

# Protecting GoldMine CRM database with DbDefence

*Version 1.1, 26 July 2013*

## Introduction

As the backbone of any digital venture, databases are essential to the running of organizations, whether they be enormous corporate entities, or a small endeavors run from home offices. For this reason, it is extremely important that databases are protected. This white paper will explain how to do that, even if you are unfamiliar with database administration and programming.

GoldMine is a popular Customer Relationship Management solution that is already in use worldwide, often for the storage of sensitive data. With digital crime increasing year on year, encryption has never been more important for protecting your databases, and the data within.

For this white paper, we used the free, thirty day evaluation of GoldMine, which includes demonstration data for testing purposes, and to allow potential customers to try the software without having to enter their own data in order to see how it performs. There is also a free version of DbDefence that is capable of working with databases of up to 200MB in size. On the other hand, the paid version of DbDefence can be trialed for thirty days. Please contact [support@dbdefence.com](mailto:support@dbdefence.com). DbDefence is available for download from their website, [www.dbdefence.com/](http://www.dbdefence.com/). If you are interested in our product, we have a number of different pricing options available, depending on the size of your database. Prices start from \$698 per server.

## Transparent Encryption

If someone were to hack into the server, they would have no problem reading the information in the database, or even stealing it entirely, because the contents would be in a clear, readable format that could be viewed with nothing more than attaching database to SQL Server.

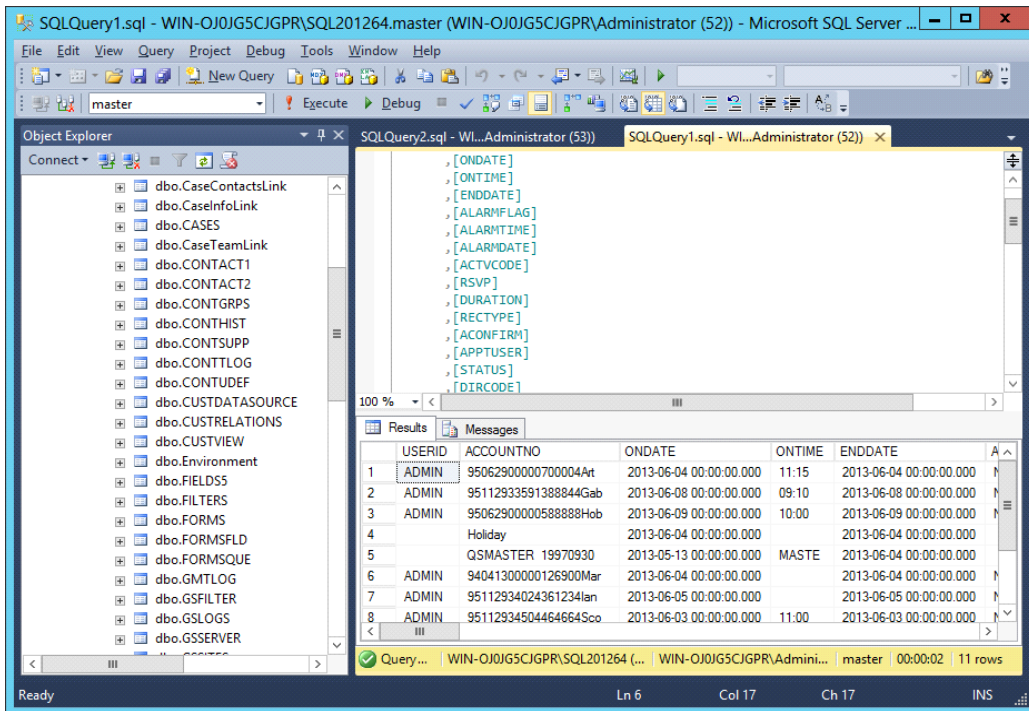


Fig. 1 - How an SQL Server management console looks when it is in the process of querying data. An SQL administrator has access to everything in the database.

DbDefence has the ability to provide transparent encryption features to any instance of an SQL Server—including modern versions—without the need to modify the applications which you use to access said database. This white paper will explain how to use DbDefence to achieve this.

There is a similar feature, called TDE (Transparent Data Encryption) that is available in SQL Server Enterprise Edition, however, the pricing for the Enterprise Edition obviously makes it an unrealistic proposition for smaller companies.

## Installation

To begin with, you will need to install DbDefence [Fig. 2]. If you haven't already, download the thirty day trial from [www.dbdefence.com](http://www.dbdefence.com). DbDefence must be installed on the same computer which SQL Server is operating from. The installation itself is very simple, but we will walk you through the steps.

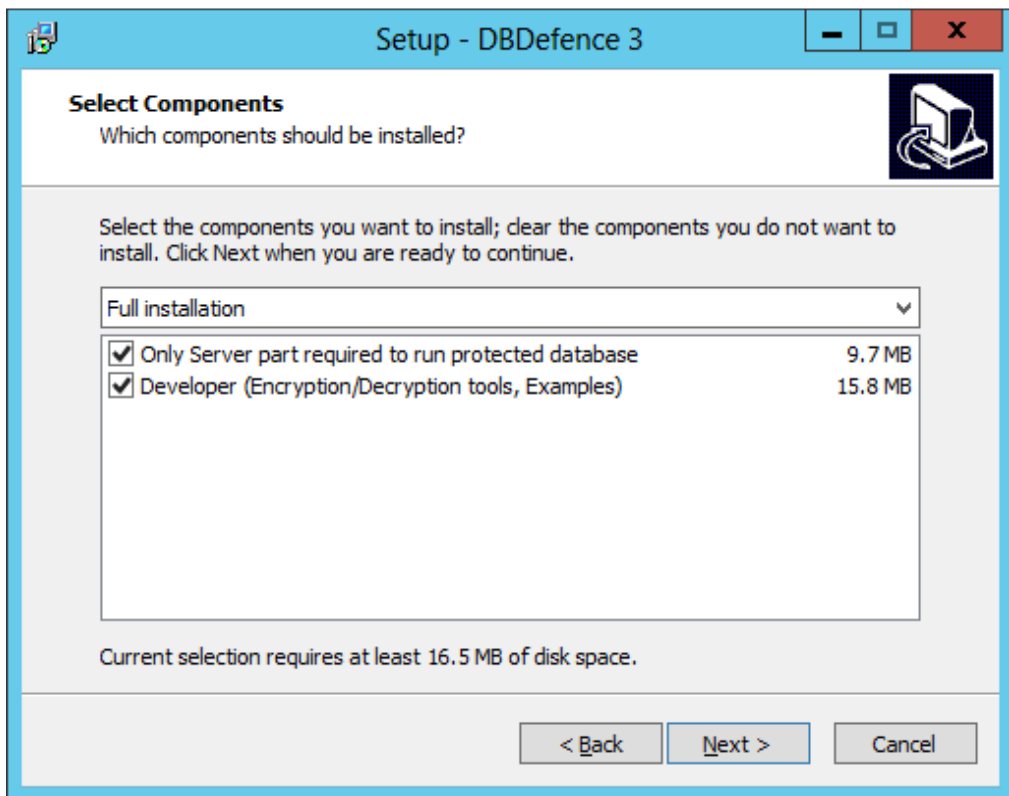


Fig. 2 - DbDefence Installation—Select components screen

The installation will ask you to login to SQL Server [Fig. 3], and will provide you with a list of locally installed SQL Servers to choose from. You only need to login to the SQL Server that you wish to protect. You must login in as admin so full access is granted to DbDefence installer.

