Table of Contents

------------------------------------------------------------------------------------------------------------------------------------------

[1) IRDB Implemented Securities 2](#_Toc476656711)

[2) Suggested Security Implementation 2](#_Toc476656712)

[3) Suggested Database Level Security 3](#_Toc476656713)

[Encrypted database backups 3](#_Toc476656714)

[Secure the database backup folder 3](#_Toc476656715)

[Audit logins 3](#_Toc476656716)

[Database Level Encryption 3](#_Toc476656717)

[Database Encryption Limitations 4](#_Toc476656718)

[Column Level Encryption 4](#_Toc476656719)

[Column Encryption Limitations 4](#_Toc476656720)

[4) IRDB Application Architecture 4](#_Toc476656721)

[5) UAT Server Details 4](#_Toc476656722)

[6) ZT SEVER Detail 5](#_Toc476656723)

[7) Microsoft Azure cloud-based 5](#_Toc476656724)

[Familiarity of Windows 5](#_Toc476656725)

[64-bit Windows VMs 5](#_Toc476656726)

[Scalability and flexibility 5](#_Toc476656727)

[Cost benefits and pricing model 6](#_Toc476656728)

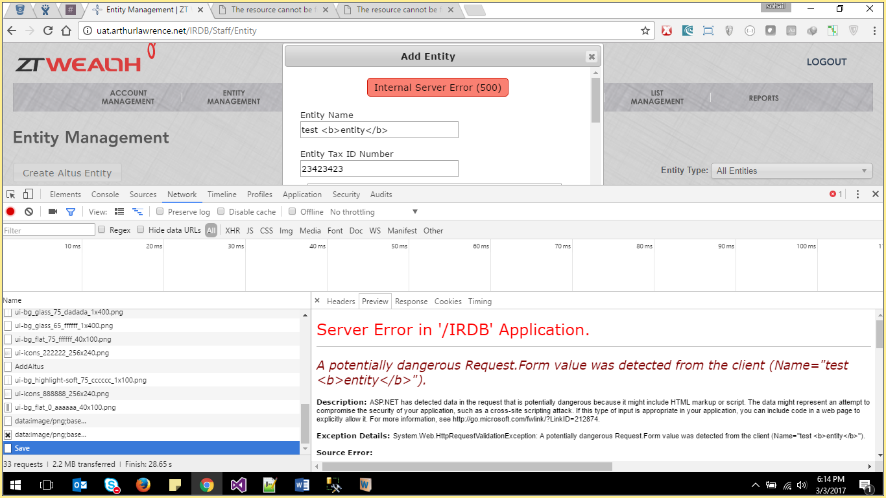
[Data center in the cloud 6](#_Toc476656729)

[Interoperability 6](#_Toc476656730)

[Security 6](#_Toc476656731)

# 1) IRDB Implemented Securities

1. **From Level Authentication**: We have implement form level authentication to validate login user credential. If user authenticated then we create authentication cookies that will help them to access their permission pages until he/she logged out.
2. **Encrypt User Password**: We have used MD5 Cryptographic algorithm with Hash techniques to encrypt user password.
3. **Role base Authorization**: We have implemented role based securities and assign to particular user. This will allow specific user to perform list of action or visit pages according to their role level. And also implemented role level Top Menu rendering.
4. **Https Implementation:**  We have used HTTPS to provide encrypted communication with and secure identification of a Web server.
5. **User Activity Logs:** We managing user activity via activity log functionality. This will help developers / Admin to trace user activities and also Errors generated during user actions.
6. **Client/Server side Validations:** We have applied validations rules on particular fields to make sure user inputs data and perform validation on client side as well as server side to avoid user tampering
7. **XSS (Cross site scripting) Validation:** we have restricted **JavaScript** and **HTML** tag in all input fields



1. **Daily Database Backups:** In case of failure or lose of data we regularly backup IRDB database

# 2) Suggested Security Implementation

1. Captcha implementation on Login Page
2. [Cross-Site Request Forgery (CSRF)](https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF)) Implementation
3. Allow only selected file extension in upload box
4. User account block on 5 wrong password attempt in login panel

# 3) Suggested Database Level Security

## **Encrypted database backups**

Database backups are always critical for an organization. If the backup files are not encrypted, they can easily be copied and restored to any other SQL Server, resulting in data theft and a security breach. A database administrator can avoid this unsavory scenario by creating database backups using the built-in MEDIAPASSWORD feature. The sample script below can be used to create encrypted database backups in SQL Server:

BACKUP DATABASE AdventureWorks  
TO DISK='C:\AdventureWorks.BAK'  
WITH MEDIAPASSWORD='C0mplexP@ssW0rd'  
GO

## **Secure the database backup folder**

A database administrator should make sure that access to the [database backup](http://searchdatabackup.techtarget.com/tip/Top-three-SQL-Server-database-backup-tips) folder is restricted and permission should only be granted to users who really need it. Unauthorized access can result in the database backup folder being vulnerable to backup files being copied to remote servers. Unauthorized access can also result in accidental deletion of critical backup files

## **Audit logins**

As a SQL Server security best practice, you should always audit failed logins to SQL Server. Once you have enabled login auditing in SQL Server, the failed and successful login information will be written to the SQL Server error logs, which can be monitored regularly to look into suspicious activities from time to time.

