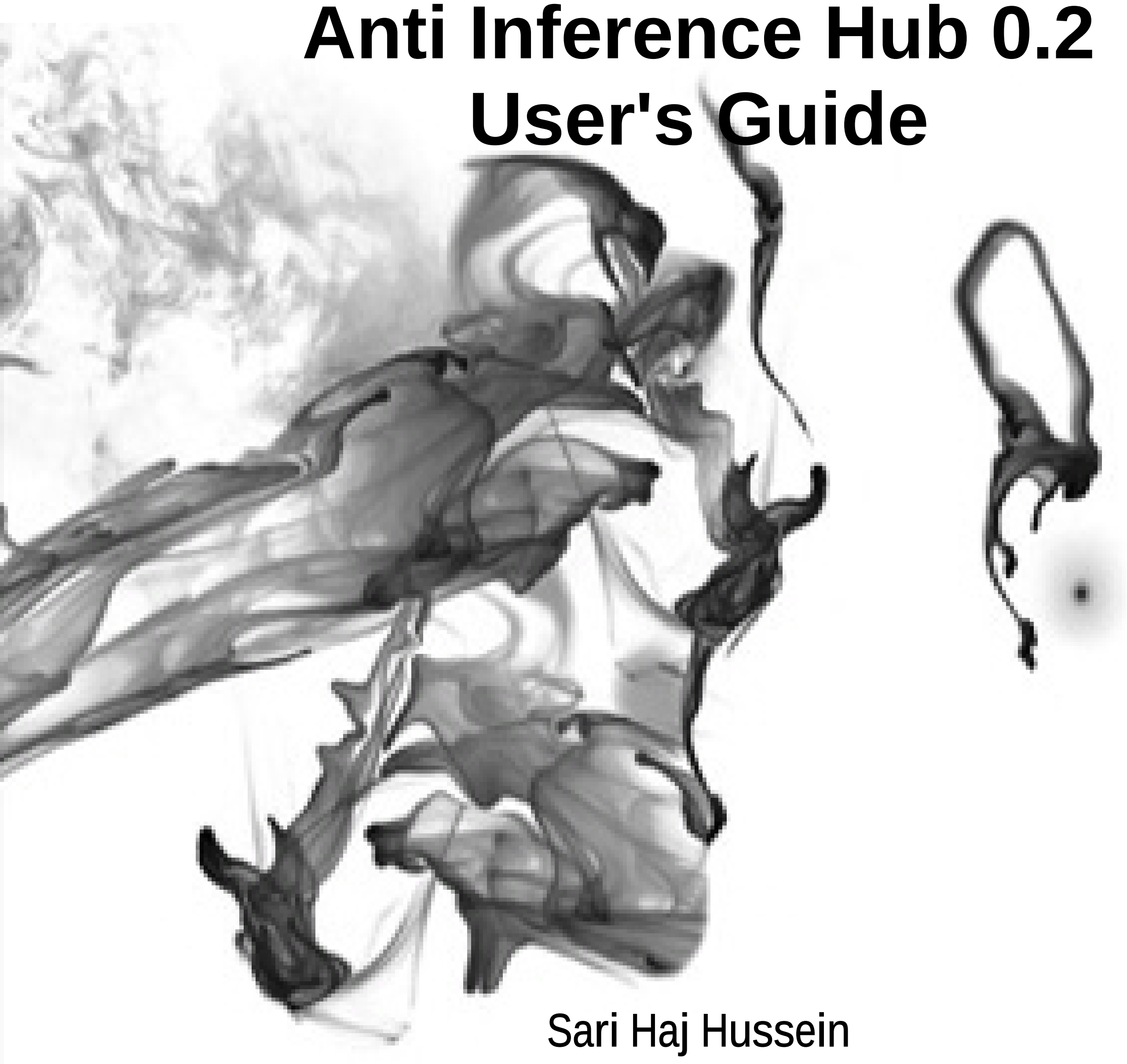
Anti Inference Hub 0.2
User's Guide



Sari Haj Hussein
Chalmers University of Technology

Table of Contents

Chapter 1. Introduction

1.1. What is Anti Inference Hub?

Anti Inference Hub is the first dynamic query processing engine that defends against the Inference Problem in Multilevel Databases by integrating smoothly with common DBMSs (Oracle, PostgreSQL, and MySQL), and monitoring queries submitted by end users.

1.2. Some Intended Purposes

Here are some example usages of Anti Inference Hub:

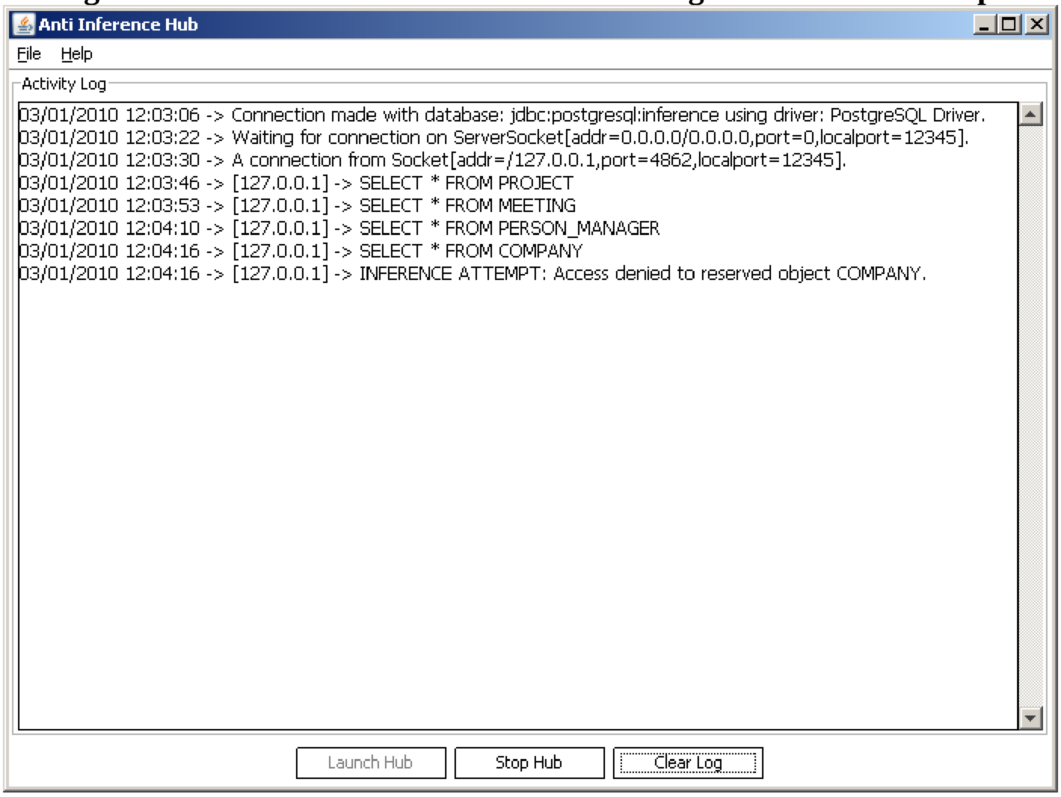
- Database designers use it to estimate the security of their database design.
- Database administrators use it to protect sensitive data stored in the database.
- Network security engineers use it to examine security problems, and identify attackers.
- People use it to learn more about the Inference Problem, and Database Security in whole.

1.3. Features

Following are some of the many features provided by Anti Inference Hub:

- Available for any operating system with Java installed.
- Integrates easily with Oracle, PotgreSQL, and MySQL.
- Fast query processing and analyzing.
- Results for safe queries are provided in a platform independent XML format.

Figure 1.1. Anti Inference Hub after Thwarting an Inference Attempt



```
03/01/2010 12:03:06 -> Connection made with database: jdbc:postgresql:inference using driver: PostgreSQL Driver.
03/01/2010 12:03:22 -> Waiting for connection on ServerSocket[addr=0.0.0.0/0.0.0.0,port=0,localport=12345].
03/01/2010 12:03:30 -> A connection from Socket[addr=/127.0.0.1,port=4862,localport=12345].
03/01/2010 12:03:46 -> [127.0.0.1] -> SELECT * FROM PROJECT
03/01/2010 12:03:53 -> [127.0.0.1] -> SELECT * FROM MEETING
03/01/2010 12:04:10 -> [127.0.0.1] -> SELECT * FROM PERSON_MANAGER
03/01/2010 12:04:16 -> [127.0.0.1] -> SELECT * FROM COMPANY
03/01/2010 12:04:16 -> [127.0.0.1] -> INFERENCE ATTEMPT: Access denied to reserved object COMPANY.
```

1.4. Open Source Software

Anti Inference Hub is an open source software project, and is released under the GNU General Public License (GPL). You can freely use Anti Inference Hub on any number of computers you like, without worrying about license keys or fees or such. In addition, all source code is freely available under the GPL. Because of that, it is very easy for people to contribute to the project.

1.5. What Anti Inference Hub is Not

- Anti Inference Hub is not an intrusion prevention system (IPS). It only prevents inference attacks.
- Anti Inference Hub does not interfere with network traffic.

1.6. Supported Operating Systems

Anti Inference Hub should run on any operating system with Java installed.

1.7. Where to Get Anti Inference Hub?

You can get the latest copy of Anti Inference Hub from its website: https://github.com/angyjoe/aih.

1.8. Development and Maintenance of Anti Inference Hub

Anti Inference Hub was developed by Sari Haj Hussein. Ongoing development and maintenance of Anti Inference Hub are also handled by Sari Haj Hussein. Anti Inference Hub is an open source software project, and is released under the GNU General Public License (GPL). All source code is freely available under the GPL. You are welcome to modify Anti Inference Hub to suit your own needs, and it would be appreciated if you contribute your improvements back to us. The Anti Inference Hub source code is available from Anti Inference Hub website: https://github.com/angyjoe/aih.

1.9. Reporting Problems and Getting Help

If you have problems, or need help with Anti Inference Hub, please post your questions to Anti Inference Hub mailing list at https://lists.sourceforge.net/lists/listinfo/aih-list.

Chapter 2. Installing Anti Inference Hub

2.1. Introduction

To use Anti Inference Hub, you must obtain a binary package for your operating system, then install it into its final destinations.

2.2. Obtaining the Binary Package

You can obtain the binary package from Anti Inference Hub website: https://github.com/angyjoe/aih.

2.3. Installing the Binary Package Under Windows

Unpack Anti Inference Hub binary package into a directory of choice.

2.4. Installing the Binary Package Under Linux

Same as in Windows, unpack Anti Inference Hub binary package into a directory of choice.

2.5. Uninstalling Anti Inference Hub

Remove the directory where you unpacked Anti Inference Hub binary package.

Chapter 3. User Interface

3.1. Introduction

By now you have installed Anti Inference Hub and are most likely keen to get started exploring the user interface.

3.2. Starting Anti Inference Hub

To start Anti Inference Hub under Windows, use:

- The file RunHub.bat to start the Hub.
- The file RunHubClient.bat to start the Hub Client.

And to start Anti Inference Hub under Linux, use:

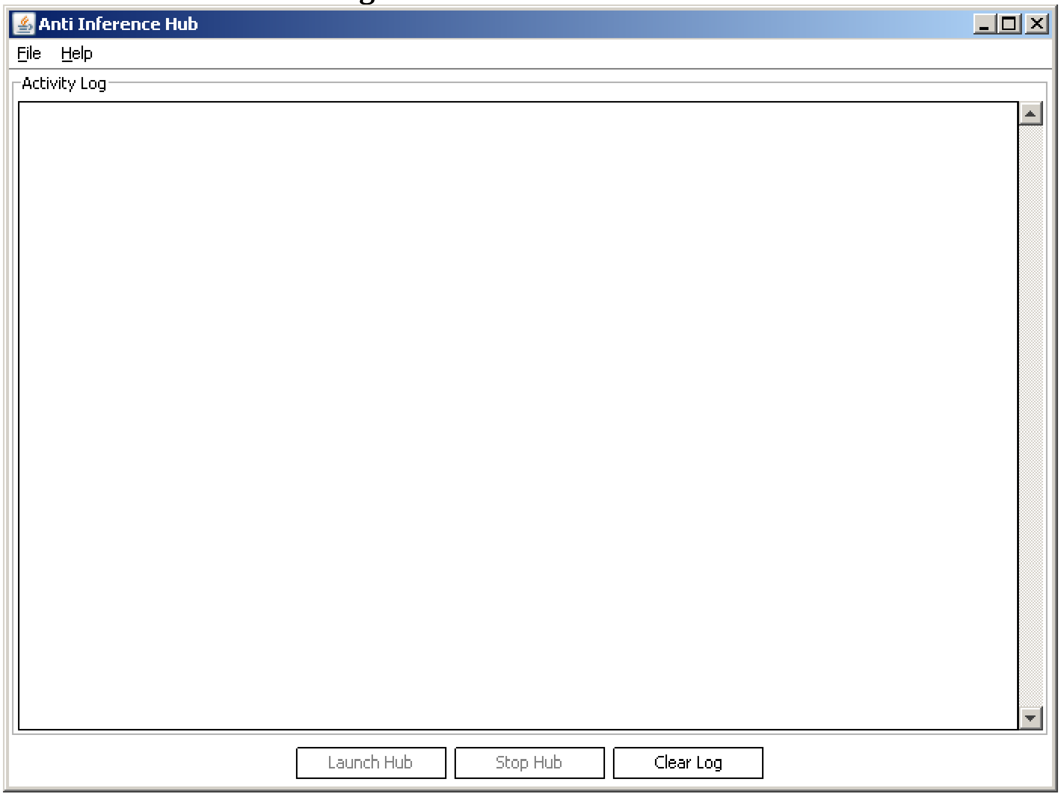
- The file RunHub.sh to start the Hub.
- The file RunHubClient.sh to start the Hub Client.

