WINDOWS AZURE: USING WINDOWS AZURE'S SERVICE BUS TO SOLVE DATA SECURITY ISSUES

Don Chambers

Columbus State University
D. Abbott Turner College of Business and Computer Science
The Graduate Program in Applied Computer Science

Windows Azure: Using Windows Azure's Service Bus to Solve Data Security Issues

A Thesis in

Applied Computer Science

By

Don Chambers

Submitted in Partial Fulfillment
of the Requirements
For the Degree of

Master of Science

August 2010

I have submitted this thesis in partial fulfillment of the requirements for the degree of Master of Science.

8/5/10

Date

Don Chambers

We approve the thesis of Don Chambers as presented here.

8/5/10

Date

Dr. Christopher C. Whitehead, Assistant Professor of Computer Science, Thesis Advisor

8/5/10

M S Khan

Date

Dr. Shamim Khan, Associate Professor of Computer Science

8/5/2010

Date

Dr. Thomas Loughman, Professor of Business Administration

Abstract

Many of the largest technology companies are heavily investing in Cloud Computing technology, making this a major force in the realm of software development. However, this software paradigm brings with it significant data related security issues. This paper discusses some of these security issues and presents a solution. The solution calls for storing data outside of the cloud, and under the control of the cloud consumer. In addition, the solution provides the ability to manipulate data using only the standard HTTP ports that are typically opened at most locations. This solution is implemented using Microsoft's Windows Azure Cloud Computing operating system. The Windows Azure Service Bus is used to facilitate data access for a cloud based application.

Table of Contents

List of Figures

List of Tables

Acknowledgements

I would like to express my genuine gratitude to my thesis advisor, Dr. Christopher C. Whitehead for helping me through this thesis. Dr. Whitehead provided the encouragement, guidance and support needed to help me successfully complete this thesis. I would also like to thank the other members of my thesis committee, Dr. Shamim Khan and Dr. Thomas Loughman for providing valuable feedback on the thesis as it progressed. In addition, I would like to thank the department chairperson Dr. Wayne Summers who provided input early in the thesis process.

I would also like to thank my family for the environment they provided while I worked on this thesis. Many long hours were spent researching this work and I appreciate their patience and support during this process. Now that it is finished, I should have many more free hours to repay them.

Finally, I would like to thank the faculty of the TSYS School of Computer Science at Columbus State University. They provided the breadth of knowledge that served as basis for this thesis. Without their wisdom and guidance, I could not have completed this work.

Chapter 1: Introduction

Cloud Computing, and its utility billing model, has the potential to transform the IT industry. It changes the way hardware is utilized while reducing the cost of unwanted idle time. In addition, Cloud Computing provides IT organizations with a robust and reliable infrastructure without the capital investment that was previously required. However, this new freedom brings with it potential data security issues. Moving data into the cloud environment presents several security, privacy, and legal issues. In many cases, these issues make the move to a cloud-based application prohibitive.

In chapter 2, this paper will define Cloud Computing. This chapter will give a brief history of Cloud Computing and explain its billing and usages models. An overview of several Cloud Computing products is also included.

Chapter 3 will present the data security problem, which this thesis will attempt to solve. This chapter will explain multiple security related problems with the data used in a Cloud Computing application. In addition, this chapter will touch on the current research in this field and present some possible solutions.

Microsoft's Cloud Computing operating system, Windows Azure, will be presented in Chapter 4. This platform will be used to solve the data security problem presented in Chapter 3. As this is a new product, this chapter will present an overview of the entire product, as opposed to focusing only on the portion needed to solve the problem identified in this thesis.

Chapter 5 will present a solution to the problem identified in Chapter 3. This chapter will start with a generic explanation of how the solution works. This will be followed by a description of the actual implementation.

In Chapter 6, the solution will be evaluated. This evaluation will compare this solution with other possible solutions identified in Chapter 3. In this chapter, the advantages and disadvantages of each approach will be discussed.

Chapter 2: Cloud Computing

What is Cloud Computing? In recent years, the Cloud Computing buzzword has grown with intensity but its meaning is not clear. It seems that many people, and companies, tout the power of Cloud Computing but have trouble explaining exactly what it is. In fact, there are many different definitions given by different technical authorities and Cloud Computing vendors. Some analysts claim that Cloud Computing has been with us for years and it is simply the modern version of Utility Computing, with a catchy new buzzword (1; 2). Others claim it is much broader and includes anything used outside of your local firewall (2). The goal of this chapter is to explain what Cloud Computing really means. This paper takes the point of view of a software developer, as a user of the Cloud Computing environment, and it does not deal with setting up a Cloud Computing data center.

Utility Computing refers to a business model that focuses on providing computing services for a fee based on usage (3). This is similar to the current business models used by public utilities. Public utilities, such as the water service, provide a continuous service that can be used as needed, and the cost is based on the amount used. This public utility business model is analogous to the Utility Computing business model, in which computing services are always available and the customer is charged based on usage. Utility Computing is not a new concept and as early as 1955, MIT professor John McCarthy compared it with a public utility company in the following quote:

> "If computers of the kind I have advocated become the computers of the future, then computing may someday be organized as a public utility just as the telephone system is a public utility... The computer utility could become the basis of a new and important industry." (4)

Utility Computing offers an economic model that, in some cases, can be superior to that of an on-site corporate data center. Studies have shown that servers residing in corporate data centers are underutilized with an average idle time of approximately 85%. The reason for this idle time is that servers in a corporate data center must be able to handle periodic spikes in demand and possible future load increases (5). However, with Utility Computing, a company does not pay for idle time. Instead, the company will only pay for the computing resources it needs, when it needs them.

