GaggleBridge: collaborative data analysis

Florian Battke*,†, Stephan Symons†, Alexander Herbig and Kay Nieselt

Integrative Transcriptomics, Center for Bioinformatics, University of Tuebingen, 72076 Tuebingen, Germany

Received on May 10, 2011; revised on June 28, 2011; accepted on July 14, 2011

1 INTRODUCTION

The Gaggle (Shannon et al., 2006) has become the standard framework for connecting different applications together to study systems biology questions more efficiently. Many different applications support Gaggle, i.e., they are implementing the Gaggle Goose interface. Data such as gene identifiers, clusters, regulatory associations, and network structures can be exchanged between all such Geese that are connected to a common Gaggle Boss instance. This can greatly speed up scientific work by removing the need for data format conversion and exporting/importing data between different programs. This communication is usually limited to programs running on the local computer.

On the other hand, the Gaggle approach could also be used to allow different researchers to work together on one dataset. For example, data could be sent from one user’s genome browser to another user’s microarray analysis software, enabling the collaborative dissection of complex datasets. The RMI methodology underlying Gaggle’s implementation is network transparent and allows a connectivity between programs over long distances. However, very few Geese support connections to remote Boss instances. Furthermore, connecting to remote instances is not always straightforward as these might be behind firewalls or security gateways. Exposing them to the world-wide network is undesirable as Gaggle does not include any security or authentication scheme.

Here we present GaggleBridge, an addition to the Gaggle that extends the existing concept to allow collaborative research over the network.

2 METHODS

We propose to increase the scope of Gaggle towards collaborative research, even over long distances, by connecting different Boss instances over the network. We include the highly secure SSH protocol to deal with the problems of authentication and encryption as well as to circumvent problems due to restrictive firewalls.

For this purpose, we implemented GaggleBridge, a program that allows the user to connect multiple Gaggle Boss instances, thus creating a sort of ‘Super-Boss’ instance (Fig. 1A). The elegance of this approach is that it requires neither modification of the existing Gaggle Boss implementation nor of any Goose. Instead, GaggleBridge represents itself as just another Goose to each of the connected Bosses, transferring data between them.

Connected Boss instances can be activated or inactivated, similar to the possibility of activating and deactivating connected Geese in the Boss main window. However, while inactivated Geese can still send data to their Boss, deactivation in GaggleBridge completely disconnects the respective Boss such that it can neither send to nor receive from other Boss instances connected to the GaggleBridge.

Since many academic users are behind restrictive firewalls, and also do not want unauthorized users to connect to their Gaggle Boss, we decided to include SSH tunneling in GaggleBridge. Thus, a group of researchers can collaborate as long as at least one of their computers is reachable from the outside via SSH login (as in Fig. 2C). Many research facilities permit SSH traffic to pass their firewalls, effectively enabling users to connect otherwise secluded Gaggle Boss instances together. Our implementation is flexible enough to support many different network configurations (see Fig. 2 for some examples), allowing users in extremely restrictive networks, or behind NAT (network address translation) routers to participate in the collaborative data analysis. Since all participating Gaggle Boss instances are equal peers (being both servers and clients at the same time), a dedicated machine allowing users to login via SSH can be used to set up a Gaggle network in a star topology. Thus, only one computer needs to be set up to allow SSH logins and administration can be centralized.

The SSH protocol encrypts all data before transmission, using a very secure encryption method. Data communicated between the Gaggle users thus is protected from eavesdropping. Furthermore, since each user needs SSH login credentials (username and password) to connect, this adds an admittedly simple form of user management to the Gaggle. We have added SSH tunneling capabilities to GaggleBridge based on the JSch implementation (http://www.jcraft.com/jsch/, licensed under a BSD-style open-source license) of the SSH2 protocol.

Furthermore, we developed an extended version of the Gaggle Boss that informs the user about Goose that try to connect and allows the user to refuse unwanted Goose before they can establish a connection. If a Goose is denied, it will neither receive any objects transmitted nor will it be able to send data to other Goose on this Boss. Both the extended Boss and GaggleBridge can be used in one single application. Alternatively, GaggleBridge can be used as a stand-alone program.
The availability of automated network communication, which is the right. this broadcast. The loci are highlighted and added to list of bookmarks on of two users (Lucía and Alex) working in two different labs on the solution for this issue. Currently, we cannot propose a technical itself. The second problem is the general problem of trust between issue should be addressed in a possible later release of the somewhat remedies the first problem, we believe that this security (Genome Browser instance (user Lucía) broadcasting a selection of three loci; was connected by one computer running both a Boss as well as GaggleBridge (circle); (B) one researcher uses GaggleBridge to connect his local Boss to a network of Gaggle users behind a restrictive firewall (rounded box) using an SSH tunnel to a computer with SSH enabled (hashed box); (C) two groups of users in different networks with restrictive firewalls are connected using one GaggleBridge for each network and one SSH tunnel between the networks. same project. Both use the Gaggle Genome Browser (Bare et al., 2010) to visualize locus data, in addition they have a conversation via a telephone conference. Their Gaggle Boss instances are connected via GaggleBridge. Now Lucía can select specific loci in the browser and broadcast the selection with Gaggle (Fig. 1B). Alex’ genome browser receives this information, the same elements are instantly selected and the loci are added to the list of bookmarks (Fig. 1C). This increases the efficiency of the communication between the users substantially as a verbal description of the locus by Lucía and a tedious manual search for the elements by Alex is no longer required. Such communication can also take place within a much larger group of users, and each user can run additional Geese, which will all receive the broadcasts processing and visualizing the data according to their specific functionality. Note that, as GaggleBridge transparently enhances Gaggle, no modification of these Geese or the connected Boss instances are required. Thus, the features of GaggleBridge can be used with minimal installation effort, allowing to swiftly introduce a collaborative approach to data analysis into any research environment.

### ACKNOWLEDGEMENT
The authors are grateful to Lucía Spangenberg-Torre for providing a transatlantic test environment.

### REFERENCES