All of these files are included in Anti Inference Hub binary package.

3.3. The Hub Window

Let's look at the Hub Window.

Figure 3.1. The Hub Window



The Hub Window consists of parts that are commonly known from many other GUI programs.

1. The Menu that is used to configure the Hub.
2. The Activity Log Panel that displays a summary of the activities done by the Hub.
3. The Buttons Panel that controls whether the Hub is on or off, and clears the log.

3.3.1. The Menu for the Hub Window

The Menu for the Hub window sits on top of the window. Menu items will be grayed out if the corresponding feature is not available. For example, you cannot setup inference channels before setting up a database connection.

Figure 3.2. The Menu for the Hub Window

The Menu contains the following items:

File	This menu contains items to setup database connection, inference channels, initialize keys, adjust super clients, and exit the program.
Help	This menu contains items to help the user, and the usual about dialog.

3.3.1.1. The File Menu for the Hub Window

Figure 3.3. The File Menu for the Hub window

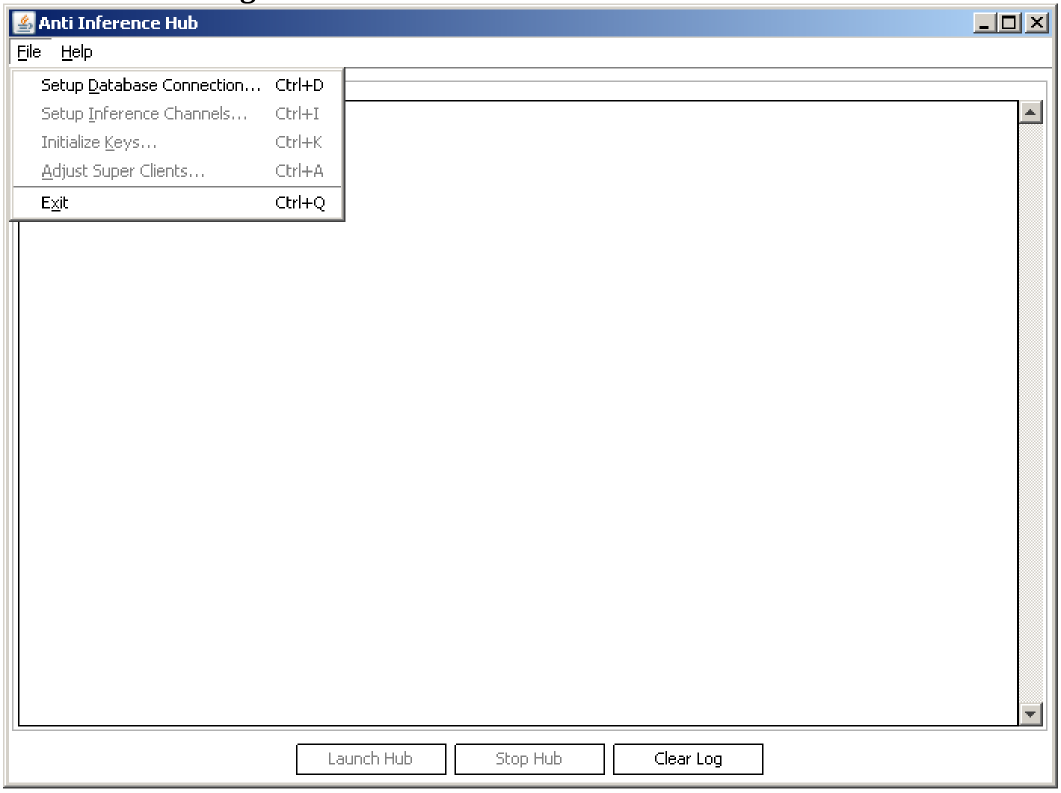

Anti Inference Hub
File
Help
Setup Database Connection... Ctrl+D
Setup Inference Channels... Ctrl+I
Initialize Keys... Ctrl+K
Adjust Super Clients... Ctrl+A
Exit Ctrl+Q
Launch Hub
Stop Hub
Clear Log

Table 3.1. Items in the File Menu for the Hub window

Menu Item	Accelerator	Description
Setup Database Connection…	Ctrl+D	This menu item brings up the Setup Database Connection Box that allows you to setup connection with supported DBMSs (Oracle, PostgreSQL, and MySQL).
Setup Inference Channels…	Ctrl+I	This menu item brings up the Setup Inference Channels Box that allows you to view, add, and remove inference channels for the database.

Initialize Keys…	Ctrl+K	This menu item brings up the Initialize Keys Box that allows you to view, initialize, and remove keys for inference channels objects.
Adjust Super Clients…	Ctrl+A	This menu item brings up the Adjust Super Clients Box that allows you to view, add, and remove super clients connecting with the Hub.
Exit	Ctrl+Q	This menu item quits the Hub.

3.3.1.2. The Help Menu for the Hub Window

Figure 3.4. The Help Menu for the Hub window

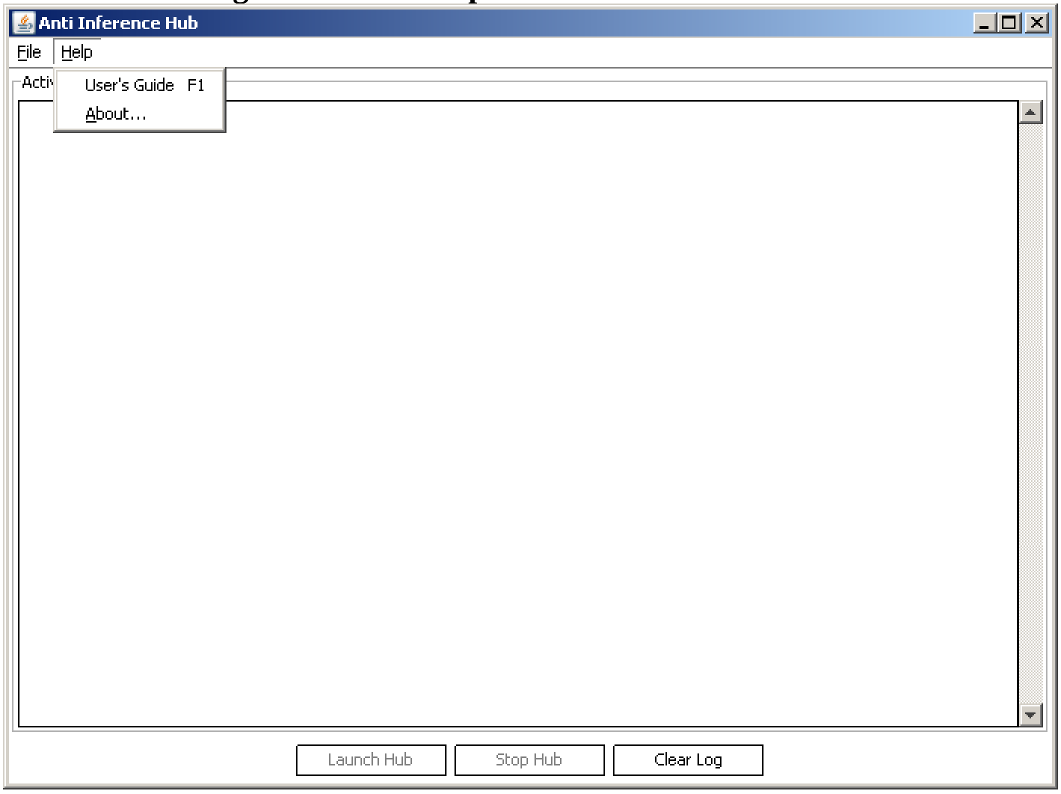


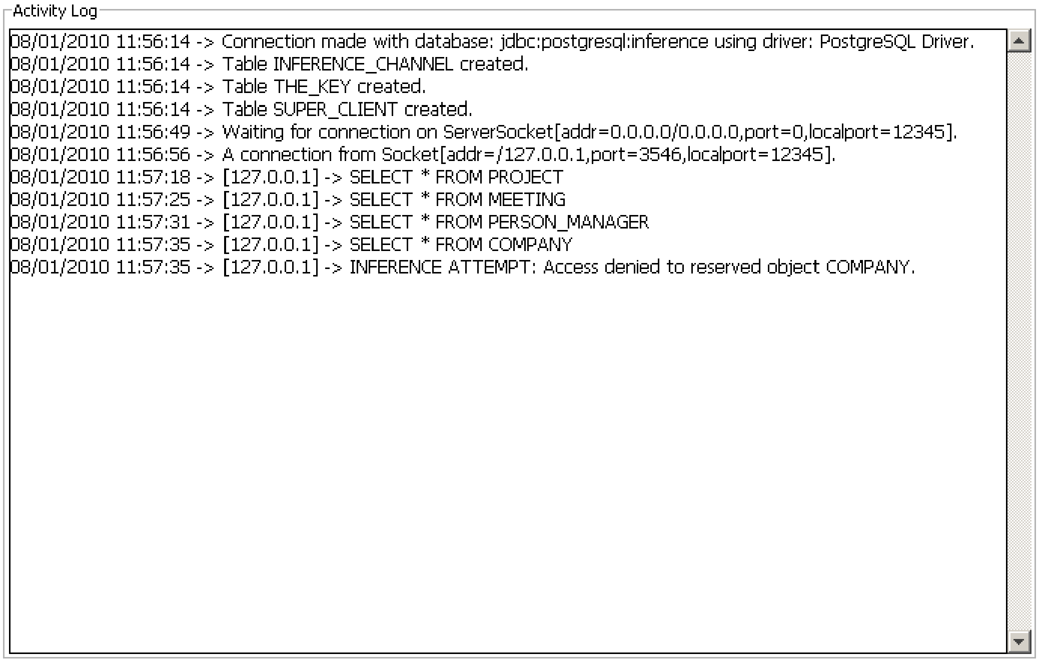
Table 3.2. Items in the Help Menu for the Hub window

Menu Item	Accelerator	Description
User's Guide	F1	This menu item opens the user's guide.
About…		This menu item brings up the About Box that provides some information on Anti Inference Hub, such as license, and third party packages used.

3.3.2. The Activity Log Panel

The Activity Log Panel displays a summary of the activities done by the Hub.

Figure 3.5. The Activity Log Panel



```
08/01/2010 11:56:14 -> Connection made with database: jdbc:postgresql:inference using driver: PostgreSQL Driver.
08/01/2010 11:56:14 -> Table INFERENCE_CHANNEL created.
08/01/2010 11:56:14 -> Table THE_KEY created.
08/01/2010 11:56:14 -> Table SUPER_CLIENT created.
08/01/2010 11:56:49 -> Waiting for connection on ServerSocket[addr=0.0.0.0/0.0.0.0,port=0,localport=12345].
08/01/2010 11:56:56 -> A connection from Socket[addr=/127.0.0.1,port=3546,localport=12345].
08/01/2010 11:57:18 -> [127.0.0.1] -> SELECT * FROM PROJECT
08/01/2010 11:57:25 -> [127.0.0.1] -> SELECT * FROM MEETING
08/01/2010 11:57:31 -> [127.0.0.1] -> SELECT * FROM PERSON_MANAGER
08/01/2010 11:57:35 -> [127.0.0.1] -> SELECT * FROM COMPANY
08/01/2010 11:57:35 -> [127.0.0.1] -> INFERENCE ATTEMPT: Access denied to reserved object COMPANY.
```

The activities logged in the Activity Log Panel are dated and timed, and they can be one of the following:

1. Establishing a successful connection with the DBMS.
2. Successful creation of statistical tables needed for query processing. The tables are INFERENCE_CHANNEL, THE_KEY, and SUPER_CLIENT.
3. Listening for connection requests from clients on a default port (which is 12345).
4. Queries received by the Hub, and clients who sent them.
5. Inference attempts thwarted by the Hub, and clients who initiated the attacks.
6. Inference attempts permitted by the Hub, and super clients who were allowed to infer.