Utility Computing is a type of Cloud Computing. However, Cloud Computing is a broader concept that describes the architecture on which Utility Computing is based. Cloud Computing can also describe categories of computing that do not meet the definition of Utility Computing. In fact, many forms of Cloud Computing offer services and benefits far in excess of those offered by standard Utility Computing (6).

The general term Cloud Computing refers to a computational model in which software is hosted, run, and administered in offsite data centers (7). This software is then provided to the end users, as a service, over the internet. A Cloud Computing environment consists of at least one to three service layers, and each layer provides different types of services. These layers are Infrastructure as a Service (Iaas), Platform as a Service (PaaS), and Software as a Service (SaaS) (2).

Layers

Infrastructure as a Service

Infrastructure as a Service (IaaS) is simply minimal computing services such as disk storage space, CPU use, and network access (2). This functionality is used over the Internet and paid for with a per-usage model. In this scenario, the hardware is fully outsourced and the cost to the consumer is based on usage. This service layer is essentially the basic service provided by Utility Computing, described above. As such, this layer does not provide software and the consumer of an IaaS cloud is responsible for installing, maintaining, and licensing any required software components. A Platform as a Service layer can be added on top of this layer in order to provide additional functionality.

Platform as a Service

Platform as a Service (PaaS) is a model that offers a higher level of service than IaaS. This layer builds on the IaaS layer and moves additional server software into the cloud (2). This type of service provides access to functionality that normally resides on corporate servers. Examples of the services provided as part of PaaS architecture include file servers, Web servers, and email servers. In addition, PaaS may include additional tools that can be used for application development, such as databases and design tools. A Software as a Service layer can be added on top of this layer in order to provide even more services from the cloud.

Software as a Service

Software as a Service (SaaS) is a method of software delivery that builds upon the services provided in the PaaS layer (2). With this type of service, software is hosted offsite and delivered via the internet. This type of computing has existed since at least 1999, which was before the term Cloud Computing was used. At that time, the technology was referred to as ASP, meaning Application Service Provider (8). In a SaaS model, the providers of the software typically license the applications to users on a subscription basis. The economies of scale that exist with such a large user base generally allow the subscription price to be much lower than the cost of purchasing, installing, and maintaining the software on desktop computers (9). This has the effect of changing the business model from one in which customers own the software, to a model in which some external provider owns the software. As a result, the responsibility for the technology infrastructure, software licenses, and patching shift to the provider and allow the end user to save both time and money.

Virtualization

Cloud Computing is based on the concept of shared server resources (5). As such, servers running in remote data centers are not dedicated to a single application or customer. Instead, several applications belonging to multiple customers may exist on the same physical server. Conversely, a single application belonging to a single customer may utilize resources from several physical servers. In addition, the cloud must manage spikes in usage that may temporarily require additional processing power. In order to provide the resource sharing and dynamic scalability required by a robust Cloud Computing environment, the data centers often employ virtual servers (5).

The server virtualization process is the act of 'masking' server resources from the user of the server (10). This masking process abstracts the physical server and operating system from the user of the virtual server. The virtual server may be comprised of one or more physical servers. In addition, the operating system of the virtual server may not be the same operating system as the underlying physical servers.

The virtual server can then be allocated to the customer, while any number of physical servers may actually be used. In addition, multiple virtual servers running on a single physical server allows multiple customers to share physical hardware while it appears that each has a dedicated environment. The elastic nature of these virtual servers allows dynamic load balancing as they pull in and release physical hardware resources as needed. With this design, processing power not needed by one customer is made seamlessly available to others, thereby eliminating or reducing system idle time. This reduction in idle time is the perfect complement to a billing model that charges based on actual usage (11).

Economics

Cloud Computing brings a different economic paradigm to the world of software development. In the past, when a company needed to grow its IT infrastructure it had to make a capital investment in infrastructure. A capital investment refers to the money a business uses to purchase fixed assets (12). This capital investment would require large upfront cash expenditures, which is not typically a deductable transaction. Instead, the capital investment results in a capitalized asset that must be depreciated over an extended period (13).

Cloud Computing brings a different economic paradigm to the information technology industry. Typically, there is no upfront capital investment in a Cloud Computing IT infrastructure. Instead, the required usage is purchased which results in an expense that frequently provides a tax deduction (9) (13).

In addition to the benefits to large corporate data centers, the economic model of a Cloud Computing infrastructure will bring benefits to small companies (14). No longer will a small startup incur large up-front costs in order to compete in the hi-tech game. This fact lowers the barriers-of-entry, which allows more competitors to enter the marketplace (15). With more competitors comes a decrease in cost and improvements in technology. With more players in the game, there is an increased potential to achieve important innovations.

Cloud Computing Products

Many vendors offer a variety of products that fall into the Cloud Computing category. Some of the products qualify as full SaaS solutions, while others may only expose lower layers of the Cloud Computing paradigm. Below are examples, and brief descriptions, of three of the more popular Cloud Computing offerings.

