the firewall. This is perhaps the best solution on a trusted network, but it is not always an option. Sometimes you have to transmit data out through the firewall to send your data across a WAN or even the Internet. In those cases, you are going to want a secure connection. Relaxed DCOM settings are simply not acceptable.

Most experts agree that there are three aspects to network security:

- **Data encryption** is necessary to prevent anyone who is sniffing around on the network from reading your raw data.
- **User authentication** validates each connecting user, based on their user name and password, or some other shared secret such as a private/public key pair.
- **Authorization** establishes permissions for each of those authenticated users, and gives access to the appropriate functionality.

There are several options open to tunnelling vendors to provide these three types of security. Some choose to develop their own security solution from the ground up. Others use standard products or protocols that many users are familiar with. These include:

SSL (Secure Socket Layer) - Provides data encryption only, but is very convenient for the user. Typically, you just check a box in the product to activate SSL data encryption. The tunnelling product must provide user authentication and authorization separately.

VPN (Virtual Private Network) - Provides both encryption and authentication. VPN does not come as part of the product, per se, but instead is implemented by the operating system. The tunnelling product then runs over the VPN, but still needs to handle authorization itself.

SSH (Secure Shell) **Tunnelling** - Provides encryption and authentication to a TCP connection. This protocol is more widely used in Unix and Linux applications, but can be effective in MS-Windows. SSH Tunnelling can be thought of as a kind of point-to-point VPN.

As none of these standard protocols covers all the three areas, you should ensure that the tunnelling product you chose fills in the missing pieces. For example, don't overlook authorization. The last thing you need is for some enterprising young apprentice or intern to inadvertently link in to your live, production system and start tweaking data items.

How Can You Know? Test!

The concept of OPC tunnelling is still new to many of us. Vendors of OPC tunnelling products spend a good deal of time and energy just getting the basic point across: eliminate the hassles of DCOM by using TCP across the network. Less attention has been put on the products themselves, and their design. As we have seen, though, these details can mean all the difference between a robust, secure connection, or something significantly less.

How can you know what you are getting? Gather as much information as you can from the vendor, and then test the system. Download and install a few likely products. (Most offer a time-limited demo.) As much as possible, replicate your intended production system. Put a heavy load on it. Pull out a network cable and see what happens. Connect multiple clients, if that's what you plan to do. Configure the security. Also consider other factors such as ease of use, OPC compliance, and how the software works with other OPC-related tasks you need to do.

If you are fed up with DCOM, OPC tunnelling provides a very good alternative. It is a handy option that any engineer or system integrator should be aware of. At the very least, you should certainly find it an improvement over configuring DCOM. And with the proper tools and approach, you can also make it as robust and secure as your network will possibly allow.

About the Authors

Bob McIlvride is the Communications Manager for Cogent Real-Time Systems. He has a Master's degree in Professional Writing, and 12 years experience writing and publishing technical documentation and marketing communications in the natural gas and process control industries.

Andrew Thomas is the President and co-founder of Cogent Real-Time Systems. He has a Master's degree in Systems Design Engineering and Artificial Intelligence. For more than 15 years he has been working in the process control industry, focusing on data communication, interoperability, and the application of AI in process control.

About the Company

Founded in 1995, Cogent Real-Time Systems is the leader in real-time data integration between Windows, Linux and QNX systems. Customers include the Bank of Canada, Cadbury Chocolate and the European Space Agency. Cogent leverages its experience in real-time data communications to provide the next generation of OPC products. For more information, please contact Cogent at info@cogent.ca or visit our web site at www.opcdatahub.com. You can also call us at +1 (905) 702 7851.