3.3.3. The Buttons Panel for the Hub Window

Figure 3.6. The Buttons Panel for the Hub Window

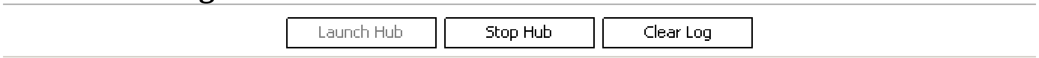


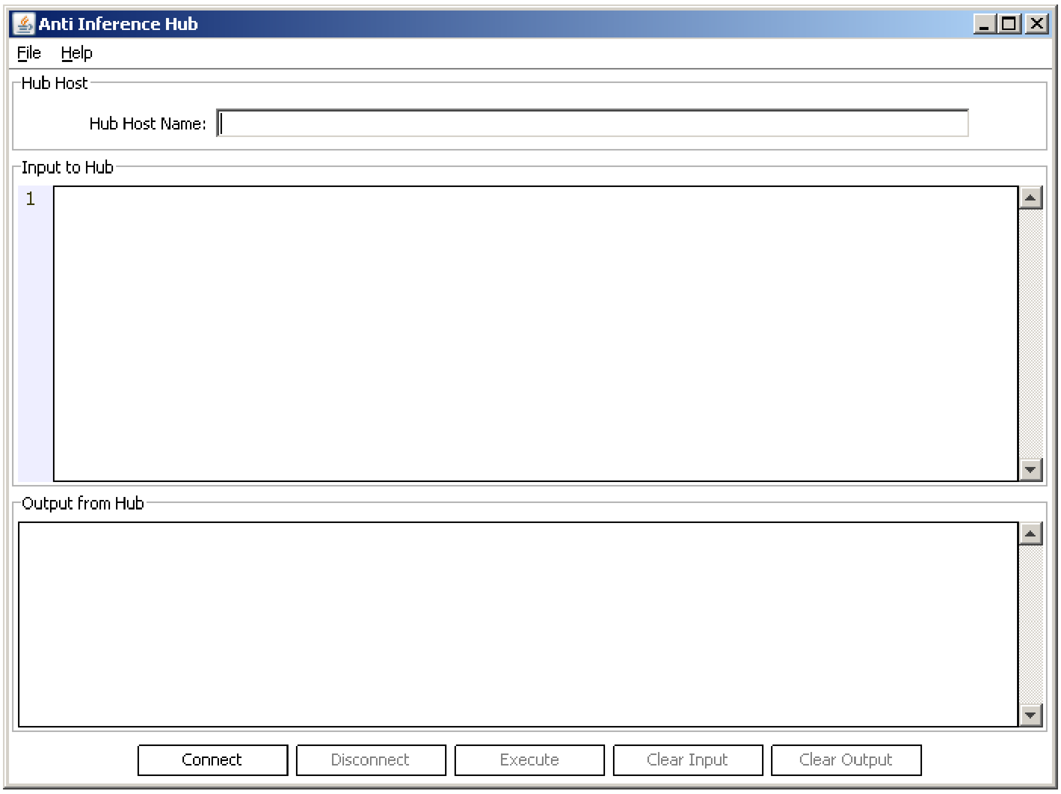
Table 3.3. Buttons in the Buttons Panel for the Hub Window

Button	Description
Launch Hub	This button is used to launch the Hub so that it listens for connection requests from clients.
Stop Hub	This button is used to stop the Hub so that no more connections are allowed.
Clear Log	This button is used to clear the content of the Activity Log Panel.

3.4. The Hub Client Window

Let's look at the Hub Client Window.

Figure 3.7. The Hub Client Window



The Hub Client Window consists of parts that are commonly known from many other GUI programs.

1. The Menu that has fewer functionalities than that of the Hub Window.
2. The Hub Host Name Panel that is used to enter the host name of the Hub the Hub Client will connect to.
3. The Input to Hub Panel that is used to enter SQL queries.
4. The Output from Hub Panel that is used to display Hub responses to SQL queries.
5. The Buttons Panel that controls whether the Hub Client is connected to the Hub or not, executes queries, and clears input/output.

3.4.1. The Menu for the Hub Client Window

The menu for the Hub Client Window sits on top of the window.

Figure 3.8. The Menu for the Hub Client Window

The Menu contains the following items:

File	This menu contains an item to exit the program.
Help	This menu contains items to help the user, and the usual about dialog.

3.4.1.1. The File Menu for the Hub Client Window

Figure 3.9. The File menu for the Hub Client Window

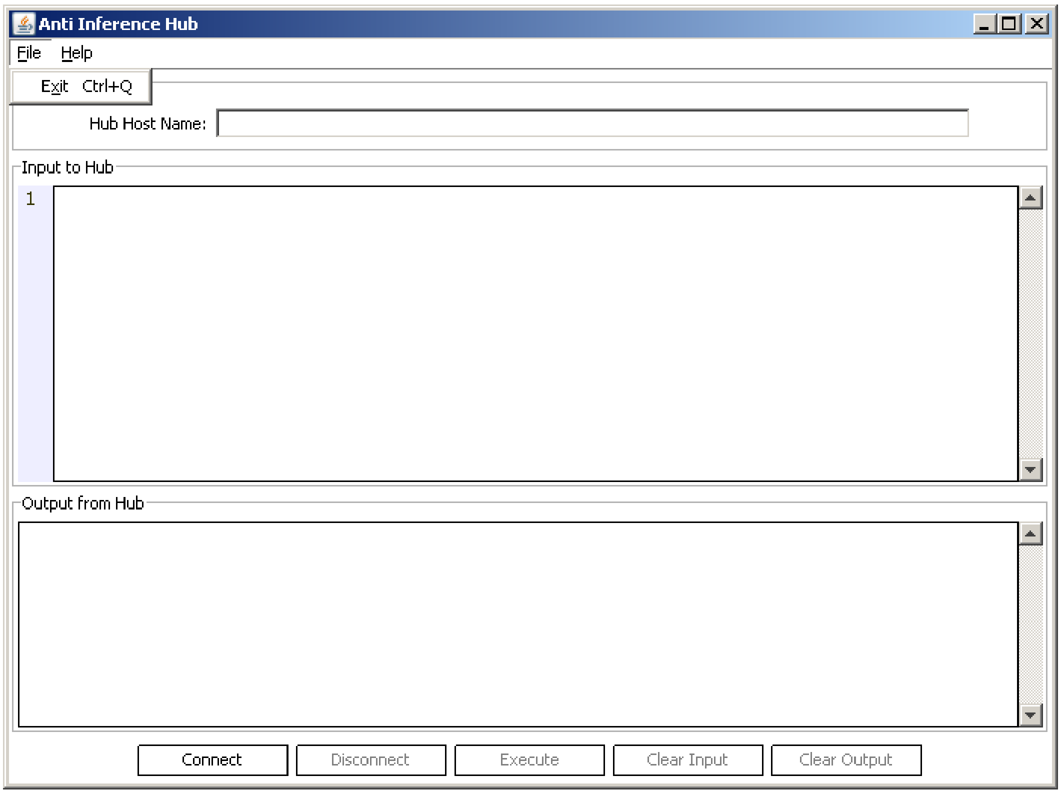


Table 3.4. Items in the File menu for the Hub Client Window

Menu Item	Accelerator	Description
Exit	Ctrl+Q	This menu item quits the Hub Client.

3.4.1.2. The Help Menu for the Hub Client Window

Figure 3.10. The Help Menu for the Hub Client window

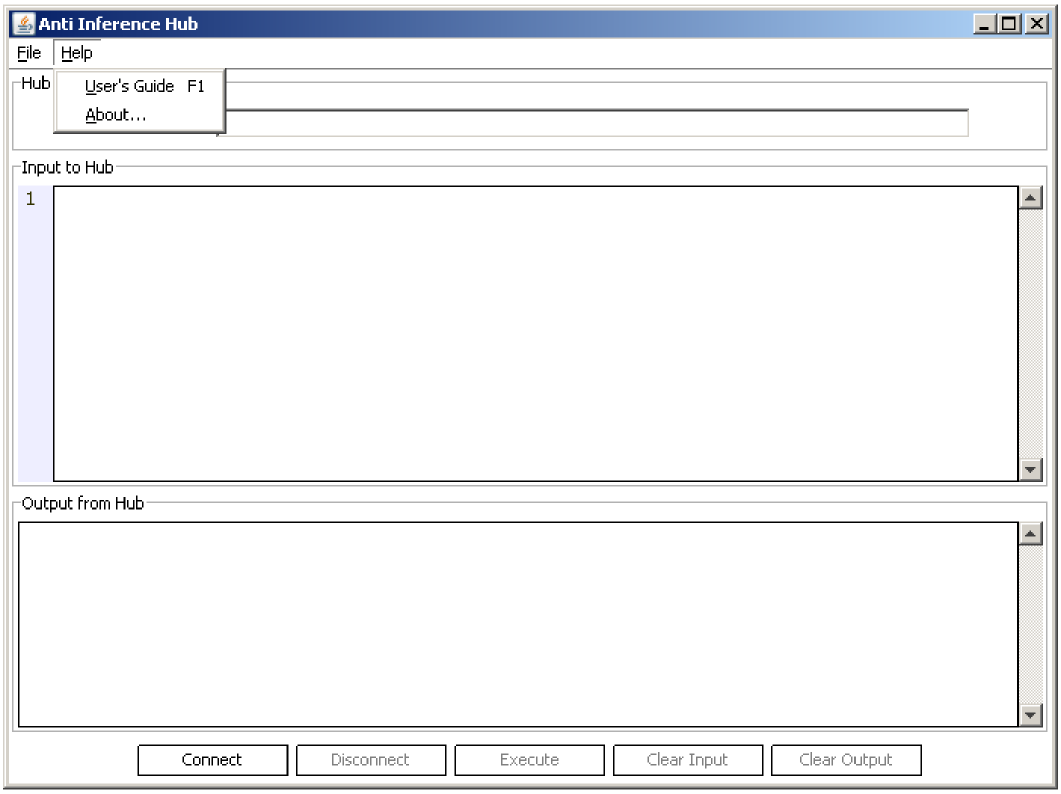


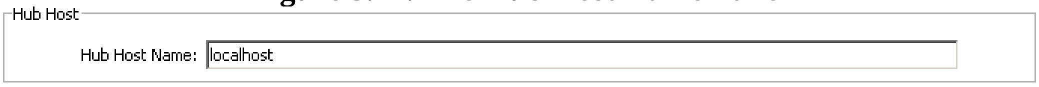
Table 3.5. Items in the Help Menu for the Hub Client window

Menu Item	Accelerator	Description
User's Guide	F1	This menu item opens the user's guide.
About…		This menu item brings up the About Box that provides some information on Anti Inference Hub, such as license, and third party packages used.

3.4.2. The Hub Host Name Panel

The Hub Host Name Panel is used to enter the host name of the Hub the Hub Client will connect to. The host name can be either a machine name or a textual representation of an IP address.

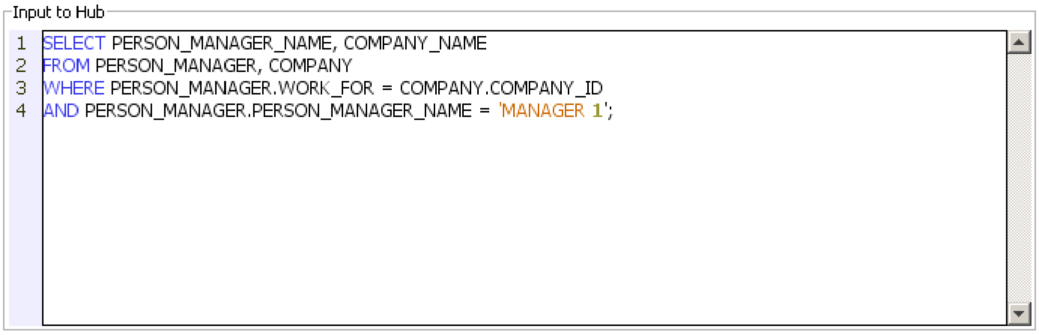
Figure 3.11. The Hub Host Name Panel



3.4.3. The Input to Hub Panel

The Input to Hub Panel is used to enter SQL queries. It comes with an out-of-the-box SQL syntax highlighter.

Figure 3.12. The Input to Hub Panel



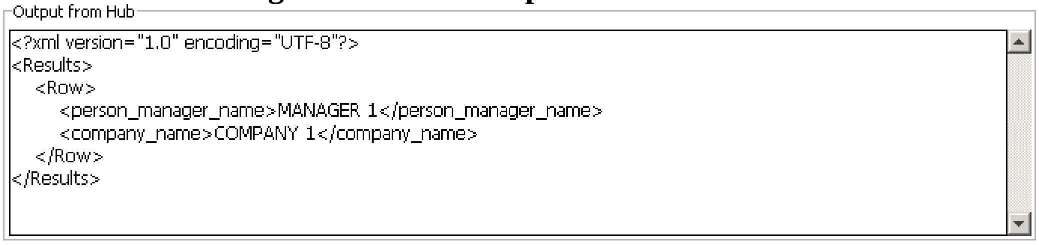
Input to Hub

```
SELECT PERSON_MANAGER_NAME, COMPANY_NAME
FROM PERSON_MANAGER, COMPANY
WHERE PERSON_MANAGER.WORK_FOR = COMPANY.COMPANY_ID
AND PERSON_MANAGER.PERSON_MANAGER_NAME = 'MANAGER 1';
```

3.4.4. The Output from Hub Panel

The Output from Hub Panel is used to display Hub responses to SQL queries. Results for safe SQL queries are provided in a platform independent XML format, whereas, notifications of blocked queries, and other error messages are provided in a text format.

Figure 3.13. The Output from Hub Panel



Output from Hub

```
<?xml version="1.0" encoding="UTF-8"?>
<Results>
   <Row>
      <person_manager_name>MANAGER 1</person_manager_name>
      <company_name>COMPANY 1</company_name>
   </Row>
</Results>
```

3.4.5. The Buttons Panel for the Hub Client Window

Figure 3.14. The Button Panel for the Hub Client Window

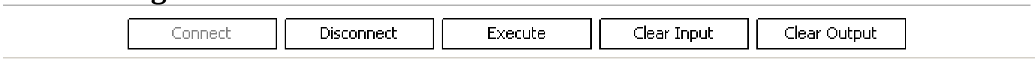


Table 3.6. Buttons in the Buttons Panel for the Hub Client Window

Button	Description
Connect	This button is used to connect the Hub Client to the Hub specified in the Hub Host Name field.
Disconnect	This button is used to disconnect the Hub Client from the Hub.
Execute	This button is used to send the content (SQL query) of the Input to Hub Panel to the Hub to be executed.
Clear Input	This button is used to clear the content of the Input to Hub Panel.
Clear Output	This button is used to clear the content of the Output from Hub Panel.