Amazon Elastic Compute Cloud

The Amazon Elastic Compute Cloud (EC2) is an example of an Infrastructure as a Service (IaaS) environment, providing the basic resources needed for customers to deploy their applications and services (16). A Web Service Interface is provided that can obtain and boot new server instances allowing the customer to scale capacity as the requirements change and the customer only pays for the resources actually consumed. A large standard on-demand instance, running a windows OS, is priced at $0.48 per instance-hour for each instance launched (16). Currently, any incoming data transfer is free and the outgoing data transfer is $0.15 per GB for the first 10 terabytes each month. After 10 TB, the cost per GB decreases. These costs ultimately stabilize at $0.08 per GB after 150 TB have been transferred in a given month (16).

Windows Azure

Microsoft's Windows Azure is a Platform as a Service (PaaS) Cloud Computing solution (17). Windows Azure provides an application server, data storage, and networking in a combined infrastructure that can be used for building and deploying Windows applications. Azure relies on virtualized data centers with pools on hardware that create a Utility Computing environment (17). Applications can be created using the existing Microsoft Visual Studio software development product and deployed into the Azure cloud.

Windows Azure offers a payment plan that bills based on computing time, data transfers, storage space, and storage transactions. Currently, a medium Azure instance cost $0.24 per CPU compute hour and data transfers cost $0.10 per GB income and $0.15/GB outgoing. In addition, storage cost $0.15/GB per month with an additional $0.01 per 10KB for storage transaction (17) (18).

Google Apps

The software giant Google is one of the leaders in the field of SaaS (19). Their Google Apps suite offers a variety of applications using a subscription model. These applications include a word processor, spreadsheet, presentation tool, email server, and more. The Google Docs portion of the Google Apps package includes applications similar to those found in Microsoft Office 2007, standard edition. The cost of Microsoft Office is $399.95 per user. The current cost for Google Apps is $50 per year for each user. The Google Apps cost is a recurring yearly cost, but upgrades and patches are implemented with no additional cost (20) (21).

With the MS Office desktop program, a license is required for each physical computer that will have the software. This means an employee would need a second license to use the product from home. In addition, the office software must be installed on a desktop and there is some maintenance cost with application patches. Periodically, the application is upgraded and the cost of the new version is usually a few hundred dollars.

Summary

The term Cloud Computing refers to any off-site computing environment with the most basic version simply being access to a server on which customers can install their applications and services. More advanced Cloud Computing environments provide services, such as email servers and applications servers that allow the customer to build and deploy custom applications to the cloud. The pinnacle of Cloud Computing is software application running in the cloud and made available over the Internet on a subscription basis.

Most Cloud Computing implementations employ virtualization technologies that allow multiple resources to appear as a single resource. This technology allows the cloud to expand and contract based on the system load and required performance levels. This elastic nature of the cloud allows for a pay-as-you-go billing model, based on usage, and the elimination of costly idle computing time.

In some cases, the type of cloud is based on the perception of the user. For example, a developer may build applications using a Platform as a Service (PaaS) cloud and make those applications available to an end user. From the end users perspective, the cloud is a Software as a Service (SAAS) cloud.

Chapter 3: Data Security Problem

Moving an application into the cloud offers numerous benefits but it is fraught with security issues (22). While many of these security issues are related to moving an entire application into to the cloud, this paper will focus on issues related to moving an application's data into the cloud. This outsourcing of data into a Cloud Computing environment is one of the biggest concerns for technology decision makers. A recent study from CIO magazine found that 50% of CEOs said data safety was one of their biggest worries when considering a move to the cloud (23). Additionally, a Cloud Computing cyber-security survey found that 70% of technology decision makers for the government were very concerned about data security (24). This section will detail three categories of data related security concerns: privacy, compliance, and legal (22).

Security Issues

Privacy

When a company stores data on another company's hardware, they lose some degree of control over that data (25). In addition, data stored in the cloud is much easier for the government to obtain than the data stored by a company, because a lower legal standard applies (23). For these reasons, access to the data must also be controlled due to privacy issues. While there are legal and privacy requirements, there is also the chance of cloud provider espionage when the data includes trade secrets or proprietary marketing data (26).

Compliance

A multitude of laws and regulations have forced specific compliance requirements onto many companies that collect, generate or store data. These policies may dictate a wide array of data storage policies, such as how long information must be retained, the process used for deleting data, and even certain recovery plans. Below are some examples of compliance laws or regulations.

- In the United States, the Health Insurance Portability and Accountability Act (HIPAA) requires a contingency plan that includes, data backups, data recovery, and data access during emergencies (27).
- The privacy laws of Switzerland demand that private data, including emails, be physically stored in the Switzerland (22; 28).
- In the United Kingdom, the Civil Contingencies Act of 2004 sets forth guidance for a Business contingency plan that includes policies for data storage (22).

Legal

In addition to regulatory compliance obligations, there may be additional legal responsibilities related to the storage of data. In a virtualized Cloud Computing environment, customers may never know exactly where their data is stored. In fact, data may be stored across multiple data centers in an effort to improve reliability, increase performance, and provide redundancies. This geographic dispersion may make it more difficult to ascertain legal jurisdiction if disputes arise (29). Some additional legal issues may result from contractual obligations that cloud customers may have with their own end customers. In addition, since the data is residing on the server of another company, it is possible that intellectual property rights will be affected (22).

The responsibility to protect against data breaches is now shared by the hosting company and the consumer. The integrity of the data must be protected, a viable disaster recovery program must be in place, and manipulation of the data must be controlled. By spreading this responsibly to multiple organizations (cloud vendor, and cloud consumer), the legal issues may become unclear.