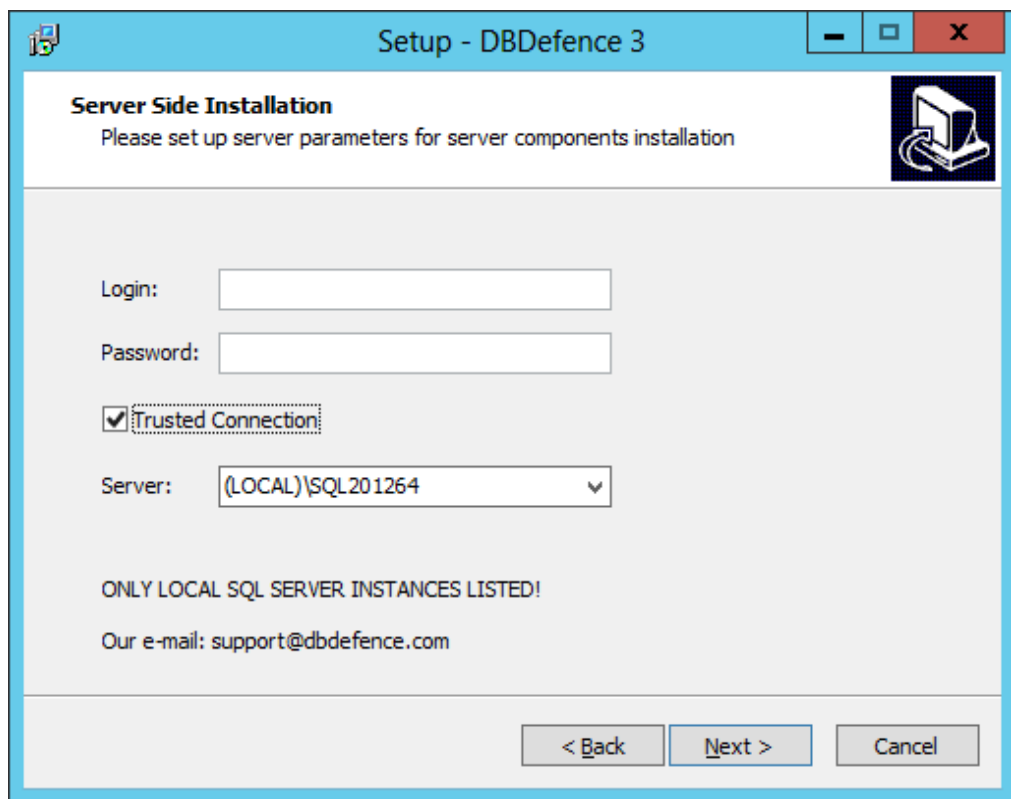


Fig. 3 - DbDefence Installation—Logging in to target SQL Server

After you have logged in to the target SQL Server, there is no further configuration necessary in the installation. Simply click Next, and the installation will proceed. Once it is finished, you will be shown one final screen [Fig. 4], informing you that installation was successful (this is also where it will inform you if the installation failed). Simply click Finish in order to close the installer.

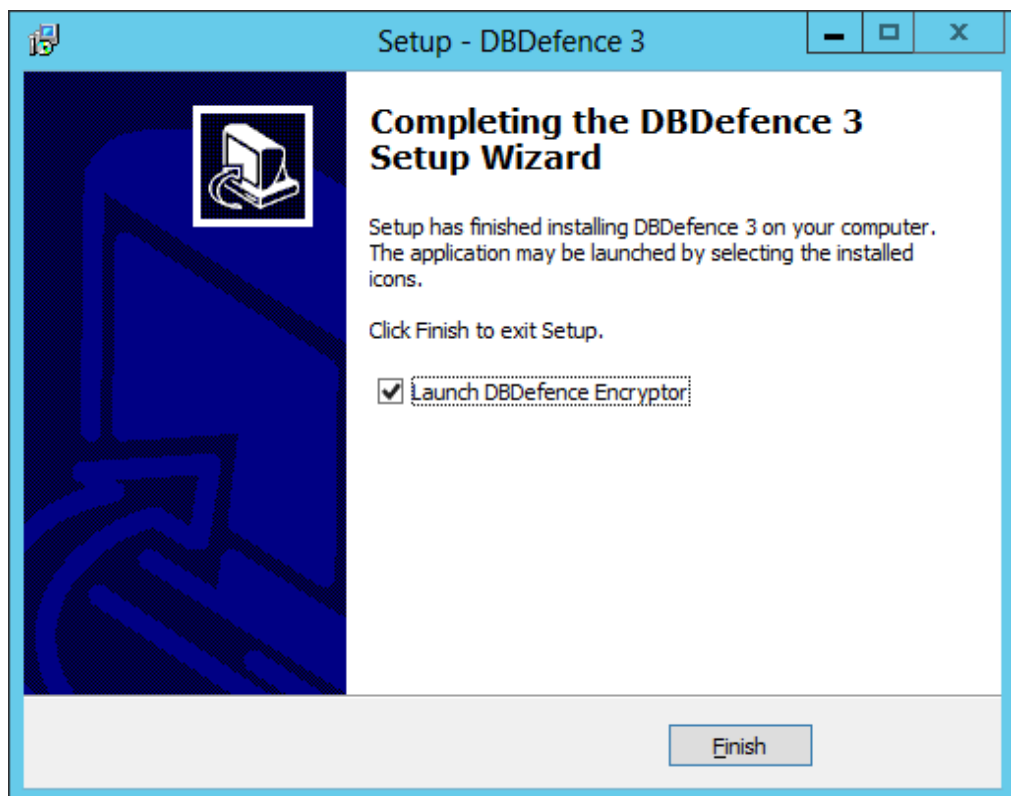


Fig. 4 - DbDefence Installation—Final screen. You can choose to launch DbDefence immediately after the installer closes by checking the box.

Once the installation is finished, you will need to run DbDefence Encryptor. If you selected “Launch DbDefence Encryptor” [Fig. 4] in the last screen of the installation process, the software will launch itself once the installer is closed. Otherwise, simply find the shortcut in your Start menu.

### Encryption and Decryption

Once DbDefence Encryptor is running, connect to the instance of the SQL Server that you wish to work with, and you will be presented with some information [Fig. 5]. Remember, the free version of DbDefence Encryptor only supports databases up to 200MB in size, but, as this white paper is intended to help you get to grips with the process, it would be advisable to use a small sample database to begin with. The process is simple, and does not take long, so it will be easy for you to repeat the steps below on the database you wish to use.