Chapter 4. Connecting with the Database

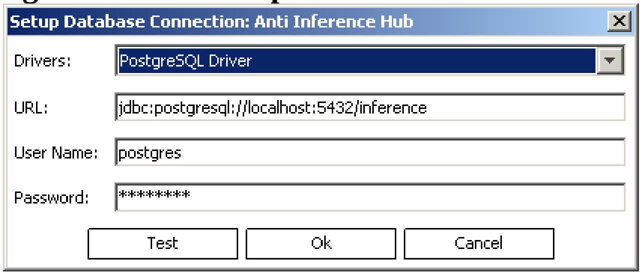
4.1. Introduction

Connecting the Hub with the database is the first step you should do to get it to work. Anti Inference Hub can protect databases designed using Oracle, PostgreSQL, and MySQL.

4.2. The Setup Database Connection Box

The Setup Database Connection Box is accessible through the Setup Database Connection menu item in the Hub Window, or the accelerator Ctrl+D.

Figure 4.1. The Setup Database Connection Box



Setup Database Connection: Anti Inference Hub

Drivers: PostgreSQL Driver

URL: jdbc:postgresql://localhost:5432/inference

User Name: postgres

Password: *********

Table 4.1. Components in the Setup Database Connection Box

Component	Description
Drivers Combo Box	This combo box is used to specify the JDBC driver the Hub will use to connect to the database. Depending on the DBMS you are using, this will be Oracle Thin Driver, or PostgreSQL Driver, or MySQL Driver.
URL Text Field	This field is used to specify the URL of the database the Hub will connect to. The URL depends on the driver specified in the Drivers Combo Box.
User Name Text Field	This field is used to specify the user name under which the Hub will connect to the database.
Password Text Field	This field is used to specify the password under which the Hub will connect to the database.
Test Button	This button is used to test the connection between the Hub and the database.
OK Button	This button is used to save the connection between the Hub and the database for ongoing usage by the Hub. It also closes the Setup Database Connection Box.
Cancel Button	This button is used to close the Setup Database Connection Box without saving any thing.

Chapter 5. Setting Inference Channels

5.1. Introduction

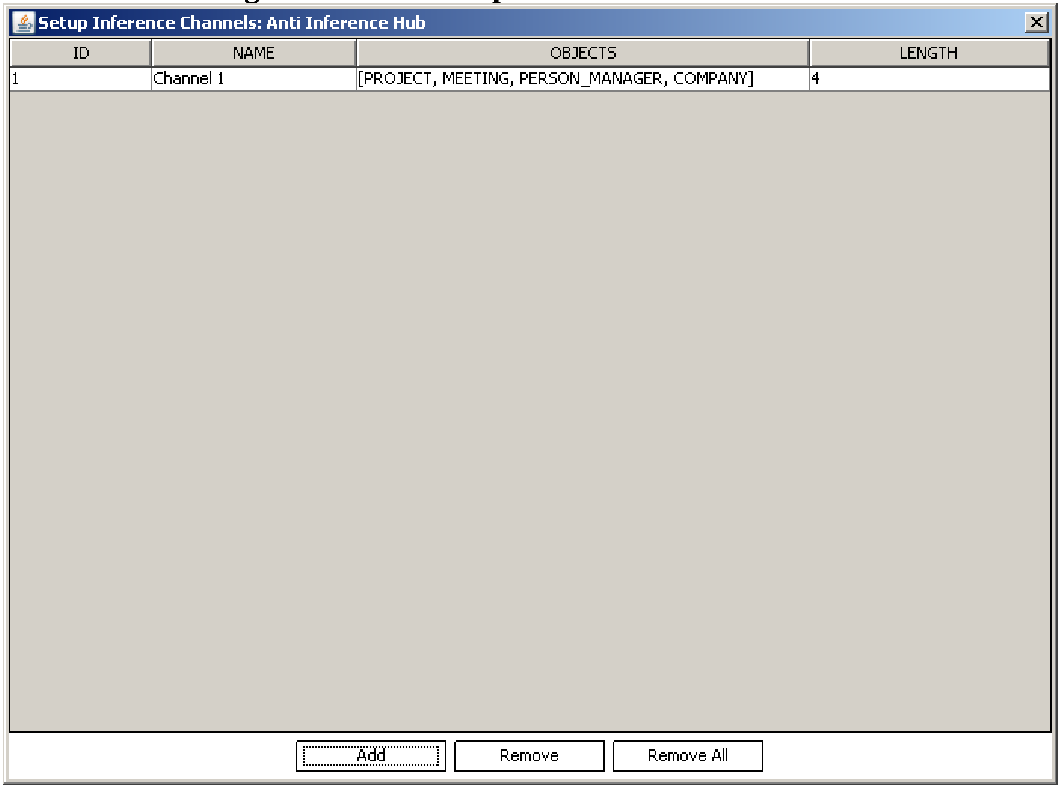
You should manually locate inference channels in the database so that the Hub can distinguish safe queries from unsafe ones. You should thoroughly consider your database design, and try to be as accurate as possible when doing this, otherwise, the Hub may block responses to safe queries!

Over time, many algorithms for locating inference channels were proposed by researches; however, all of them tend to generate inaccurate and unsatisfying results, therefore, Anti Inference Hub does not implement any of these algorithms, rather, it transfers the task of locating inference channels to the database designer (or the database security specialist). When a sound algorithm for locating inference channels has been proposed in the scientific community, we will make it available in Anti Inference Hub. For further information about this issue, please refer to Section 8.4. Tools and Techniques for Dealing with Inference Channels.

5.2. The Setup Inference Channels Box

The Setup Inference Channels Box is accessible through the Setup Inference Channels menu item in the Hub Window, or the accelerator Ctrl+I.

Figure 5.1. The Setup Inference Channels Box



ID	NAME	OBJECTS	LENGTH
1	Channel 1	[PROJECT, MEETING, PERSON_MANAGER, COMPANY]	4

Table 5.1. Components in the Setup Inference Channels Box

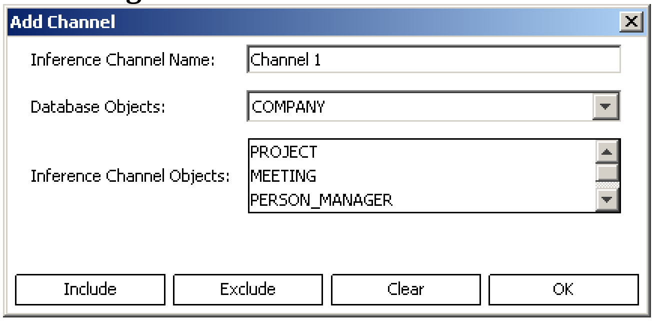
Component	Description
Inference Channels Table	This table lists currently located inference channels in the database. Every inference channel is characterized by its ID, name, objects contained in it, and its length.
Add Button	This button is used to locate a new inference channel in the database. When clicked, it brings

	up the Add Channel Box.
Remove Button	This button is used to remove the selected inference channel in the Inference Channels Table.
Remove All Button	This button is used to remove all inference channels in the Inference Channels Table whether selected or not.

5.3. The Add Channel Box

The Add Channel Box is used to locate a new inference channel in the database.

Figure 5.2. The Add Channel Box



Inference Channel Name: Channel 1

Database Objects: COMPANY

Inference Channel Objects: PROJECT
MEETING
PERSON_MANAGER

Table 5.2. Components in the Add Channel Box

Component	Description
Inference Channel Name Text Field	This field is used to specify the name of the new inference channel. A name must be specified.
Database Objects Combo Box	This combo box is used to specify a database object to include in the new inference channel. One object at least must be included.
Inference Channel Objects List	This list displays the current selection of database objects that will be included in the new inference channel.
Include Button	This button is used to include the selected database object from Database Objects Combo Box in the Inference Channel Objects List.
Exclude Button	This button is used to exclude the selected database objects in the Inference Channel Objects List. Note that more than one database object can be selected in the Inference Channel Objects List.
Clear Button	This button is used to clear the content of the Inference Channel Objects List.
OK Button	This button is used to save the new inference channel in the database. It also closes the Add Channel Box.

Chapter 6. Initializing Keys

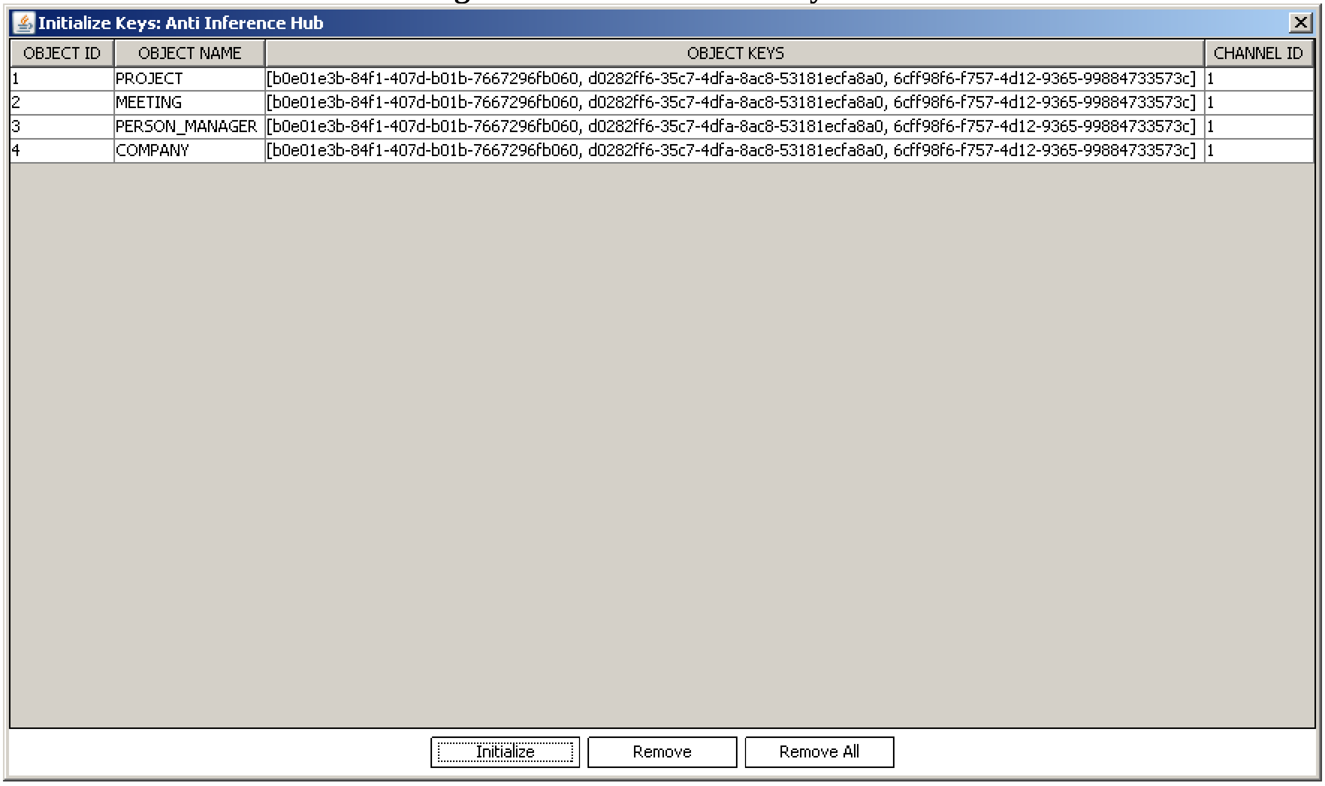
6.1. Introduction

Initializing keys for inference channels objects is an automatic process that you should consider right after locating inference channels in the database. For further information about the purpose of the keys, please refer to Chapter 9. Technology behind Anti Inference Hub.

6.2. The Initialize Keys Box

The Initialize Keys Box is accessible through the Initialize Keys menu item in the Hub Window, or the accelerator Ctrl+K.

Figure 6.1. The Initialize Keys Box



Initialize Keys: Anti Inference Hub

OBJECT ID	OBJECT NAME	OBJECT KEYS	CHANNEL ID
1	PROJECT	[b0e01e3b-84f1-407d-b01b-7667296fb060, d0282ff6-35c7-4dfa-8ac8-53181ecfa8a0, 6cff98f6-f757-4d12-9365-99884733573c]	1
2	MEETING	[b0e01e3b-84f1-407d-b01b-7667296fb060, d0282ff6-35c7-4dfa-8ac8-53181ecfa8a0, 6cff98f6-f757-4d12-9365-99884733573c]	1
3	PERSON_MANAGER	[b0e01e3b-84f1-407d-b01b-7667296fb060, d0282ff6-35c7-4dfa-8ac8-53181ecfa8a0, 6cff98f6-f757-4d12-9365-99884733573c]	1
4	COMPANY	[b0e01e3b-84f1-407d-b01b-7667296fb060, d0282ff6-35c7-4dfa-8ac8-53181ecfa8a0, 6cff98f6-f757-4d12-9365-99884733573c]	1

Table 6.1. Components in the Initialize Keys Box

Component	Description
Keys Table	This table lists all objects in all inference channels along with their keys. Information in this table is characterized by object ID, object name, object keys, and inference channel ID the object belongs to.
Initialize Button	This button is used to initialize keys for all objects in all inference channels in the database.
Remove Button	This button is used to remove the selected association between an object and its keys in the Keys Table. Note though that objects are not removed from inference channels.
Remove All Button	This button is used to remove all associations between objects and their keys in the Keys

	Table whether selected or not. Note again that objects are not removed from inference channels.