The legal issues are not confined to the time period in which the cloud based application is actively being used. There must also be consideration for what happens when the provider-customer relationship ends. In most cases, this event will be addressed before an application is deployed to the cloud. However, in the case of provider insolvencies or bankruptcy the state of the data may become blurred (28).

Provider Lock In

Storing data in the cloud opens the cloud consumer to the possibility of lock-in with the Cloud Computing provider (26). The cloud may store data in a proprietary format that cannot be assessed without the cloud. Thus, if a cloud goes out of business, some user data may no longer be accessible. While this is not directly a security issue, it may have implications with legal and compliance issues, which fall into the security category.

Current Research/Possible Solutions

Data in the Cloud

The most straightforward method for storing application data may be to store the data in the cloud. The programming model for this method is very close to existing non-cloud development models making it easier to implement. In addition, application performance is usually at a peak with fewer protocol layers and reduced data transfer times. However, the data in the cloud must be protected from unauthorized use.

1. *Encrypting Data*

In order to protect data in the cloud, the data can be encrypted before being stored in the cloud. Conversely, data being returned from the cloud will be decrypted. Glenn Brunette, an engineer at Sun Microsystems is working on a project called the cloud safety box (30). The goal of this project is to create an Interface to a cloud storage provider that enables encryption/decryption of content stored in the cloud.

2. *Information-Centric*

An information-centric approach to storing data in the cloud is a self-protection scheme. Data is encrypted and packaged with a usage policy. When the data is accessed, the data item should only reveal itself to a trustworthy caller based upon the policy (26).

3. *High-Assurance Remote Server Attestation*

The High Assurance Remote Server Attestation method provides a mechanism for the data owner to audit how the data is being used. This can be done to ensure that data is not being abused or leaked. This method does not actually protect the data, but it provides a mechanism for ensuring that security has not been breached (26).

4. *Privacy-Enhanced Business Intelligence*

The Privacy-Enhanced Business Intelligence method encrypts all data stored in the cloud. This is similar to the Encrypting Data method described above; however, special features have been added that allow the data to be searched. This searchability allows a search query to be encoded, in which case the cloud can then decide if the stored data matches the encoded search query (26).

Data External from the Cloud

One option for retaining control of the data is to store it outside of the cloud, in an on-site data center. The primary drawback of this approach lies in accessing data from the cloud-based Web application

1. *Firewall Exceptions*

In order for the application to access the data, certain firewall excepts will need to be made. At the very least, the required ports will need to be opened in the cloud and at the on-site data center storing the data.

2. *Web Service Lookup*

Web Services, running in the on-site data center, can be used to make the data available to the application. This approach overcomes problems with the Firewall Exceptions method.

Summary

Moving proprietary data from the enterprise data center into the cloud brings with it a set of unique data security related challenges. These challenges are not simply theoretical exercises; instead, they can have real world consequences. In March 2010, Yale University decided to postpone a decision to adopt Google Apps for Education, citing privacy concerns (31). Among the reasons given was the fact that data may be stored in other countries, subject to foreign law. In addition, there were concerns related to intellectual property rights (31). In order to increase the viability of Cloud Computing, a method for overcoming these data security related issues must be devised. This thesis will devise and implement such a method.

Chapter 4: Windows Azure

The Microsoft Windows Azure platform is a Cloud Computing environment that allows customers to run applications and store data in Microsoft data centers around the world (32). The core of the Azure platform is the Windows Azure cloud-based operating system. This operating system provides the compute, storage, and networking capabilities required by Cloud Computing applications, as well as essential connectivity and security related services, such as the Service Bus and the Access Control module (32) (33). In addition, there is a cloud-based relational database, known as SQL Azure, which can be used to hold proprietary data that is normally held in a customer's relational database (33).

Microsoft Windows Azure is the Cloud Computing platform that will be used to solve the problem described in Chapter 3. Therefore, this chapter will provide a high-level overview of the platform in order to build a foundation for the solution that will be presented in Chapter 5. This chapter will cover the common components used to build a Windows Azure application.

REST

REST (Representation State Transfer) is an architectural style that abstracts the underlying physical architecture used by a distributed system. The term REST was first used by Roy Fielding, one of the primary authors of the HTTP specification, in his doctoral dissertation (34). A brief description is included here because Windows Azure makes heavy use of this architectural style.

With the REST architecture, all resources are accessed through a standard Interface. In the Windows Azure environment, this standard Interface is HTTP. A component that implements the REST architecture is known as a RESTful component (35). A user that calls a REST component is known as the consumer, and the consumer accesses the component using a URL (34). The consumer only needs the URL, and a method for interpreting the data returned by the response. The underlying application and the system architecture behind a RESTful component are irrelevant. Changes in the underlying architecture and implementation can change, as long as the response data and reference URL remain the same.

Fabric

Windows Azure uses a computing fabric to provide the common building blocks needed to create robust enterprise-level applications. A computing fabric is a highly virtualized software infrastructure that spans multiple computing resources. These resources include processors, memory, physical storage and peripherals, which are loosely coupled with each other (36). The goal of the fabric is to abstract the physical computing architecture from the developer. Thus, a fabric appears as a single system regardless of how many actual systems are used make up the fabric. This fabric infrastructure allows for the dynamic scalability that is essential for a pay-for-usage Cloud Computing environment. The Windows Azure Fabric, known as the AppFabric, is comprised of two primary areas of functionality, compute and storage.