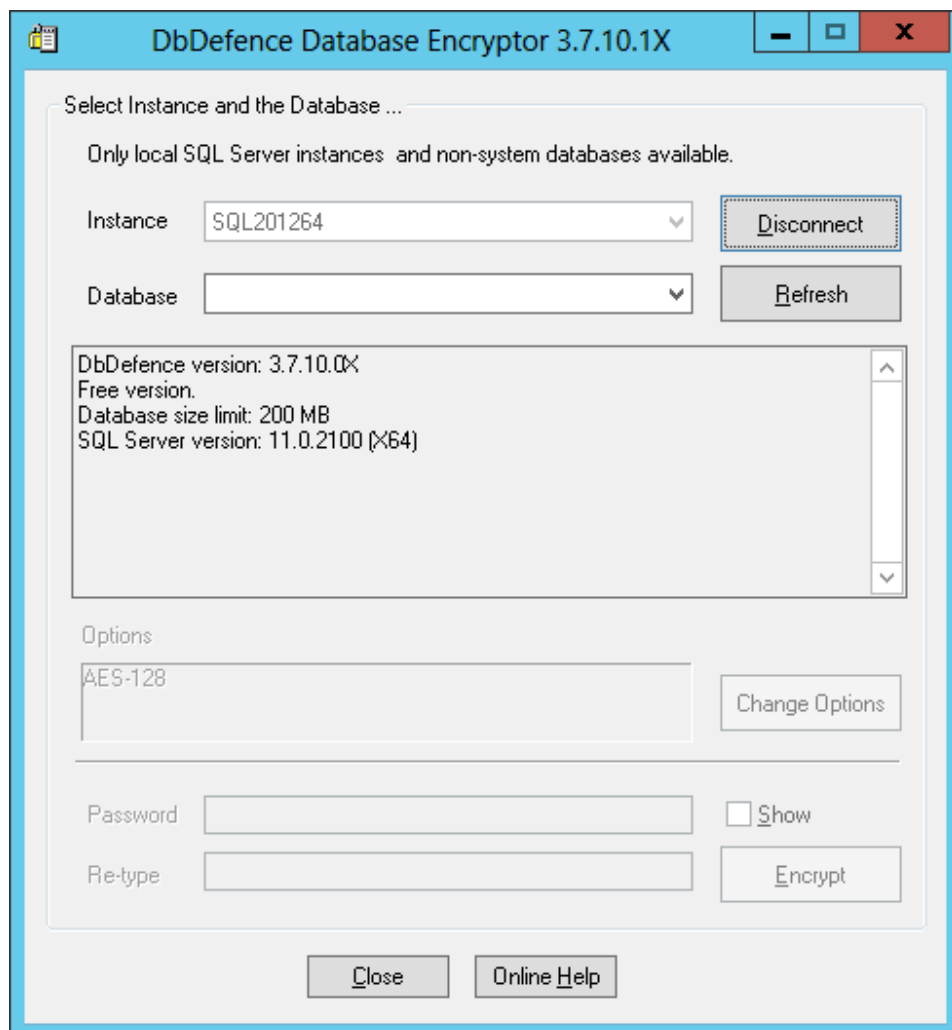


Fig. 5 - DbDefence Encryptor select instance screen. Here, you will select the instance of the database you wish to work with. (Database must be smaller than 200MB for free version)

Once you have selected the appropriate instance of SQL Server that you wish to encrypt, you will then need to select the database from the appropriate drop down box, which is located just below the drop down box where you selected the SQL Server instance [Fig. 6]. The default encryption method is AES 128-bit, AES being the USA encryption standard, and in wide use. It stands for Advanced Encryption Standard. For most cases, 128-bit encryption is sufficient for the task. However, if stronger encryption is necessary, 256-bit is also supported. You can change the encryption by clicking the Change Options button, which is located just below the information box [Fig. 6]. This will load up the Encryption Options dialog window. At the top of the window are two bullet point options (128-bit and 256-bit). Simply select the one you wish to use, and click OK. There are more options in this dialog window, but we will cover them later.

Finally, you will need to enter a password. This password is the key to accessing the database once it has been encrypted, so it is important that you memorize your password—there is no “Reset Password” feature on encryption. It is also important that you enter a strong password, not only for the purposes of security, but also because SQL Server policies (depends on operating system) may refuse to accept passwords that

are deemed too weak. A strong password should contain both upper and lower case characters, as well as at least one number and one symbol.

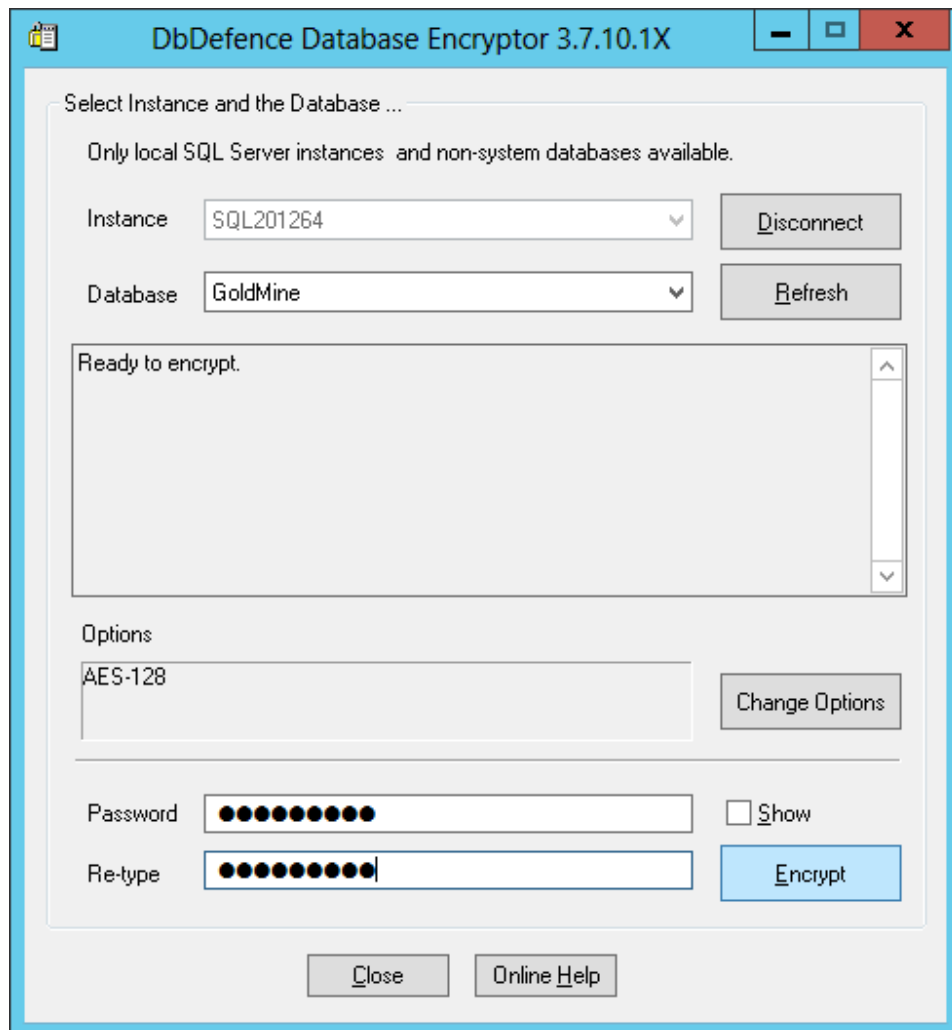


Fig. 6 - The select instance screen again, this time with all the information filled out.