Chapter 7. Adjusting Super Clients

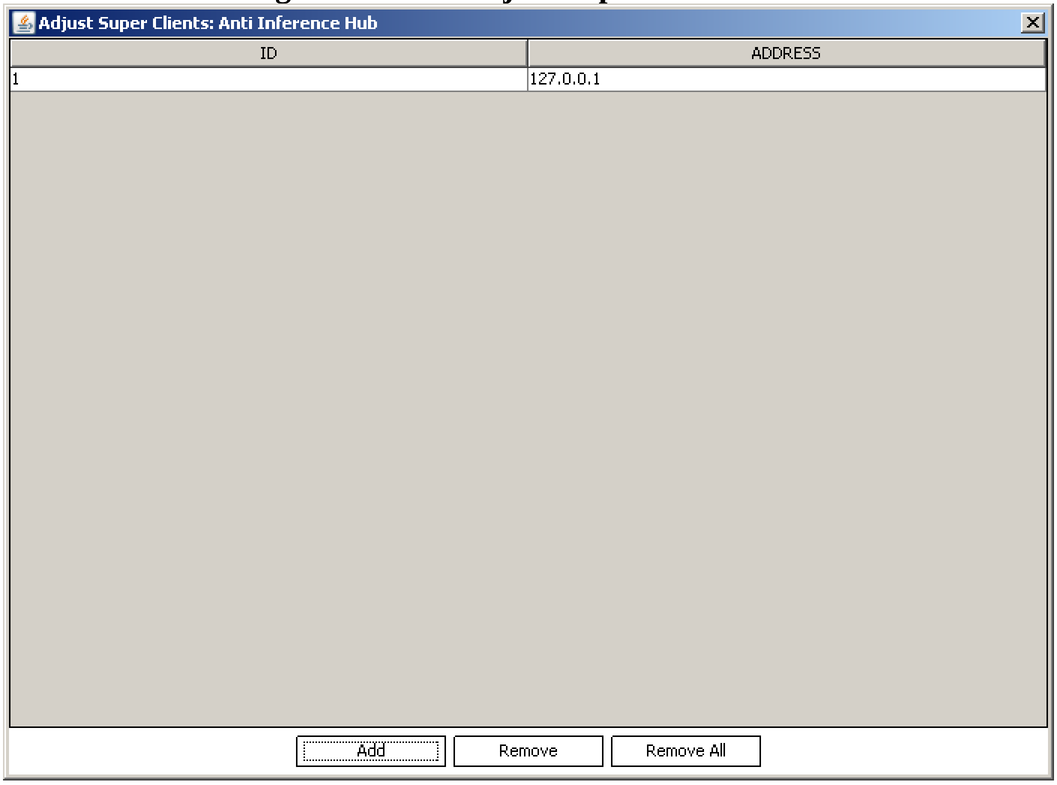
7.1. Introduction

Adjusting super clients is an optional step that you may consider before launching the Hub. A super client is a Hub Client that is allowed to infer, or in other words, a super client is a client that is considered database-friendly by the Hub! There is a very sound rationale behind allowing super clients in Anti Inference Hub. Read more about this by referring to Chapter 9. Technology Behind Anti Inference Hub.

7.2. The Adjust Super Clients Box

The Adjust Super Clients Box is accessible through the Adjust Super Clients menu item in the Hub Window, or the accelerator Ctrl+A.

Figure 7.1. The Adjust Super Clients Box



Adjust Super Clients: Anti Inference Hub

ID	ADDRESS
1	127.0.0.1

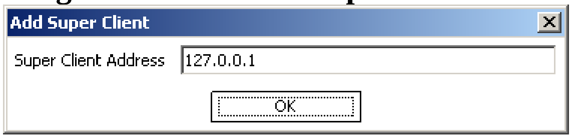
Table 7.1. Components in the Adjust Super Clients Box

Component	Description
Super Clients Table	This table lists all super clients. Every super client is characterized by its ID, and the textual representation of its IP address.
Add Button	This button is used to adjust a new super client. When clicked, it brings up the Add Super Client Box.
Remove Button	This button is used to remove the selected super client from the Super Clients Table.
Remove All Button	This button is used to remove all super clients in the Super Clients Table whether selected or not.

7.3. The Add Super Client Box

The Add Super Client Box is used to adjust a new super client.

Figure 7.2. The Add Super Client Box



Add Super Client

Super Client Address: 127.0.0.1

Table 7.2. Components in the Add Super Client Box

Component	Description
Super Client Address Field	This field is used to specify the textual representation of the IP address of the new super client. This field has an installed IPv4 address filter, therefore, any attempt by the user to fill it with an invalid IP address will cause the field to revert to its default value which is 127.0.0.1.
OK Button	This button is used to save the new super client. It also closes the Add Super Client Box.

Chapter 8. The Inference Problem: Demystified

8.1. Introduction

An inference channel in a database is a construction by which an attacker can deduce sensitive data from nonsensitive data. The inference problem is the problem of identifying and then removing any inference channel in a database. This problem is rather difficult to resolve due to the fact that a database is normally open to queries, and when queries' results are put together, we can learn about new data. This makes it difficult to determine the exact amount of data that can be learnt by an outsider [1]. The inference problem usually occurs in multilevel secure databases, which are characterized by the following :

- The security of one element may be different than the security of other elements in the same row or column.
- We need several levels of security in a database; only sensitive and nonsensitive are not enough.
- The security of a sum, a count or an average of elements may be different from the security of individual elements.

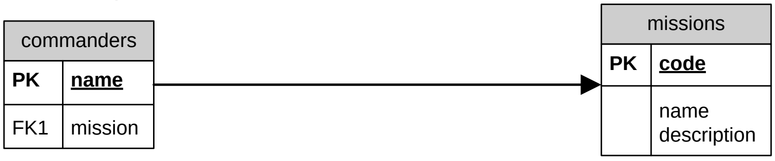
The inference problem can also occur in statistical databases that run as Online Analytical Processing (OLAP) systems and enable their users to retrieve only aggregate statistics (count, sum, average or standard deviation). These databases are typically used as data warehouses or data mines for the purpose of business intelligence. Securing this type of databases is arguably more difficult than securing multilevel secure databases and it falls within what researchers call "Privacy Preserving Data Mining (PPDM)". Anti Inference Hub is centric around defending multilevel secure databases against the inference problem. The work may later be extended to secure statistical databases as well. The inference problem, in both multilevel secure databases and statistical databases, is discussed in the following sections.

8.2. Example Inference Problems

In this section, we example some of the inference channels that have been identified and published in database security literature.

8.2.1. Inference from Queries Based on Sensitive Data

Suppose we have the following two relations in a database [1]:



Classification: Nonsensitive Data

Classification: Sensitive Data

Suppose further that a low user issued the following SQL query:

```
SELECT commanders.name
FROM commanders, missions
WHERE commanders.mission = missions.code;
```

In relational algebra, the above query can be expressed using the formula:

$\pi_{\text{commanders.name}}\ \sigma_{\text{commanders.mission = missions.code}}$ (commanders x missions)

This query is evaluated by first taking the Cartesian product of the relations commanders and missions, selecting rows that satisfies the condition commanders.mission = missions.code, then projecting the values of the commanders.name column. Thus, query result will contain only commanders.name, which is classified as nonsensitive data. Nevertheless, we have an inference channel (!) because we are using sensitive data to create the Cartesian product; namely the column missions.code. Inference channels of this type are easy to identify and remove. We either modify user query excluding sensitive data or we abort its execution.

8.2.2. Inference in Statistical Databases

Suppose we have a hospital database that stores patients' medical records [3]. Each patient's record contains the following data {Age, Sex, Employer, Social Security Number, Diagnosis Type}. Suppose further that physicians in the hospital are allowed to access all the medical records, whereas researchers are only allowed to perform aggregations on subsets of the medical records. For example, a researcher can get the result of the following aggregation: COUNT [(Sex = Male) & (Employer = Volvo)], which means the number of male patients who work for Volvo.

Suppose now that an evil researcher wants to illegally determine the diagnosis type of a patient Lisbeth whose age is 34 and works for Ericsson. He first issues the following query:

Query 1: COUNT [(Age = 34) & (Sex = Female) & (Employer = Ericsson)]

If query 1 returns 1, it means that the evil researcher succeeded at locating Lisbeth in the database. He then issues the following query:

Query 2: COUNT [(Age = 34) & (Sex = Female) & (Employer = Ericsson) & (Diagnosis Type = Insomnia)]

Now the evil researcher is before one of the following two cases:

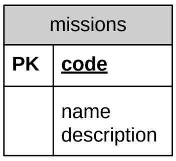
- If query 2 returns 1, we say that the hospital database is completely (or positively) compromised, and the evil researcher succeeded at determining that Lisbeth is insomniac.
- If query 2 returns 0, we say that the hospital database is partially (or negatively) compromised, and the evil researcher succeeded at determining that Lisbeth is (not) insomniac.

8.2.3. Inference from Data Combined with Metadata

In a database management system, meta data are data that describe a database or one of its parts e.g. a table that describes all tables in a database is a meta data. When ordinary data retrieved from a database are combined with knowledge of meta data, new ways of making inference arise. In this section we focus on some of these ways.

8.2.3.1. Inference from Key Integrity

Suppose we have the following relation in a database [1]:



Suppose further that a high user inserted the following tuple into the relation:

classification	code	name	description
sensitive data	0XX	Skyscraper	Move the artillery to Ohio

Note that data is classified at the tuple level. Suppose now that a low user would like to insert the following tuple into the relation:

classification	code	name	description
nonsensitive data	0XX	Dogscratcher	Move the artillery to Missouri

If we want to preserve key integrity, then we are before one of the following two cases:

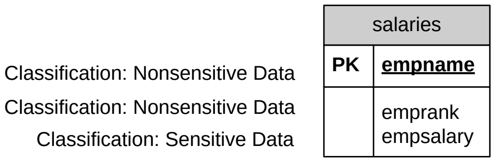
- Delete the tuple inserted by the high user. In this case, we have an integrity problem and an opening for a denial of service attack, because data inserted by a high user are deleted by a low user.
- Reject the low user insertion. In this case, we have an inference channel since the low user knows now that there is a defined mission under code 0XX.

This problem can be resolved using Polyinstantiation meaning that one tuple can appear (be instantiated) many times, with a different classification level each time.

classification	code	name	description
sensitive data	0XX	Skyscraper	Move the artillery to Ohio
nonsensitive data	0XX	Dogscratcher	Move the artillery to Missouri

8.2.3.2. Inference from Functional and Multivalued Dependencies

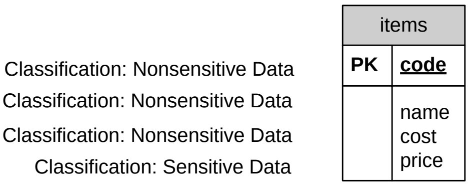
Suppose we have the following relation in a database [1]:


salaries
PK
empname
emprank
empsalary
Classification: Nonsensitive Data
Classification: Nonsensitive Data
Classification: Sensitive Data

Note that data is classified at the attribute level. Suppose further that every employee in the company knows that same rank means same salary. We have an inference channel since a low employee can determine the salary (which is sensitive) of any other employee provided that she knows the rank. The inference channel here stems from the functional dependency emprank → empsalary holding on the relation salaries since the rank (nonsensitive data) determines the salary (sensitive data). To avoid this inference channel, we simply raise the classification level of emprank to sensitive data.

8.2.3.3. Inference from Value Constraints

Suppose we have the following relation in a database [1]:


items
PK
code
name
cost
price
Classification: Nonsensitive Data
Classification: Nonsensitive Data
Classification: Nonsensitive Data
Classification: Sensitive Data

Note that data is classified at the attribute level. Suppose further that the constraint price – cost ≤ 1500 holds on the relation items, and that low users are aware of this constraint. We have an inference channel since a low user can determine probable prices of any item provided that she knows its cost; using the formula price ≤ cost + 1500. The inference channel here stems from the constraint defined over several sensitivity levels. To avoid this inference channel, we simply partition the constraint into two or more constraints each belonging to a single sensitivity level. Suppose that max(cost) = 12000 and max(price) = 17000, then we rewrite the constraint as: price ≤ 12000 + 1500, cost ≥ 17000 – 1500.