Compute

The Azure AppFabric Compute service is what makes it possible for an application to execute in the cloud. This service is a Windows Sever virtual machine hosted in a Microsoft Data Center. As a result, the scalability of this operating environment is simply a configuration matter and the deployed application can scale without theoretical limits (32).

The Azure Compute Service provides support for the .Net framework, unmanaged code, and other development approaches. Applications written with Visual Studio, using languages such as C#, Visual Basic, C++, and Java are supported. In addition, other technologies such as PHP and Windows Communication Foundation (WCF) are supported (33).

Storage

The Azure AppFabric Storage service provides data storage for simple tables, blobs (binary objects), and queues (32). This is not a relational database and the storage service is not queried using SQL. The data in the storage server can be manipulated with the REST API (application programming Interface), which is based on HTTP requests. This approach to data storage allows anyone, regardless of platform, to integrate with the storage services (32). The Azure Storage service can be accessed by applications running in the cloud, and by on-site applications outside of the cloud. In either case, the REST API is used (33).

SQL Azure

The Windows Azure platform offers the optional SQL Azure database as a supplement to the rudimentary data storage provided by the Storage AppFabric. SQL Azure is a cloud-based relational database that is an extension of Microsoft SQL Server. This multi-tenant, highly scalable database runs in the Azure cloud. This cloud-based database supports the same T-SQL version of SQL as existing SQL Server databases. In addition, SQL Azure offers standard relational database features such as triggers, views, stored procedures, and indexes (32).

Access Control

The Access Control module of the Windows Azure AppFabric is a claims-based access control in the cloud that provides single-sign on capability to all applications running in the cloud. In addition, this access control module can be used for applications running outside of the cloud. This Access Control module of Windows Azure is accessed using a collection of REST Web Services. The current architecture behind the Access Control service is an Active Directory implementation. However, the use of RESTful components abstracts this implementation from the developer permitting future implementation changes (32).

Service Bus

The primary purpose of the Windows Azure Service Bus is to relay messages through the cloud in order to support application connectivity. The Service Bus gets its name from the Enterprise Service Bus (ESB) Pattern. This ESB pattern defines a standards based integration model that combines Web Services, data transformation, messaging and intelligent routing. The ESB platform is used to coordinate the interaction of diverse applications (37). The Windows Azure Service Bus makes it possible to integrate third party services from Microsoft or other vendors. In addition, custom applications running inside of a firewall can utilize the Service Bus to connect to applications outside of the firewall.

The Azure Service Bus exposes an application's services through an endpoint. Each endpoint is assigned a URI (Universal Resource Identifier), which is published using the service register. Endpoints can then be 'discovered' by clients that use the service register. Each endpoint provides a rendezvous address that can be used for communication. Some of the available communication types are listed below (32).

One-Way Messaging

With one-way messaging, a single on-site service registers with the Service Bus relay to "listen" for messages at a particular rendezvous address. Clients can then send messages to the Service Bus and they will be relayed to the registered service (32).

Publish/Subscribe Messaging

Publish/Subscribe messaging, also known as multicast messaging, is very similar to the one-way messaging described above. The difference is that Publish/Subscribe messaging allows multiple services to listen to the same Service Bus rendezvous address. When a client submits a message to this address, the relay will distribute the messages to all services that have registered (32).

Direct Connectivity

In addition to the relayed communication methods described above, the Service Bus also allows direct connectivity between clients and services. In this scenario, the relay is still used but the relay will attempt to connect the clients and services directly to one another. This is done through a port prediction algorithm and the purpose is to increase throughput (32).

Summary

Microsoft Windows Azure is a Cloud Computing platform allowing applications to be deployed to the cloud and accessed using the Web. By default, the cloud provides computation and storage resources. An optional cloud-based relational database is also available for more complex data storage needs. Interaction with the cloud is controlled by the Service Bus, which facilitates interaction between multiple applications, both on-site and cloud-based. The Access Control service is used to authorize users to access services in the cloud. The next chapter will describe a solution, using Windows Azure, to solve the problem described in Chapter 2.

Chapter 5: The Solution

Chapter 3 presented several security issues that result from storing data in a cloud. The primary cause of these problems is that data is stored with an outside source. When discussing the privacy implications of Cloud Computing, the nonprofit consumer advocacy organization Privacy Rights Clearinghouse said, "Obviously, the safest approach is to maintain your data under your own control." (25)

The solution to this data security problem lies with storing data outside of the cloud. However, in order to realize benefits from Cloud Computing, the application must still execute in the cloud. The challenge is to solve the problem of using a cloud-based application to access and alter data that is hosted outside of the Cloud Computing environment. This chapter will detail this solution, and describe a simple research project that validates the solution.

The Solution

The solution begins with a .Net Web application running in the Windows Azure Cloud. The data used by this application will be stored in an Oracle database outside of the cloud. All data manipulation and retrieval will be performed using the Windows Azure Service Bus.

Service Bus

The Windows Communication Foundation (WCF) framework is a platform for building distributed service-oriented applications (38). A WCF application can be accessed by any application that can read and send messages in a SOAP (Simple Object Access Protocol) format. This framework will be utilized to create a service and client that both communicate with the Windows Azure AppFabric Service Bus. The Service Bus will expose an Endpoint, which is the object exposed by Service Bus (or any WCF application). This endpoint is composed of an address, binding, and service contract.

End Point Address

The Endpoint is the object exposed by Service Bus (or any WCF application). The address is the URL that is used to access the endpoint (32).