These days, encryption routines, which are optimized for modern processors, are handled particularly well by modern machines, meaning that they are very fast on new computers. A database of 40GB, for example, can be encrypted in around seven minutes. Once finished, the database will be completely encrypted including log file. From this point on, access to the database without the encryption password is almost impossible (it is possible to break encryption, but very, very difficult, and requires a lot of time and resources).

Through the same dialog window you used to encrypt your database, when an already-encrypted database has been selected, you will be given the option to decrypt said database, which you can easily do. Simply enter the password you set in the previous step, and click Decrypt [Fig. 7]. DbDefence will then decrypt the database, restoring it to its previous state.

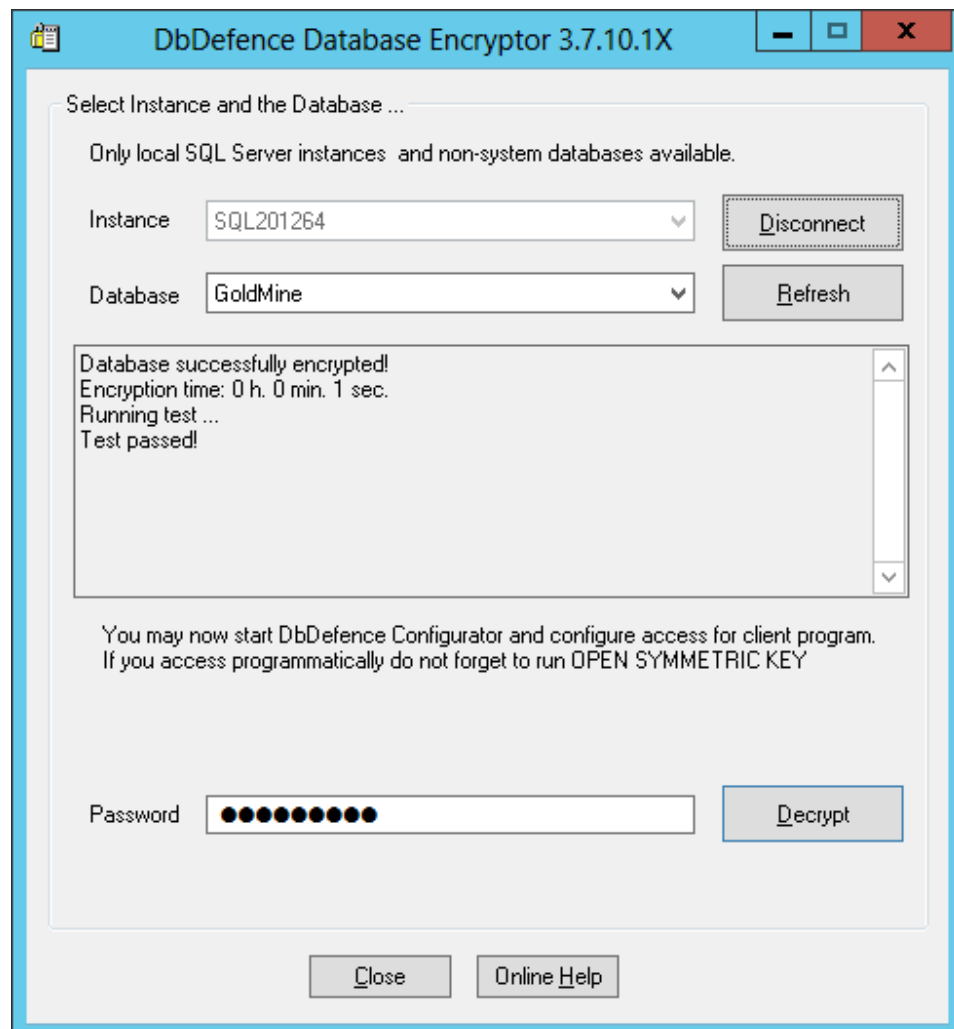


Fig. 7 - DbDefence showing that the database has been successfully encrypted.

## Using an Encrypted and Protected Database

Now that your database is encrypted, access is restricted to only those applications and services that can provide the correct password. This includes any application trying to access the database, and even the database administrators. Opening the raw database file itself will reveal only garbled, indecipherable code. Attempting to view the contents of the database without the correct password will result in an error message being displayed [Fig 8].



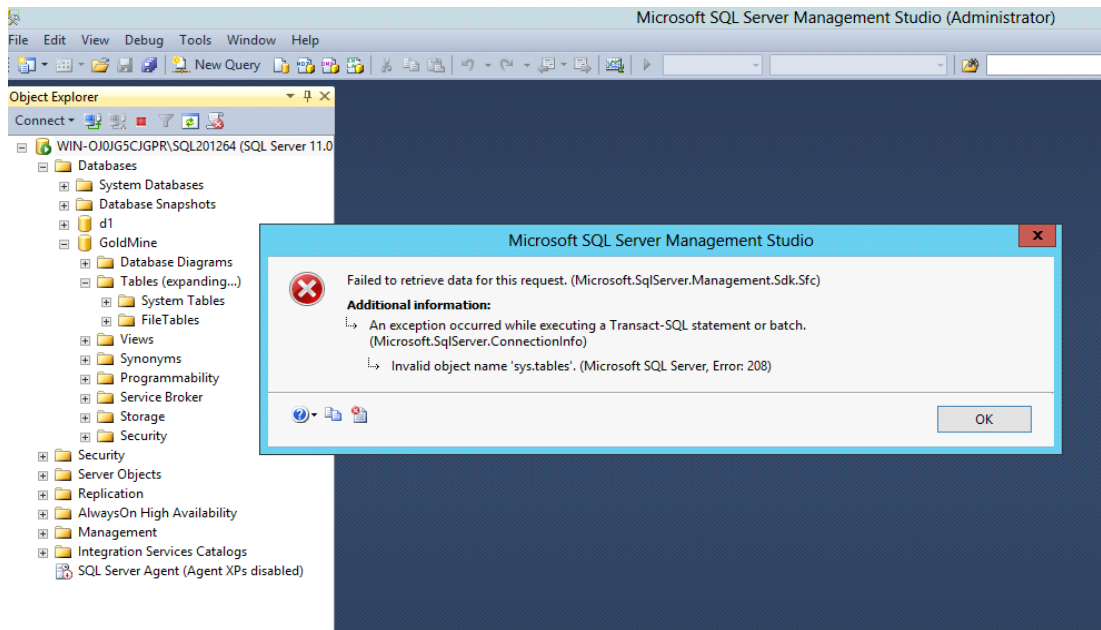


Fig. 8 – An error generated when attempting to view the contents of an encrypted database without the correct password.