8.3. Techniques for Dealing with Inference Channels

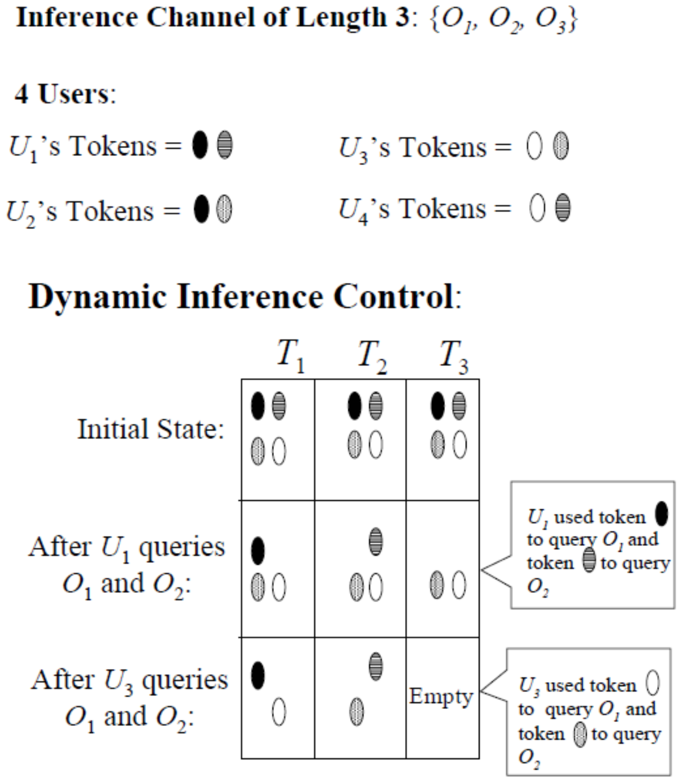
Techniques presented so far for dealing with inference channels fall in one of two categories; the first one employs semantic data modeling while the second one employs query analysis [1]. Semantic data modeling techniques basically search an entire database for illegal information flow, and almost all of them suffer of high false positive rate (identifying an inference channel when it is not inference channel), or high false negative rate (missing an inference channel when it is an inference channel). A high false positive rate means that these techniques are less trustworthy while a high false negative rate means that inference attacks are passing through, which may be catastrophic. Query analysis techniques are much more promising, therefore we will summarize them very quickly:

- Mazumdar, Stemple, and Sheard developed a theorem prover to evaluate transaction security when the transaction is compiled. Their technique helps a database designer in reconsidering the design of a database.
- In the Lock Data Views (LDV) project, classification constraints are defined on data according to the level of information that can be inferred from the data. When a user submit a query to the

system, query result is checked against the classification constraints, updated if necessary, and then returned to the user. It is also possible to save query history and use it with this technique to update classification constraints.

- Thuraisingham described a technique that evaluates a query, and if it leads to illegal inference, the query is modified to prevent the inference. The problem with this techniques is that it needs very careful implementation since a user can monitor the difference in the way the system responds to legal and illegal queries in order to perform inference!
- Probably, Staddon [4] is the first to present a dynamic query analysis technique that does not largely slow down query processing time. In Staddon's technique, query processing time depends only on the length of the inference channel (not on the length of the query history). Staddon demonstrated that her technique provides two additional features:
  1) C-collusion resistance meaning that a coalition of c users cannot together query all the objects in an inference channel (we call c the degree of collusion resistance).
  2) Crowd control meaning that even if a coalition of users have queried all but one object in an inference channel, none of them will be able to query the remaining object.
  Staddon idea is derived from a cryptographic concept called "secure group communication". Her technique assumes that inference channels have already been identified at pre-query processing time. Staddon's technique consists of the following three steps:
  1) Key allocation: allocate a key set for each user based on the maximum length of an inference channel and the desired degree of collusion resistance.
  2) Database initialization: allocate a token set for each object in an inference channel. Staddon suggested that tokens can be generated by encrypting an attribute value or a combination of attribute values under a key. She explained that the purpose of a token is to prove that a user is really in possession of a key.
  3) Dynamic query processing: if a token t ∊ T_i is used to gain access to object O_i, then for every s ≠ i, any token in T_s that was generated using the same key is deleted. Note how token sets change as more queries are processed.

Figure 8.1. Illustration of Staddon's Technique for Dynamic Inference Control


Inference Channel of Length 3: {O1, O2, O3}
4 Users:
U1's Tokens =
U2's Tokens =
U3's Tokens =
U4's Tokens =
Dynamic Inference Control:
T1
T2
T3
Initial State:
After U1 queries O1 and O2:
After U3 queries O1 and O2:
Empty
U1 used token to query O1 and token to query O2
U3 used token to query O1 and token to query O2

Query analysis techniques are favored over semantic data modeling techniques for two main reasons:

1) Evaluating a query dynamically is less expensive than searching an entire database for possible information flow.
2) Data is constantly added to (or updated in) a database. This may open up new inference channels that cannot be identified other than dynamically.

Anti Inference Hub is based on a query analysis technique. For further information, please refer to Chapter 9. Technology behind Anti Inference Hub.

8.4. References

[1] Sushil Jajodia, and Catherine Meadows, "Inference Problems in Multilevel Secure Database Management Systems," in "Information Security: An Integrated Collection of Essays, "IEEE Computer Society Press, Los Alamitos, CA USA, 1995.

[2] Charles P. Pfleeger, and Shari Lawrence Pfleeger, "Security in Computing, Fourth Edition," Prentice Hall, 2006.

[3] Adam, Nabil R., and John C. Wortmann, "Security-Control Methods for Statistical Databases: A Comparative Study, "ACM Computing Surveys, Vol. 21, No. 4, Dec. 1989, pp. 515-556.

[4] Staddon, Jessica, "Dynamic Inference Control," Proceedings of the 8th ACM SIGMOD workshop on Research issues in data mining and knowledge discovery, 2003.

Chapter 9. Technology behind Anti Inference Hub

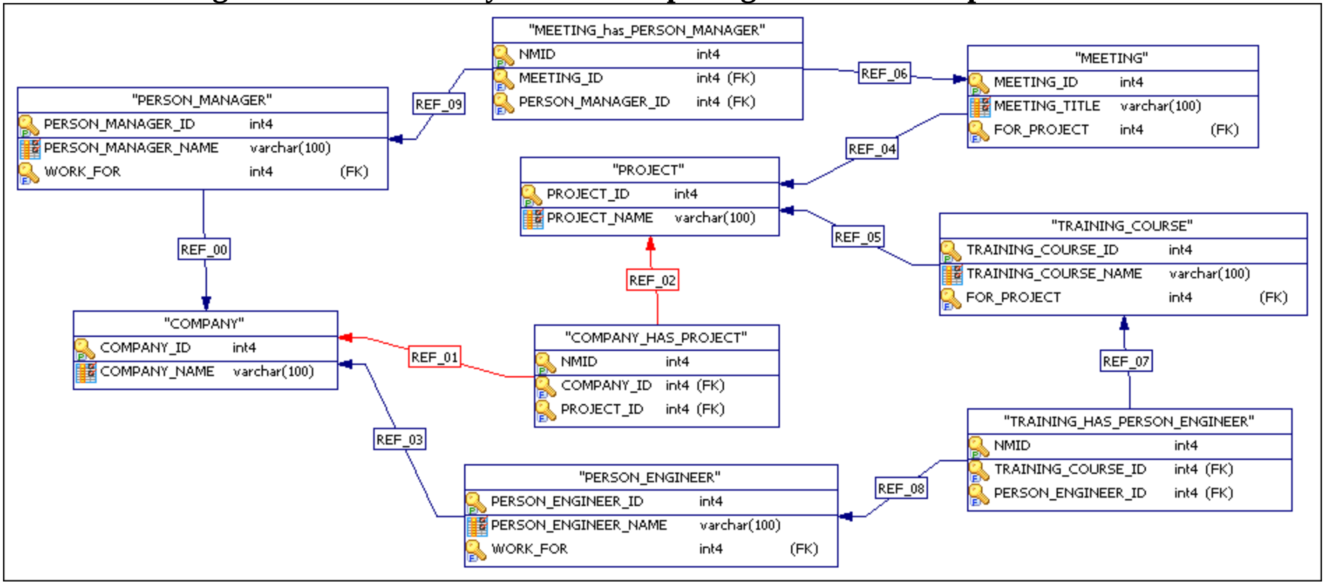
Anti Inference Hub is based on a dynamic query processing technology described in the following paper: X. Chen, R. Wei, "A Dynamic Method for Handling the Inference Problem in Multilevel Secure Databases," Proc. International Conference on Information Technology: Coding and Computing (ITCC'05) - Volume I - Volume 01, 2005, pp. 751-756.

Chapter 10. The Sample Database

10.1. Introduction

Anti Inference Hub comes with a sample database that may be used for testing and understanding purposes.

Figure 10.1. The Entity-Relationship Diagram of the Sample Database



10.2. Location of the Sample Database

The sample database is included in the directory database in Anti Inference Hub binary package.

Table 10.1. Content of the database Directory

File	Description
inferenceoracle.sql	This file contains SQL script needed to create the sample database with an Oracle DBMS.
inferencepostgresqlmysql.sql	This file contains SQL script needed to create the sample database with a PostgreSQL DBMS, or a MySQL DBMS.
createoracledb.bat	This file executes SQL script in the file inferenceoracle.sql in accordance with an Oracle DBMS running under Windows.
createoracledb.sh	This file executes SQL script in the file inferenceoracle.sql in accordance with an Oracle DBMS running under Linux.
createpostgresqldb.bat	This file executes SQL script in the file inferencepostgresqlmysql.sql in accordance with a PostgreSQL DBMS running under Windows.
createpostgresqldb.sh	This file executes SQL script in the file inferencepostgresqlmysql.sql in accordance with a PostgreSQL DBMS running under Linux.
createmysqldb.bat	This file executes SQL script in the file inferencepostgresqlmysql.sql in accordance with a MySQL DBMS running under Windows.

createmysqldb.sh	This file executes SQL script in the file inferencepostgresqlmysql.sql in accordance with a MySQL DBMS running under Linux.
entity-relationship diagram.pdd	This file is the entity-relationship diagram of the sample database built using MicroOLAP Database Designer for PostgreSQL.
entity-relationship diagram.bmp	This file is the entity-relationship diagram of the sample database exported as a bitmap image.

10.3. Creating the Sample Database

To create the sample database, first install a DBMS of choice under an operating system of choice, then navigate to the database directory, and execute the file corresponding to your platform.

10.4. Our Test Environment

The sample database was successfully tested with the following:

- Oracle Database 10g Release 2 (10.2.0.1.0) under Windows XP Service Pack 2 and under Linux Ubuntu.
- PostgreSQL 8.4.2-1 under Windows XP Service Pack 2 and under Linux Ubuntu.
- MySQL 5.1.42 under Windows XP Service Pack 2 and under Linux Ubuntu.

Chapter 11. Anti Inference Hub in Action

11.1. Introduction

In this chapter, we put Anti Inference Hub in action using the sample database created in Chapter 10.

11.2. The Inference Attempt to Address

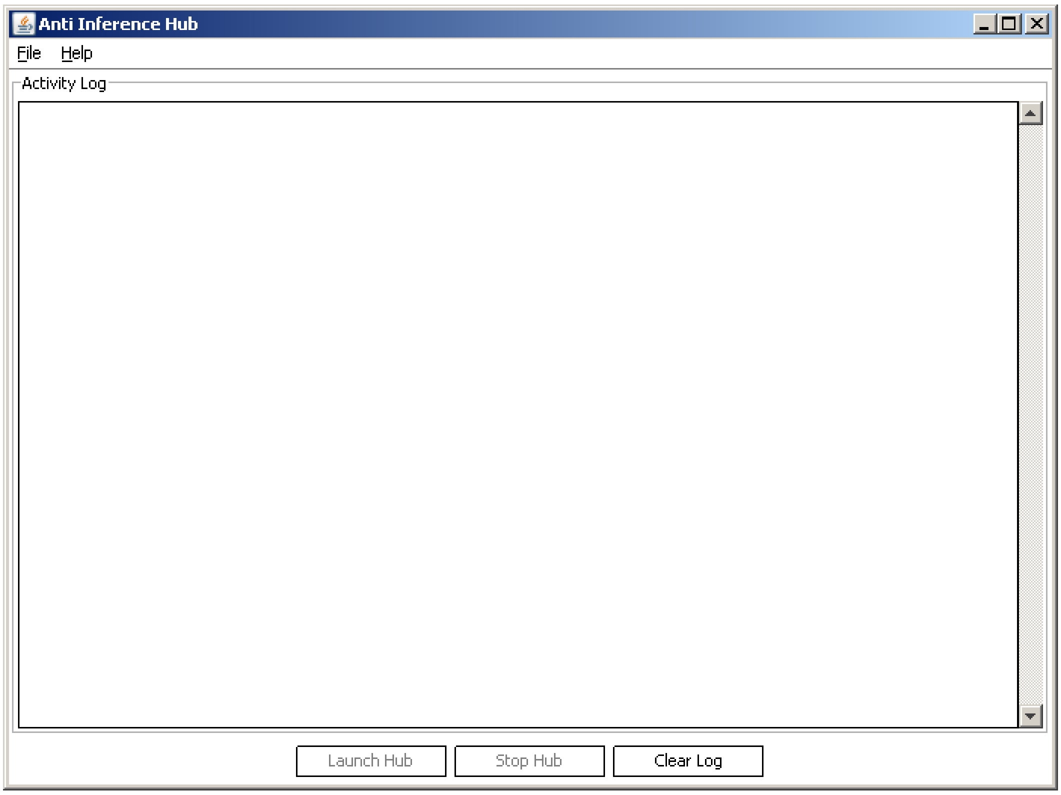
In the sample database, suppose that a low user is able to know the following by executing queries against the database:

- For which COMPANY a PERSON_MANAGER works.
- PERSON_MANAGER attending a MEETING.
- MEETING on a PROJECT.