Binding

The binding will specify the transport, encoding, and protocol details that are needed for clients to communicate with the Service Bus (39). WCF provides several bindings with common default settings. This project will be using the BasicHttpRelayBinding because it uses HTTP as the transport and does not require the opening of ports other than the standard HTTP port, port 80 (40). There are several other bindings available. Some bindings are more secure and encrypt the data; other bindings offer better performance but require opening additional ports. The BasicHttpRelayBinding binding will suffice for this project. However, in a real-world implementation, other bindings may be more appropriate and the decision should be based on the actual needs of the application.

Service Contract

The service contract describes the operations that are exposed by the Endpoint. The service contract is created using an Interface, which will be implemented by both the service and the client. The Interface must have the ServiceContractAttribute applied to it, which will indicate that the Interface defines a service contract for a WCF application. Methods use the OperationContractAttribute to indicate that a given method defines an operation within the WCF service contract (41).

Channel Interface

A channel is the WCF object that is responsible for sending and receiving message between the service and the client. The channel Interface must implement service contract Interface as well as the IClientChannel Interface. The IClientChannel Interface defines the operations that are supported by the channel (42).

Implement Service and Client

The service is implemented as a class that implements the Interface for the Service Contract that is explained above. This class must have the ServiceContractAttribute attribute applied to it, with the same properties as the service contract Interface it implements. In addition, each method from the service contract Interface must be implemented in this class.

The project used in this thesis has a client side class that also implements the Service Contract. The purpose is to provide a client side class, identical to the server side class, that abstracts the Service Bus calls. The Service Bus calls could be included in the Web application, but this additional layer abstracts the data source from the presentation layer.

A service is an application that exposes endpoints, which can perform operations. This is similar to existing WCF endpoints, except in this case the endpoint is in the cloud – not on a local server. The client is an application that invokes the service operation. Below is a description of how the service and clients are built, starting with a section that covers materials common to both types of connections.

Service Bus Connection Common Steps

1. Credentials are required to connect to the Azure Service Bus. An instance of the TransportClientEndpointBehavior class is used to store the credential information. This includes the type of credential used, the user, and the password.
2. An instance of a binding class is created. In this case, the BasicHttpRelayBinding will be used.

3. The Service Bus Connectivity mode is set. In this case, the mode is set to AutoDetect, which automatically selects between TCP (Transmission Control Protocol) and HTTP modes.
4. The CreateServiceURI method, of the ServiceBusEnvironment class, is used to create a Service Bus URI for the application.
5. An instance of the ContractDescription class is instantiated. This object defines the operations that the endpoint will provide. It holds the type of the Service Contract Interface and the type of the actual service implementation.

Service Specific Steps

This section explains the service specific steps for setting up a Service Bus connection.

1. A ServiceEndpoint object is instantiated, which contains the URI and binding created in the "Service Bus Connection Common Steps" section above.
2. Once the steps in the common section above are complete, an instance of the ServiceHost class is created. This object is provided the type of the actual service implementation as well as an instance of the URI that was created in step 4 of the "Service Bus Connection Common Steps" in the section above.
3. A reference to the TransportClientEndpointBehavior object, created in step 1 of the "Service Bus Connection Common Steps" section above, is added to the ServiceEndpoint object.

4. A reference to the ServiceEndpoint created in step 1 is added to the ContractDescription defined in step 5 of the "Service Bus Connection Common Steps" in the section above.
5. A reference to the ServiceEndpoint is added to the ServiceHost object created in step 1.
6. The open method of the service host is called, which makes the endpoint become useable on the Service Bus.
7. The close method of the service host is called when the endpoint should no longer be available.

Client Specific Steps

The section explains the client specific steps used to communicate with the Service Bus.

1. An EndpointAddress object is instantiated, with a reference to the URI created in step 4 of the "Service Bus Connection Common Steps" in the section above.
2. A reference to the TransportClientEndpointBehavior object, created in step 1 of the "Service Bus Connection Common Steps" section above, is added to the EndpointAddress object.
3. A ChannelFactory object is created, using the EndpointAddress object. This factory creates channel objects of different types that are used by clients to send messages to Service Bus endpoints.

4. A channel object, implementing the Channel Interface is created by the ChannelFactory object. This channel will be used to communicate with the Service Bus.
5. The open method of the channel object is invoked, which opens a connection to the Service Bus.
6. The channel object also implements the Service Contract; therefore, each service exposed by the service, on the Service Bus, has a corresponding method in the channel object. These methods can be invoked as any other method, and the call will invoke the service-side code via the Service Bus.
7. When all channel operations are complete, the close method of the channel object should be called to terminate the connection to the Service Bus.

The Implementation

The first step for creating a project that validates the proposed solution was to create a Windows Azure account. This project uses the "Development Accelerator Core" package, which cost $59.95 per month, and includes 750 hours of a small compute instance. When the application is deployed, the compute hours are being used, even if the application is not actually being accessed by a user. This package was needed to leave the application deployed full-time with no overage charges.

This package includes 10 GB of storage, which is more than enough because the data will be stored outside of the cloud. Five Service Bus connections, with 1 million transactions, are provided with this package. In addition, 7GB of incoming data, and 14GB of outgoing data are provided. A description of this package can be found at the following URL: http://www.microsoft.com/windowsazure/offers/popup.aspx?lang=en&locale=en-US&offer=MS-AZR-0006P#.