## **Database Level Encryption**

1. Simplest to implement – a few clicks and it is done.
2. No code changes necessary to applications
3. Negligible performance impact on typical transactions on a multi-processor server. More efficient and less impact than Column Encryption.
4. Prevents the database from being attached to an unauthorized instance of SQL Server .
5. Can protect databases on backup media
6. Can protect databases from the network, domain, or Windows administrators
7. Can protect databases from the SQL sysadmin in certain cases (requires a dedicated SQL instance for the database).

## **Database Encryption Limitations**

1. Single key for the entire database
2. Cannot protect against the SQL sysadmin in a SQL instance shared with other databases.
3. Does not encrypt data travelling over a network.

## **Column Level Encryption**

1. More flexible in choosing which pieces of data to encrypt. Applications can be written to ultimately control when, where, by whom, and how data is viewed.
2. Can protect data from the SQL sysadmin even where there is no dedicated SQL instance.
3. Contains both an API interface for maximum power and flexibility, and a seamless point-and-click interface for maximum ease of deployment.
4. Different columns (and even different rows) can be encrypted with different keys.
5. Can be combined with Encryptionizer DE to encrypt data over the network.

## **Column Encryption Limitations**

1. Small but present impact on performance in typical transactions – 5-6 percent on average- slower on accessing/updating an encrypted column versus plaintext column in typical transactions. The greater the number of columns encrypted, the greater potential for performance impact.
2. Limitations on types of database searches that can be performed. For example, comparison searches on an encrypted column (e.g., Last Name begins with “S”, Salary between $50,000 and $60,000) can be slow in a large database.
3. Cannot protect the intellectual property of the database (e.g., schema, views).

# 4) IRDB Application Architecture

1. ASP.NET MVC 5.0
2. Entity Framework 6.0 (ORM)
3. Design Patterns (Generic Repository , Inversion of Control (Dependency Injection) , Unit of Work
4. RDLC Reports
5. Client side frameworks (JQuery, Bootstrap, JQuery Datatable.NET, High Charts)
6. Microsoft SQL Server 2008 R2 Database

# 5) UAT Server Details

1. **Machine**: Microsoft Azure A3 – 4 Cores
2. **Operating System**: Windows Server 2008 R2 Datacenter 64-Bit
3. **Memory**: 7.0 GB
4. **Hard Disk**: First HDD 285 GB , Second HDD 127 GB
5. **Internet Information Services 7.5.7600**
6. **Microsoft SQL Server 2008 R2**

# 6) ZT SEVER Detail

# 7) Microsoft Azure cloud-based

Windows Azure is Microsoft's cloud-based application platform for developing, managing, and hosting applications off-site. Azure consists of several components: the cloud operating system itself; SQL Azure, which provides database services in the cloud; and .NET services. Azure runs on computers that are physically located in Microsoft data centers.

## **Familiarity of Windows**

Azure is based on Windows, so you can write applications in the same programming languages you've used for Windows apps: Visual Basic, C++, C#, etc. You can also use familiar tools such as Visual Studio, along with ASP.NET and other familiar Windows technologies. This makes it easy for organizations to find developers who already have the skills to create applications for the Azure platform

## **64-bit Windows VMs**

Applications running on Azure run in virtual machines, with each instance of the app running in its own VM on the 64-bit Windows Server 2008 operating system. The hypervisor on which they run is designed specifically for the cloud. You don't have to supply your own VMs or deal with managing and maintaining the OS because apps are developed using Web role instances or worker role instances that run in their own VMs. The apps interoperate with other Azure components through a Windows Azure agent that runs in each VM.

## **Scalability and flexibility**

Using Azure, you can easily create applications that run reliably and scale from 10 to 10 thousand or even 10 million users — without any additional coding. Azure Storage provides scalable, secure, performance-efficient storage services in the cloud.

After you create a Web app, you can specify the number of processors for the application to use. If the application needs to scale up to meet growing demand, it's easy to change the settings to use more processors. The "pay as you go/pay as you grow" approach lets you bring your new apps to market sooner and respond more quickly to changes in your customers' needs.

## **Cost benefits and pricing model**

Taking advantage of resources in the cloud allows you to decrease your costs for building and expanding your on-premises resources. You can also reduce the cost of IT administration because the hardware is being taken care of for you, off-premises. The cost of creating, testing, debugging, and distributing Web-based applications goes down because you have to pay only for the computer processing time and storage space you need at a given time.

Windows Azure pricing will be based on consumption, with a per-hour fee that's dependent on the size of the instance for Azure computing services and per-month or per-transaction fees for Azure storage services based on data size.

## **Data center in the cloud**

SQL Azure provides organizations with all the benefits of an enterprise-class data center without the hassle, headaches, and cost of maintaining such an entity. You get high availability and reliability with redundant copies of your data and automatic failover. No more worries about backing up data yourself.

## **Interoperability**

With Azure, you can develop hybrid applications that allow your on-premises applications to use cloud services, such as the cloud database and storage services. Communications services work between on-premises applications and the cloud, as well as mobile devices.

Azure supports open standards and Internet protocols, such as HTTP, XML, SOAP, and REST. There are SDKs for Java, PHP, and Ruby, for applications written in those languages, and Azure tools for Eclipse.

## **Security**

Knowing that security is one of the biggest concerns for companies considering a move to the cloud, Microsoft designed Azure with security in mind. The .NET Access Control Service provides a way to integrate identities, and Security Assertion Markup Language (SAML) tokens are used by applications to determine whether a user is allowed access.