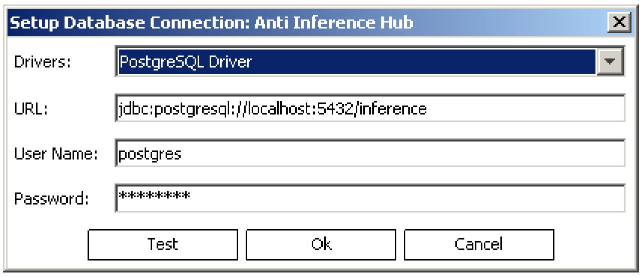
If that was true, then the low user can immediately infer the COMPANY supporting the PROJECT. In the following section, we show how Anti Inference Hub can be used to thwart this inference attempt.

11.3. Thwarting the Inference Attempt

Start the Hub as described in Section 3.2. Starting Anti Inference Hub.



Press Ctrl+D, and establish the connection with the DBMS you are using.



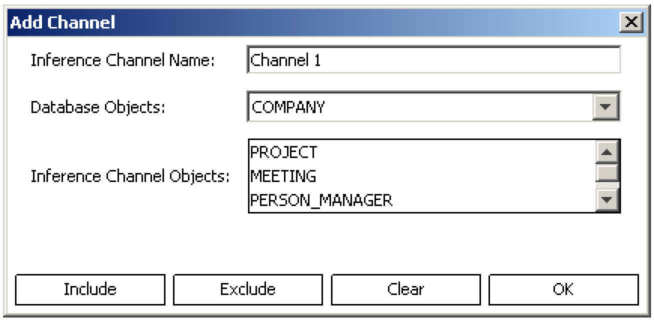
Drivers: PostgreSQL Driver

URL: jdbc:postgresql://localhost:5432/inference

User Name: postgres

Password: *********

Press Ctrl+I, then press the button Add in the box that shows to setup an inference channel named "Channel 1", of length 4, containing the following objects: PROJECT, MEETING, PERSON_MANAGER, and COMPANY.



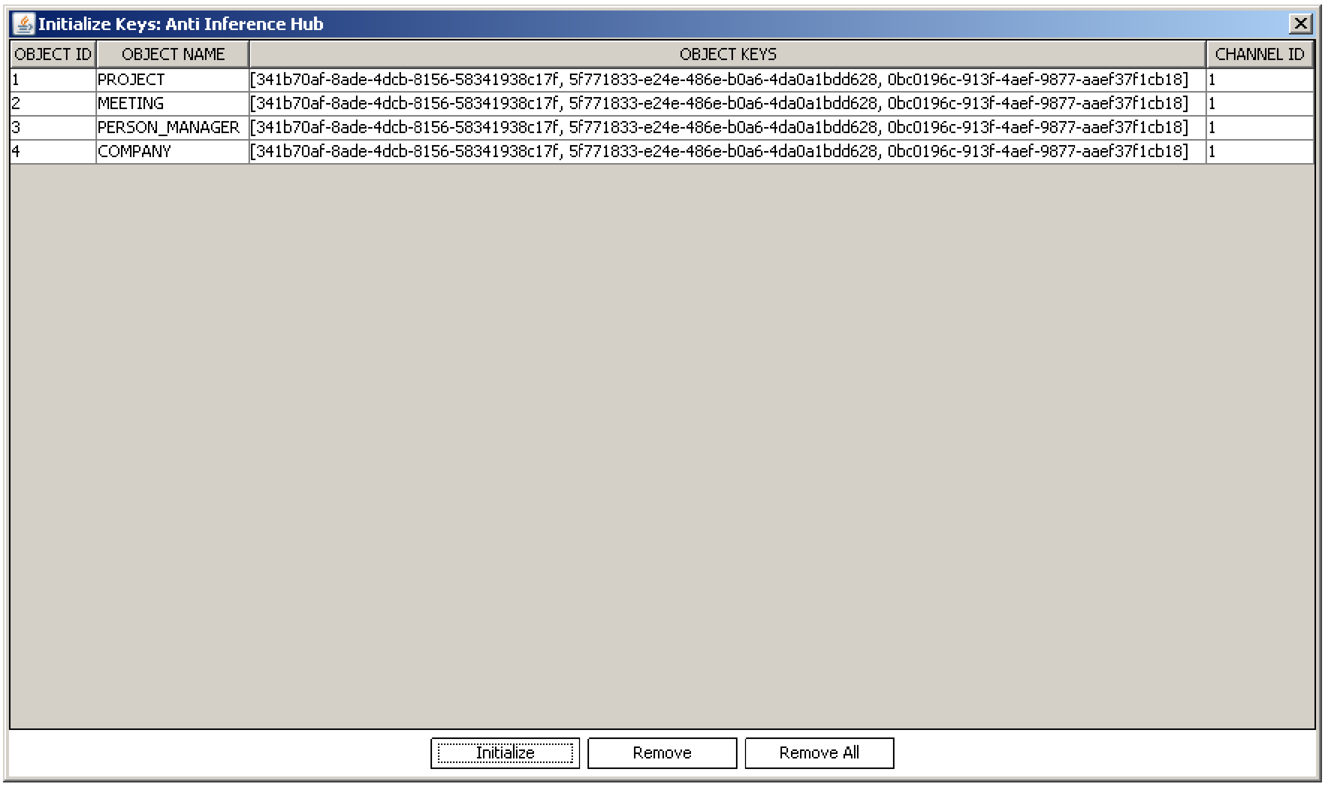
Add Channel

Inference Channel Name: Channel 1

Database Objects: COMPANY

Inference Channel Objects:
PROJECT
MEETING
PERSON_MANAGER

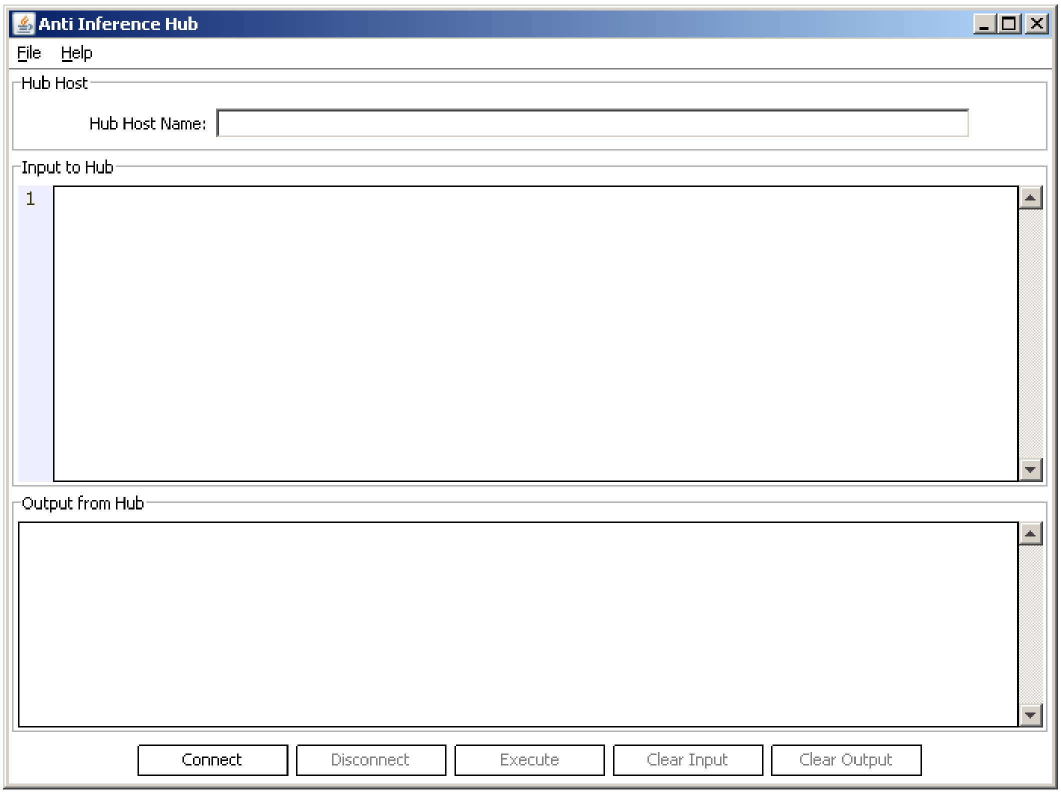
Close the Setup Inference Channels Box, and press Ctrl+K, then press the button Initialize in the box that shows to initialize "Channel 1" objects keys.



Initialize Keys: Anti Inference Hub

OBJECT ID	OBJECT NAME	OBJECT KEYS	CHANNEL ID
1	PROJECT	[341b70af-8ade-4dcb-8156-58341938c17f, 5f771833-e24e-486e-b0a6-4da0a1bdd628, 0bc0196c-913f-4aef-9877-aaef37f1cb18]	1
2	MEETING	[341b70af-8ade-4dcb-8156-58341938c17f, 5f771833-e24e-486e-b0a6-4da0a1bdd628, 0bc0196c-913f-4aef-9877-aaef37f1cb18]	1
3	PERSON_MANAGER	[341b70af-8ade-4dcb-8156-58341938c17f, 5f771833-e24e-486e-b0a6-4da0a1bdd628, 0bc0196c-913f-4aef-9877-aaef37f1cb18]	1
4	COMPANY	[341b70af-8ade-4dcb-8156-58341938c17f, 5f771833-e24e-486e-b0a6-4da0a1bdd628, 0bc0196c-913f-4aef-9877-aaef37f1cb18]	1

Close the Initialize Keys Box, then press the button "Launch Hub" to launch the Hub. Now, start the Hub Client as described in Section 3.2. Starting Anti Inference Hub.



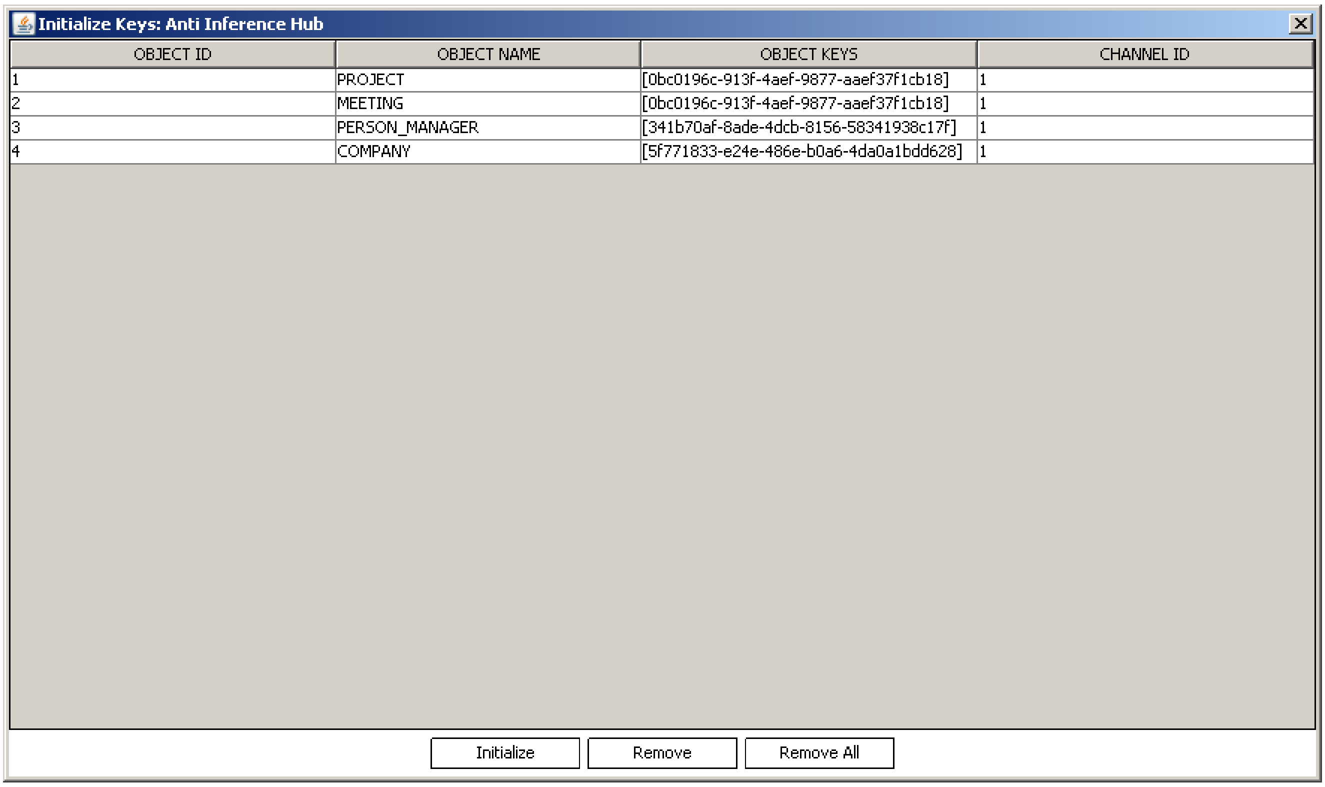
Enter the appropriate Hub Host Name in its field, and connect to the Hub by pressing the Connect button. Act as a low user, and execute the following query in the Hub Client:

```
SELECT PERSON_MANAGER_NAME, COMPANY_NAME
FROM PERSON_MANAGER, COMPANY
WHERE PERSON_MANAGER.WORK_FOR = COMPANY.COMPANY_ID
AND PERSON_MANAGER.PERSON_MANAGER_NAME = 'MANAGER 1';
```

In this query, you are trying to know for which COMPANY a PERSON_MANAGER works. You should receive the following XML output from the Hub:

```
<?xml version="1.0" encoding="UTF-8"?>
<Results>
    <Row>
        <person_manager_name>MANAGER 1</person_manager_name>
        <company_name>COMPANY 1</company_name>
    </Row>
</Results>
```

Which tells you that "MANAGER 1" works for "COMPANY 1". Note the change to "Channel 1" objects keys by opening the Initialize Keys Box in the Hub.