Once your database has been encrypted, if you start up GoldMine as usual you should receive an error telling you that the login was unsuccessful due to the USERS table being empty [Fig. 9]. This is because, without the encryption password, GoldMine cannot make sense of the data it is seeing.



Fig. 9 – GoldMine error message generated when attempting to log in after encrypting the database.

The database is still in a fully functional state, however, and authorized apps will still work correctly with the database, but DbDefence needs to know which applications are allowed, so that it can decrypt the database for them.

We will now need to “approve” the GoldMine application in order to access the data within the encrypted database. If you are a programmer, doing so would require nothing more than a single SQL statement in your code. However, as this white paper has been written for non-programmers likely using an application (like GoldMine), we will assume you are not a programmer.

**Update: There is a newer method accessing encrypted database. You may allow access to encrypted database for certain SQL Server logins automatically.**

**Read more at [http:// www.dbdefence.com/support/dbdefence-documentation/automatic-keys](http://www.dbdefence.com/support/dbdefence-documentation/automatic-keys)**

### Password Prompt

First, start DbDefence Configuration. In the left hand panel, find *Client Program Configuration*, and right click on it. In the contextual menu that pops up, select “Add Program” [Fig. 10], from here, any desktop program can be added, but we will use GoldMine for our example.

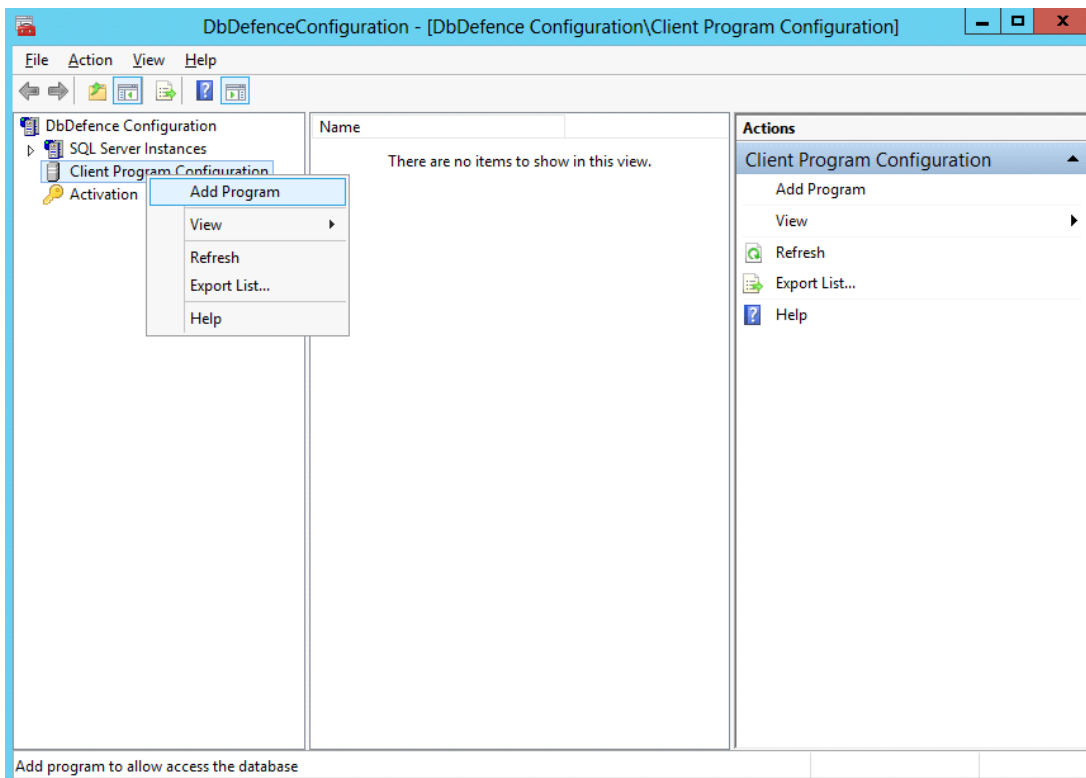


Fig. 10 – DbDefence Configuration window. Right click on “Client Program Configuration”, and then, “Add Program”

In the *Open* window, navigate to the GoldMine installation folder. This is where the GoldMine program files have been installed to. The address will usually look something like “C:\Program Files\GoldMine\”. Once there, you should see a file called *gmw*. Select that file. This is the main executable application file for GoldMine. Once it is selected, click Open.

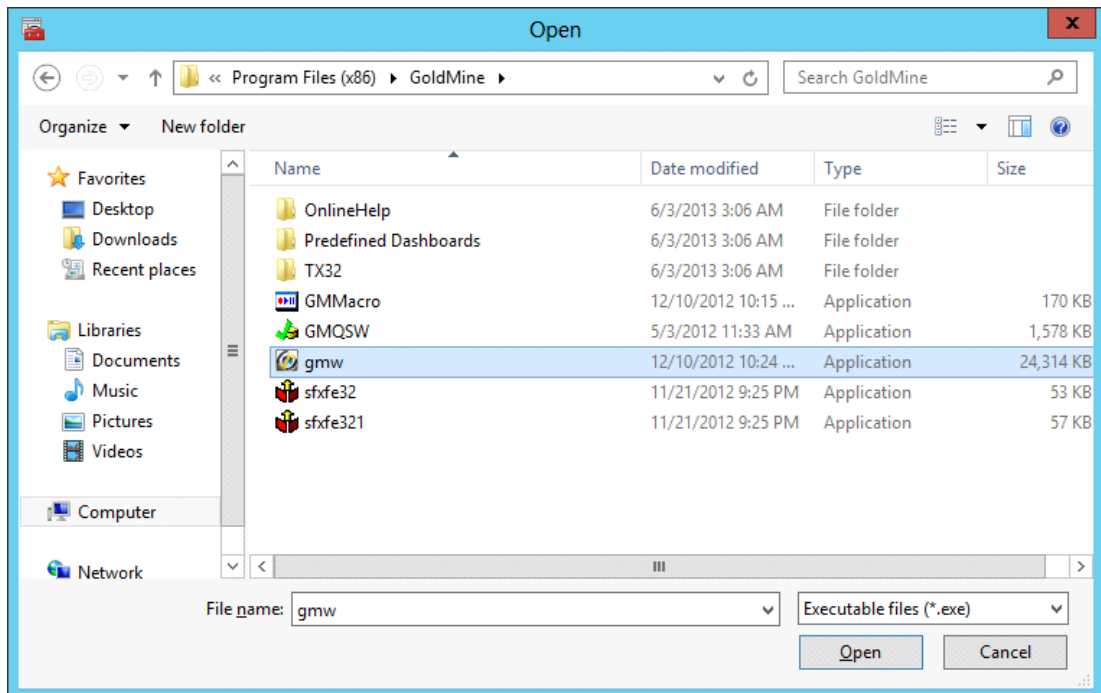


Fig. 11 – The *Open* window, with the GoldMine executable selected.