For the project, an existing Web Application was ported to the Cloud environment. This application was developed as a custom product for a company that manufactures power meters that can wirelessly transmit power readings to the power company. This application was modified so that all database interaction uses the Windows Azure Service Bus. The application is large and only two screens were ported to the Cloud. These screens are sufficient for validating the solution. They will illustrate retrieving data from the database, as well as updating data in the database.

Database

The database used for this project is Oracle 11G. The structure and data used in this implementation was ported from the SQL Server database that the application originally used. Only the database objects required for the scaled down application were moved to Oracle.

The database was converted to Oracle, to illustrate the decoupled nature of the Azure Service Bus. Oracle is used to illustrate that the solution does not require all technologies come from the same vendor. By default, Oracle exchanges data on port 1521 (43) and this default is used for this project. However, the Azure Service Bus operates on the standard HTTP port, 80. For this project, only port 80 is used to exchange data between the cloud-based application and the database.

For legal and privacy reasons, the data was scrubbed prior to entry. The scrubbing consists of changing some indentifying characteristics of the data, resulting in fictional data that resembles real-world data. The data used for this project is not actual data, but it is an accurate representation of the real-world data used by this application. In addition, only a minimum amount of data was moved to the database. This small amount of data is sufficient to illustrate the proposed solution. Furthermore, some table objects were renamed and view definitions were slightly altered. The database Entity Relationship diagram is shown in figure 5-1 below.

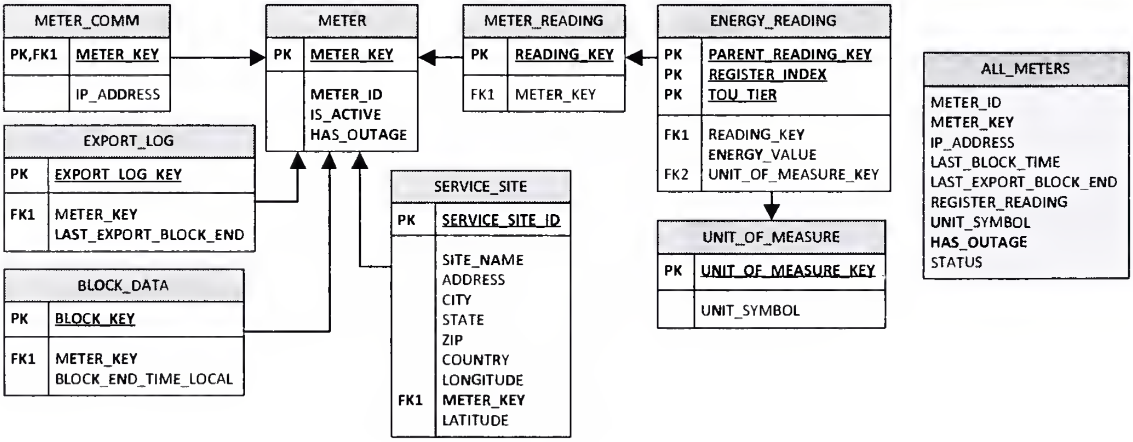


Figure 5-1 Entity Relationship Diagram

Application

The cloud based solution uses .Net 3.5, and was developed using Visual Studio 2008. This application is comprised of five subsystems. Each subsystem is represented in Visual Studio as a separate project as shown in figure 5-2.