Initialize Keys: Anti Inference Hub

OBJECT ID	OBJECT NAME	OBJECT KEYS	CHANNEL ID
1	PROJECT	[0bc0196c-913f-4aef-9877-aaef37f1cb18]	1
2	MEETING	[0bc0196c-913f-4aef-9877-aaef37f1cb18]	1
3	PERSON_MANAGER	[341b70af-8ade-4dcb-8156-58341938c17f]	1
4	COMPANY	[5f771833-e24e-486e-b0a6-4da0a1bdd628]	1

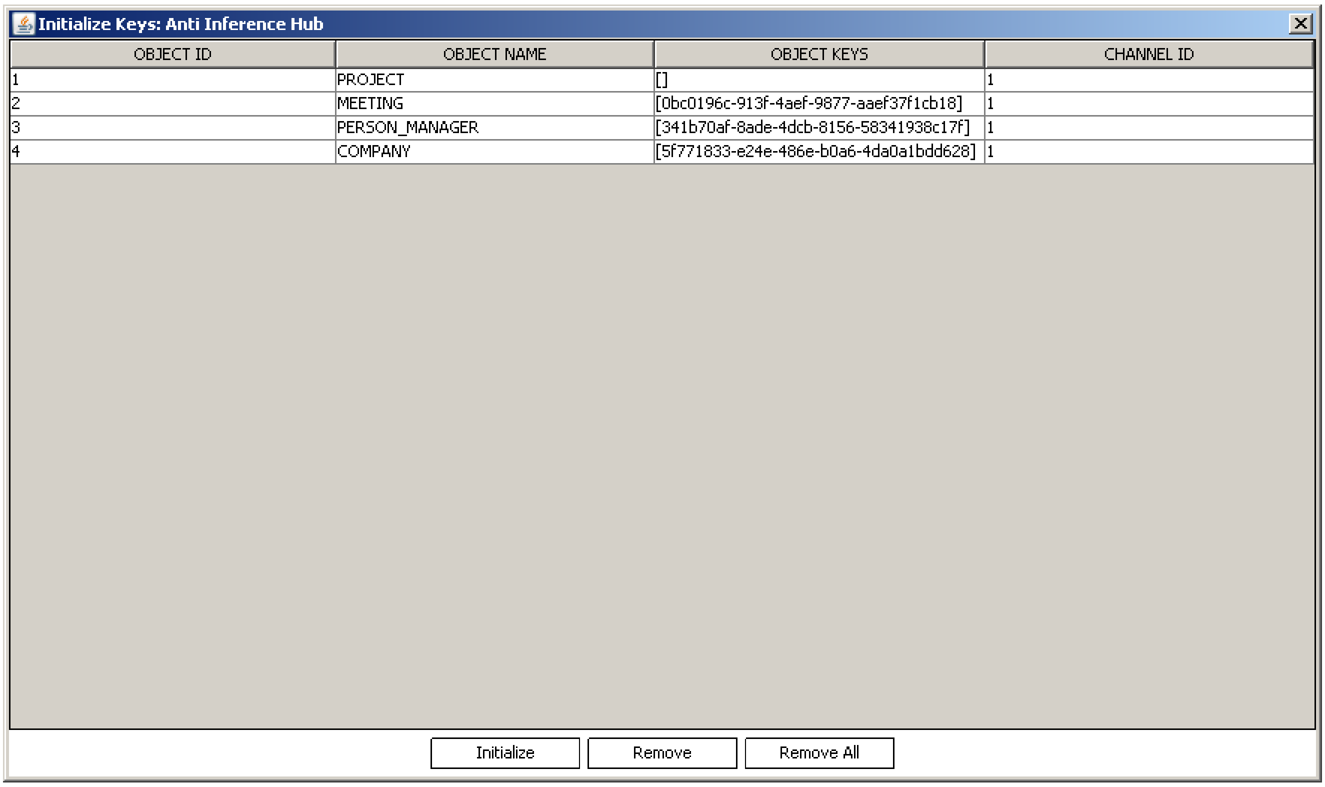
Now, execute the following query in the Hub Client:

```
SELECT MEETING_TITLE, PERSON_MANAGER_NAME
FROM MEETING, MEETING_HAS_PERSON_MANAGER, PERSON_MANAGER
WHERE MEETING.MEETING_ID = MEETING_HAS_PERSON_MANAGER.MEETING_ID
AND PERSON_MANAGER.PERSON_MANAGER_ID =
MEETING_HAS_PERSON_MANAGER.PERSON_MANAGER_ID
AND MEETING.MEETING_TITLE = 'MEETING 1';
```

In this query, you are trying to know the PERSON_MANAGER attending a MEETING. You should receive the following XML output from the Hub:

```
<?xml version="1.0" encoding="UTF-8"?>
<Results>
    <Row>
        <meeting_title>MEETING 1</meeting_title>
        <person_manager_name>MANAGER 1</person_manager_name>
    </Row>
</Results>
```

Which tells you that "MANAGER 1" attends "MEETING 1". Note the change to "Channel 1" objects keys by opening the Initialize Keys Box in the Hub.



Initialize Keys: Anti Inference Hub

OBJECT ID	OBJECT NAME	OBJECT KEYS	CHANNEL ID
1	PROJECT	[]	1
2	MEETING	[0bc0196c-913f-4aef-9877-aaef37f1cb18]	1
3	PERSON_MANAGER	[341b70af-8ade-4dcb-8156-58341938c17f]	1
4	COMPANY	[5f771833-e24e-486e-b0a6-4da0a1bdd628]	1

Note that PROJECT object is now a reserved object in "Channel 1" because it has an empty key set. Now, execute the following query in the Hub Client:

```
SELECT MEETING_TITLE, PROJECT_NAME
FROM MEETING, PROJECT
WHERE MEETING.FOR_PROJECT = PROJECT.PROJECT_ID
AND PROJECT.PROJECT_NAME = 'PROJECT 1';
```

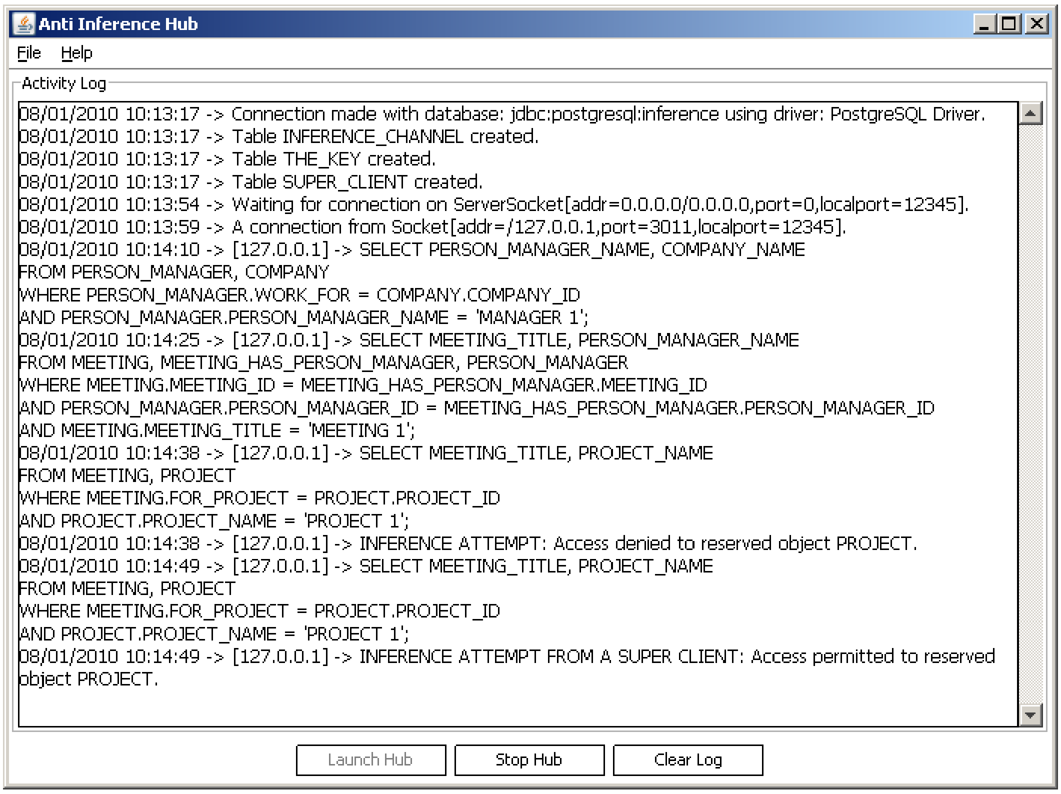
In this query, you are trying to know the MEETING on a PROJECT. If you received a response to this query (actually the response will tell you that "MEETING 1" is on project "PROJECT 1"), then you can immediately infer that "COMPANY 1" is supporting "PROJECT 1"; an inference you base on the results for queries you executed so far. Therefore, result for this query should be blocked. Indeed, it is! The output from the Hub will be:

```
INFERENCE ATTEMPT: Access denied to reserved object PROJECT.
```

Why don't you adjust yourself as a super client and re-execute the same query! Press Ctrl+A, then press the button Add in the box that shows to adjust yourself as a super client, then re-execute the same query. You should receive the following XML output from the Hub:

```
INFERENCE ATTEMPT FROM A SUPER CLIENT: Access permitted to reserved
object PROJECT.
<?xml version="1.0" encoding="UTF-8"?>
<Results>
    <Row>
        <meeting_title>MEETING 1</meeting_title>
        <project_name>PROJECT 1</project_name>
    </Row>
</Results>
```

Thus, as a super client, the Hub allows you to infer. Please note that Anti Inference Hub does not take any IP spoofing attacks into consideration. Securing a network against such attacks falls beyond the purpose of Anti Inference Hub. You may take a look at the Hub log which should be as follow:



```
08/01/2010 10:13:17 -> Connection made with database: jdbc:postgresql:inference using driver: PostgreSQL Driver.
08/01/2010 10:13:17 -> Table INFERENCE_CHANNEL created.
08/01/2010 10:13:17 -> Table THE_KEY created.
08/01/2010 10:13:17 -> Table SUPER_CLIENT created.
08/01/2010 10:13:54 -> Waiting for connection on ServerSocket[addr=0.0.0.0/0.0.0.0,port=0,localport=12345].
08/01/2010 10:13:59 -> A connection from Socket[addr=/127.0.0.1,port=3011,localport=12345].
08/01/2010 10:14:10 -> [127.0.0.1] -> SELECT PERSON_MANAGER_NAME, COMPANY_NAME
FROM PERSON_MANAGER, COMPANY
WHERE PERSON_MANAGER.WORK_FOR = COMPANY.COMPANY_ID
AND PERSON_MANAGER.PERSON_MANAGER_NAME = 'MANAGER 1';
08/01/2010 10:14:25 -> [127.0.0.1] -> SELECT MEETING_TITLE, PERSON_MANAGER_NAME
FROM MEETING, MEETING_HAS_PERSON_MANAGER, PERSON_MANAGER
WHERE MEETING.MEETING_ID = MEETING_HAS_PERSON_MANAGER.MEETING_ID
AND PERSON_MANAGER.PERSON_MANAGER_ID = MEETING_HAS_PERSON_MANAGER.PERSON_MANAGER_ID
AND MEETING.MEETING_TITLE = 'MEETING 1';
08/01/2010 10:14:38 -> [127.0.0.1] -> SELECT MEETING_TITLE, PROJECT_NAME
FROM MEETING, PROJECT
WHERE MEETING.FOR_PROJECT = PROJECT.PROJECT_ID
AND PROJECT.PROJECT_NAME = 'PROJECT 1';
08/01/2010 10:14:38 -> [127.0.0.1] -> INFERENCE ATTEMPT: Access denied to reserved object PROJECT.
08/01/2010 10:14:49 -> [127.0.0.1] -> SELECT MEETING_TITLE, PROJECT_NAME
FROM MEETING, PROJECT
WHERE MEETING.FOR_PROJECT = PROJECT.PROJECT_ID
AND PROJECT.PROJECT_NAME = 'PROJECT 1';
08/01/2010 10:14:49 -> [127.0.0.1] -> INFERENCE ATTEMPT FROM A SUPER CLIENT: Access permitted to reserved
object PROJECT.
```

When done, disconnect from the Hub, and stop it.

Appendix A. Anti Inference Hub License

The Apache License explains all the things that you are allowed to do with Anti Inference Hub code and documentation.

Appendix B. Anti Inference Hub Code Disclaimer

The author of this software code has used his best efforts in preparing the code. These efforts include the development, research, testing, and optimization of the theories and programs to determine their effectiveness. This software code is not designed or intended for use in the design, construction, operation or maintenance of any nuclear facility. Author disclaims any express or implied warranty of fitness for such uses. The author makes no warranty of any kind, expressed or implied, with regard to this software code or to the documentation accompanying it. In no event shall the author be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption whatsoever) arising out of, the furnishing, performance, or use of this software code, even if advised of the possibilities of such damages.