Once you have selected and opened *gmw.exe*, you will be presented with a new dialog panel within the DbDefence Configuration window [Fig. 12]. From here, you can set things up so that GoldMine automatically enters the encryption password whenever it attempts to access the database by entering the database name and password into the appropriate fields, a procedure we will explain later in this paper. However, for the purposes of this exercise, ignore that, and simply check the “Show Password Dialog” box, and then press *Start !* to launch GoldMine.

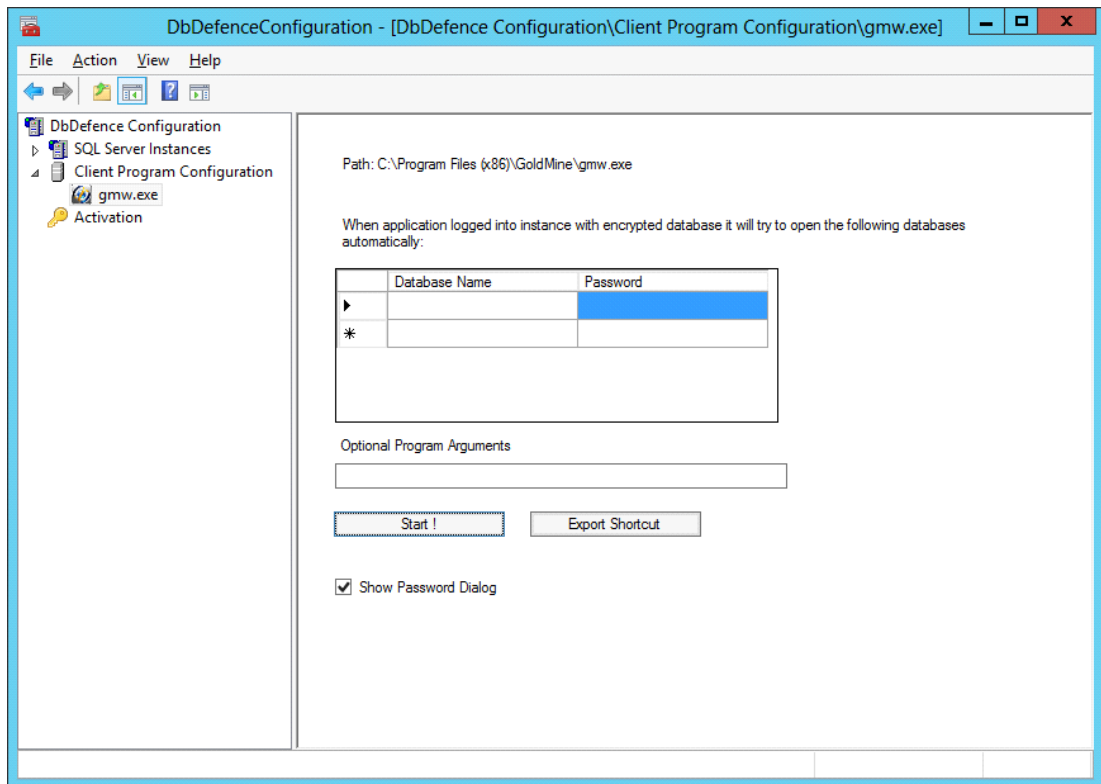


Fig. 12 – Choose how GoldMine will access the encrypted database – automatically, or prompted.

Now, when GoldMine is launched, and it attempts to make a connection to your encrypted database, a password request dialog box from DbDefence will pop up [Fig. 13]. Make sure the correct database is selected to unlock, and enter the password in the field below the drop down box.

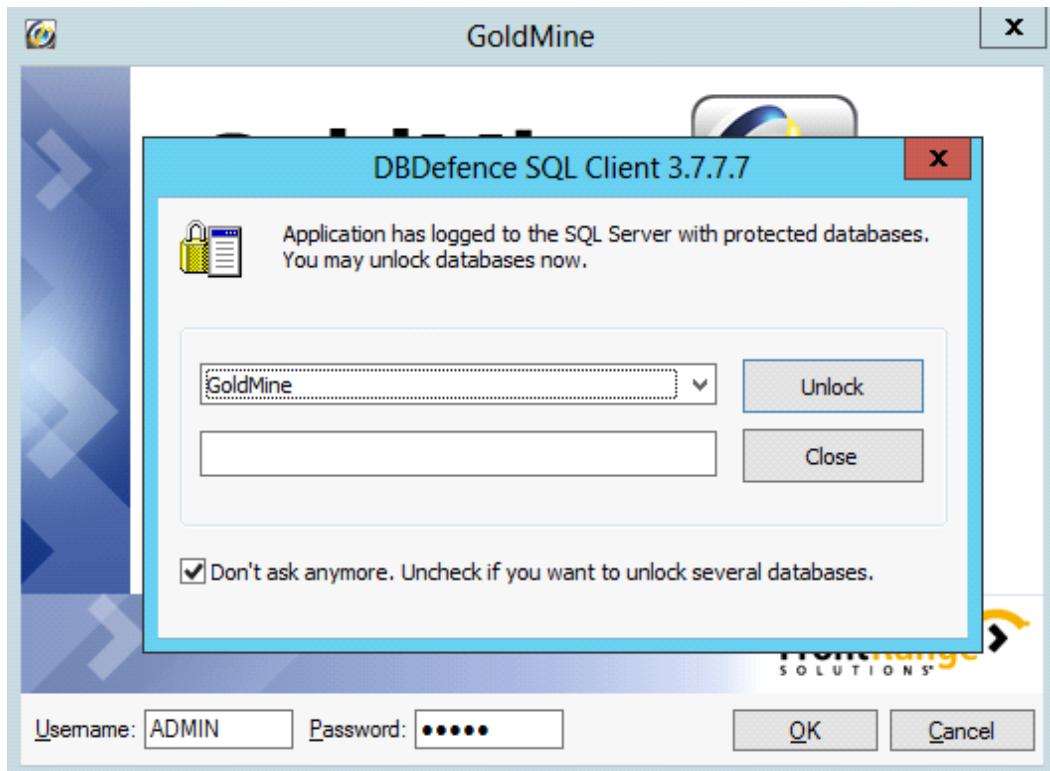


Fig. 13 – Once GoldMine has been approved, the DbDefence password dialog will popup when GoldMine attempts to access the database.

With the correct database selected, and the correct password entered, click Unlock. Now that GoldMine is authorized, it will load the database as normal, showing contents of the database just as it would if it were working with an unencrypted database [Fig. 14]. Meanwhile, if we try to access the database at the same time with SQL Server Management Console, we will receive an access denied error [Fig. 15], because SSMS has not been cleared in DbDefence to access the database.

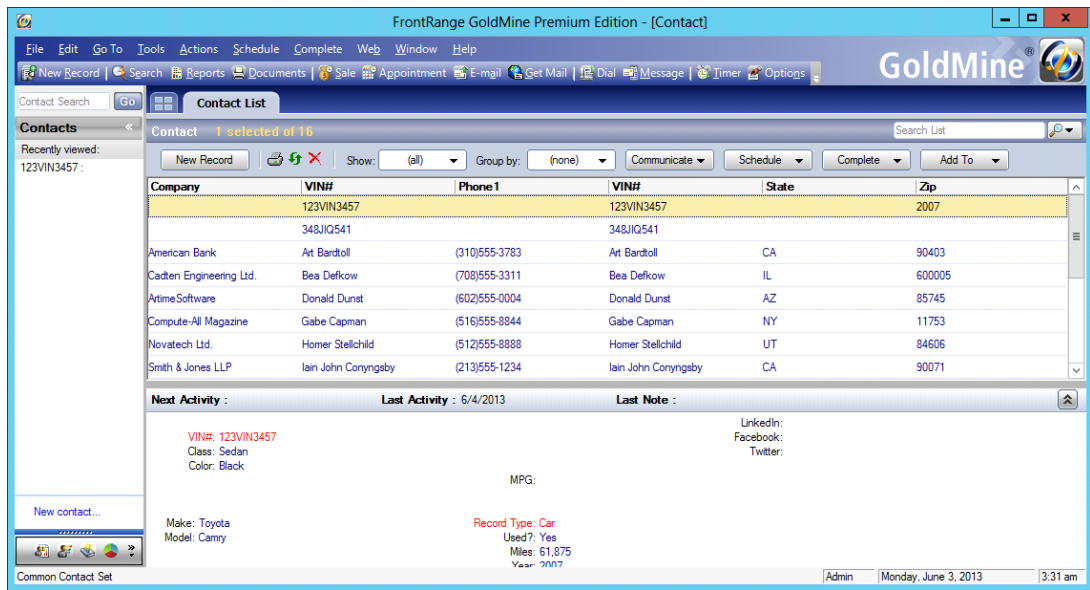


Fig. 14 - GoldMine working as normal with the now encrypted database.

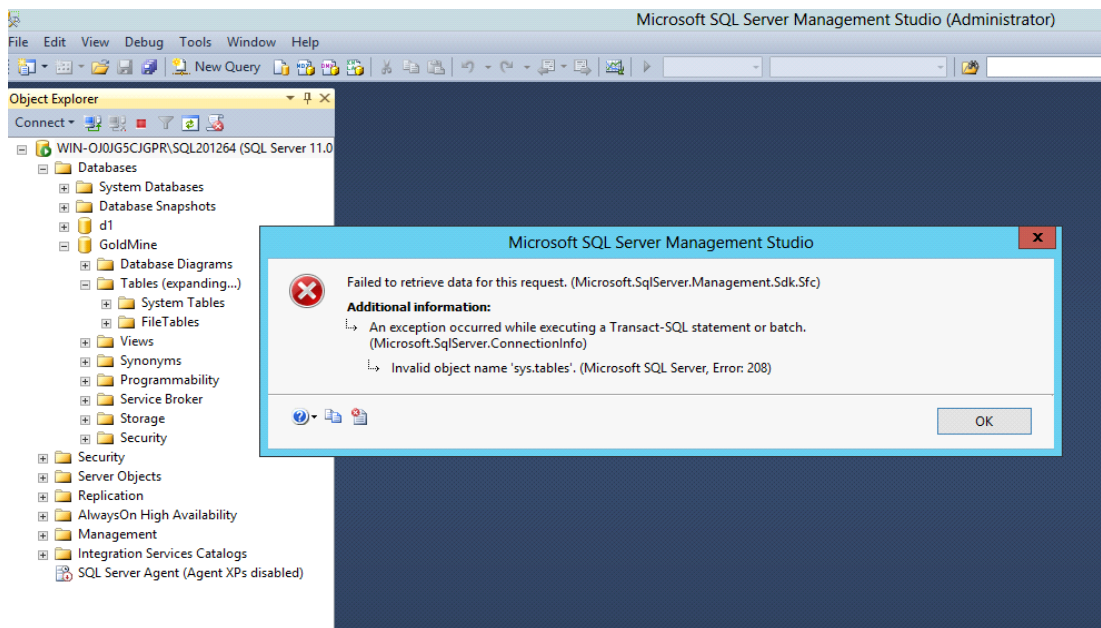


Fig. 15 - At the same time as GoldMine is accessing the encrypted database, SQL DBA cannot.

## Automatic Login

If you're working from a secure system, you may find it an unnecessary pain to have to enter your password every time you wish to access the database. Fortunately, as mentioned earlier, you can set DbDefence to recognize certain applications, and have the password automatically entered upon any attempts by it to access the encrypted database.

To do this, you will need to go back to DbDefence Configuration, and, in the left hand panel, find Client Program Configuration. *gmw.exe* should be underneath it. If you don't see *gmw.exe*, click on Client Program Configuration to expand it. You should now be looking at the same screen you were viewing at the start of this section, only, this time, you do not need to find and add *gmw.exe* to the list, because you have already done it.

This time, we are going to add the login credentials into the appropriate fields of the table in the centre of the right hand panel [Fig. 16]. You will need to enter the database name (probably GoldMine), and your encryption password. In cases where you have multiple encrypted databases that you wish to allow access to, you can enter multiple database names and passwords in this box. Once you have done that, uncheck the "Show Password Dialog" box so that you are not prompted to enter a password the next time you try and access the database through GoldMine.

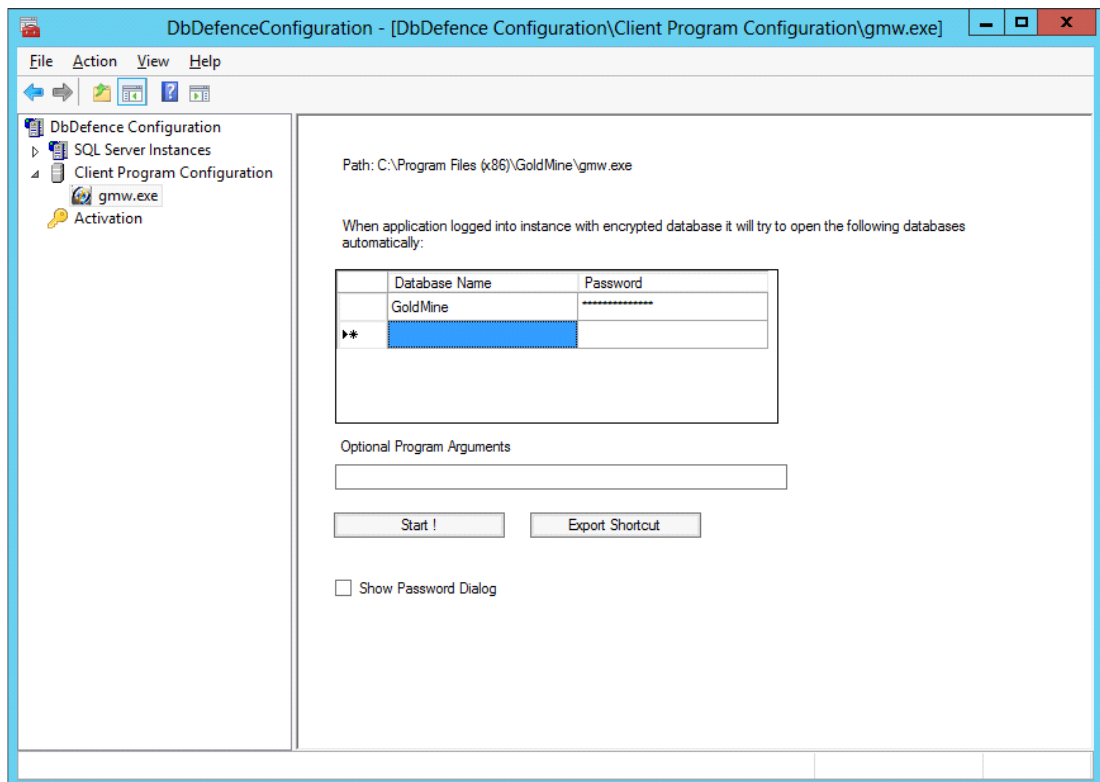


Fig. 16 - Entering the password so that approved applications can automatically gain access to the database.

Now, if you open GoldMine by clicking *Start !* in the configuration window, it will start as usual, without the need to enter the encryption password to gain access.

## Creating a Shortcut

If you would prefer not to have to open DbDefence Configuration every time you want to access your encrypted database, you can create a shortcut [Fig. 17] which can then be placed on your desktop. At the moment, doing this will expose the password, but this will feature will be improved in future versions. It is also important to ensure that the location of the shortcut is safe with regards to other users. If you place the shortcut to your database on the desktop of a shared computer which is also used by those who should *not* have access to the database, they can simply click on the shortcut to gain access. Ensure your shortcut is, at the very least, in your own password-protected user area, if not on a secure system that only you have access to.

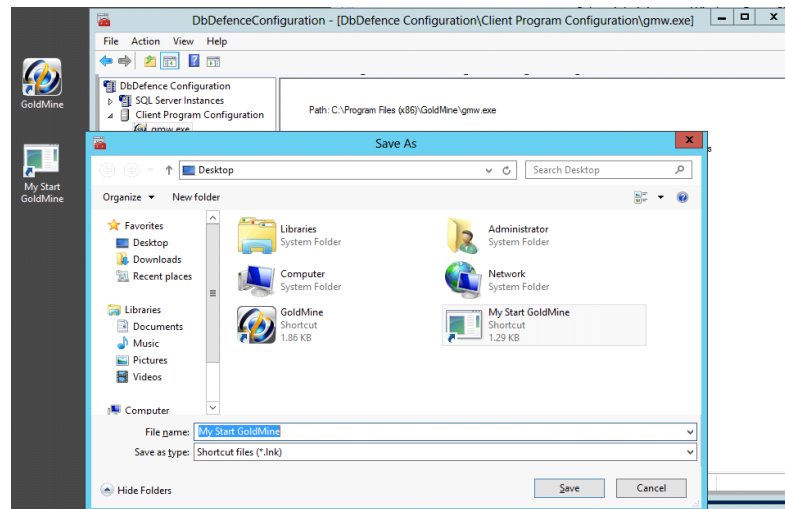


Fig 17 - Exporting a shortcut to the desktop so as to avoid having to open DbDefence every time you wish to access your encrypted database.

Accessing your encrypted database this way is the same as accessing it through DbDefence Configuration, without the hassle of opening DbDefence Configuration. The process is completely transparent to the application (GoldMine, in this case).

Adding an application besides GoldMine is just as simple, all you need to know is the name and location of the main executable file. Then repeat the steps outlined above, replacing *gmw.exe* with the executable for the application you wish to add. SQL Server Management Studio from SQL Server 2012 usually located at "C:\Program Files (x86)\Microsoft SQL Server\110\Tools\Binn\ManagementStudio\ssms.exe" SQL Server Management Studio from SQL Server 2008 at "C:\Program Files (x86)\Microsoft SQL Server\100\Tools\Binn\VSShell\Common7\IDE\ssms.exe"

## Encrypting database without protection

There are occasions when you may want to encrypt database files, but do not want to prevent access by or through any application or web service. Fortunately, there is a feature in DbDefence that can achieve this.

Using this feature, you can specify which parts of the database are permitted to be accessed without the encryption password. This is done before the database is encrypted, through the same dialog window mentioned near the start of this paper, which you would use to choose between 128-bit and 256-bit encryption.



From the instance selection screen [Fig. 6], click on the *Change Options* button. This will present you with the Encryption Options dialog [Fig. 18]. In the Encryption Options dialog, check the “*Allow Access without Encryption Password*” box.

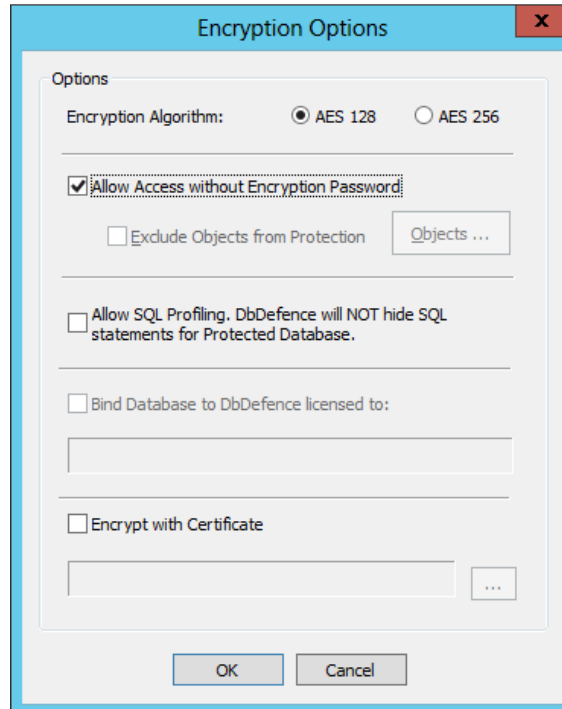


Fig. 18 - The Encryption Options dialog.

Once you have encrypted your database with this option in place, it will be possible for any application to access the database without the encryption password. The database file, however, *will* be encrypted, meaning that, were someone (a hacker, for example) to try and view the contents of the database by looking at the raw file, they would be unsuccessful without the encryption password.

## Conclusion

The importance of protecting your data cannot be understated in this day and age, where the value of information in all forms is increasing exponentially, but that doesn't mean it should be difficult to protect your valuable information. With our software, you can ensure that your (or your company's) databases are fully secure, without the need of years of technical experience and/or programming ability.

If you are interested in our product, we have a number of different pricing options available, depending on the size of your database. Prices start from \$698 per server.

We hope this white paper has been helpful. *Thank you for your time!*