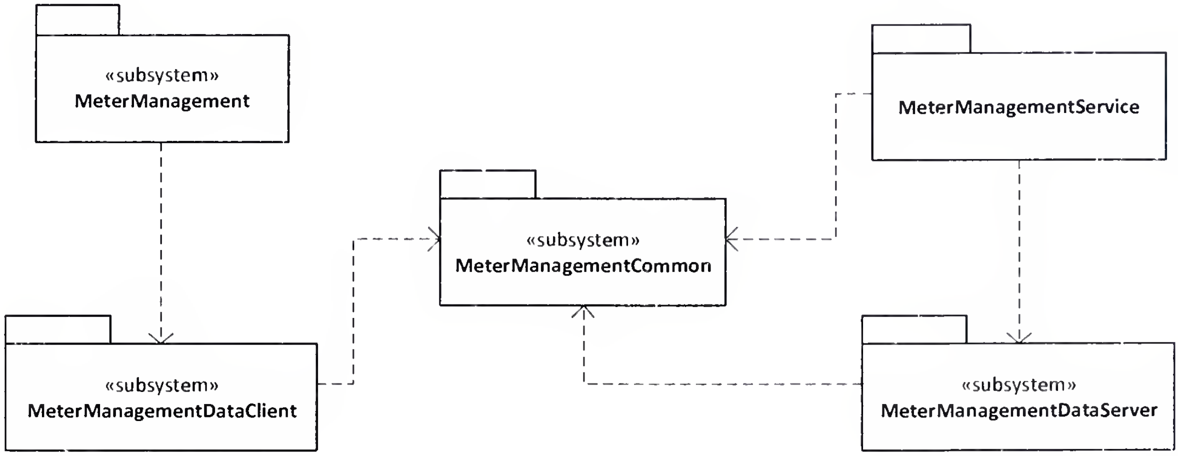


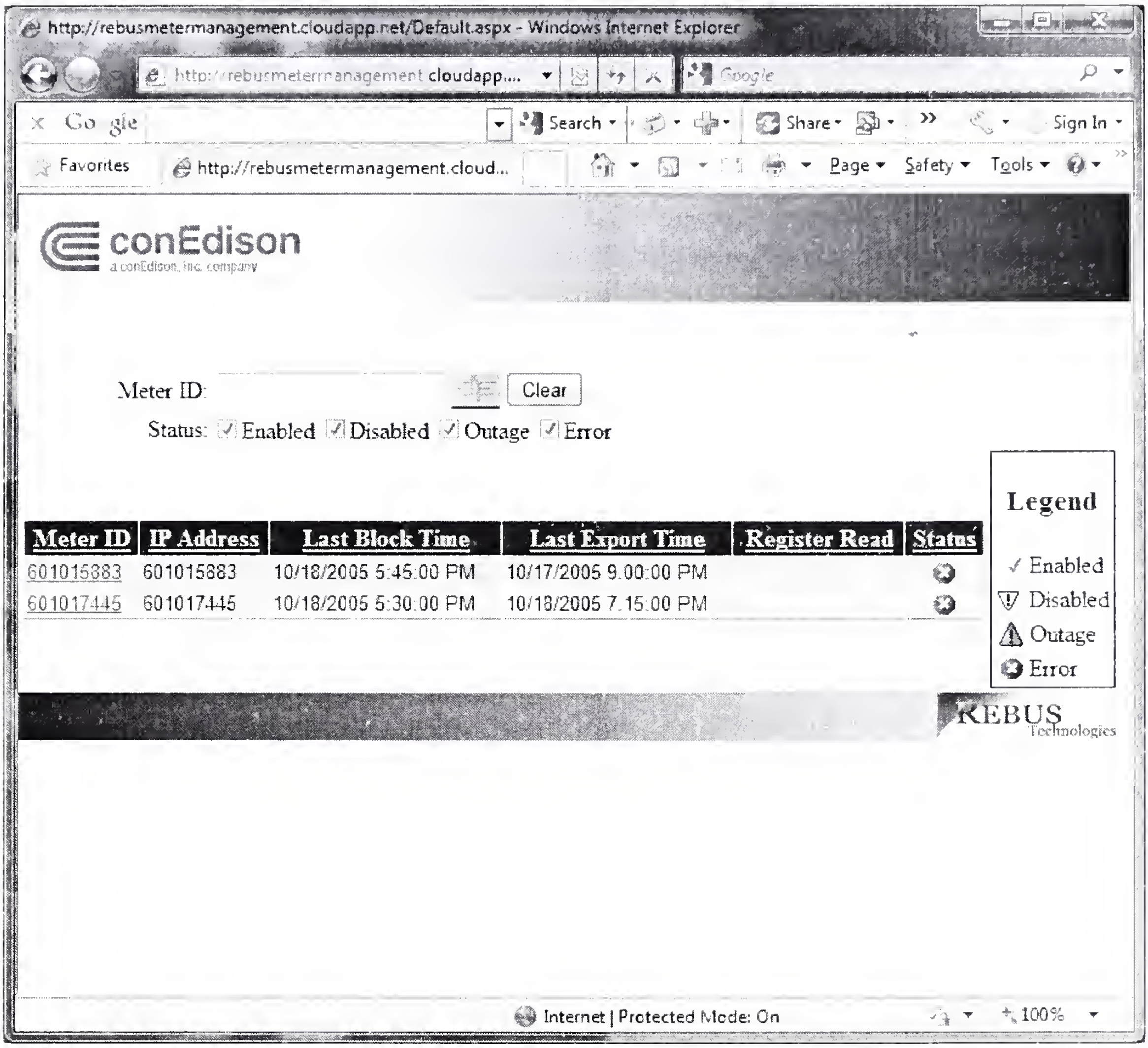
Figure 5-2 Subsystem Diagram

1. MeterManagement is the cloud based Web Application. It manipulates data by making calls to classes in the MeterManagementDataClient subsystem.
2. MeterManagementDataClient is a class library, which includes all the Service Bus calls. This layer abstracts the Service Bus from the Web application. If the data source is changed, this class could be updated to reflect the new location.

3. The MeterManagementService is a Windows service that interacts with the Azure Service Bus. It ensures the server-side Service Bus connection is always operational.
4. MeterManagementDataServer is a class library containing the objects that directly connect to Oracle to manipulate data. The channel connection to the Service Bus calls this class for database interaction.
5. MeterManagementCommon is a class library containing the Interfaces used by MeterManagementDataClient, MeterManagementDataServer, and MeterManagementService. This ensures that the database actions needed by the client are actually implemented on the server side. In addition, it defines the Interface for the Service Bus Channel.

MeterManagement - Web Application

MeterManagement is a Web application, implemented using ASP.Net 3.5, and consisting of two screens. The first screen, the Meter List, will show a list of all power meters in the database. The screen allows the user to filter the results based on the meterID, or the meter status. This screen contains an ObjectDataSource, which is configured to call the GetMeterData method, of the Meter object, in the client-side class library. This screen is shown in the figure 5-3.



http://rebusmetermanagement.cloudapp.net/Default.aspx - Windows Internet Explorer
http://rebusmetermanagement.cloudapp....
Google
Google
Search
Share
Sign In
Favorites
http://rebusmetermanagement.cloud...
Page
Safety
Tools
conEdison
a conEdison, inc. company
Clear
Legend
REBUS
Technologies
Internet | Protected Mode: On
100%

Meter ID:

Status: ☑ Enabled ☑ Disabled ☑ Outage ☑ Error

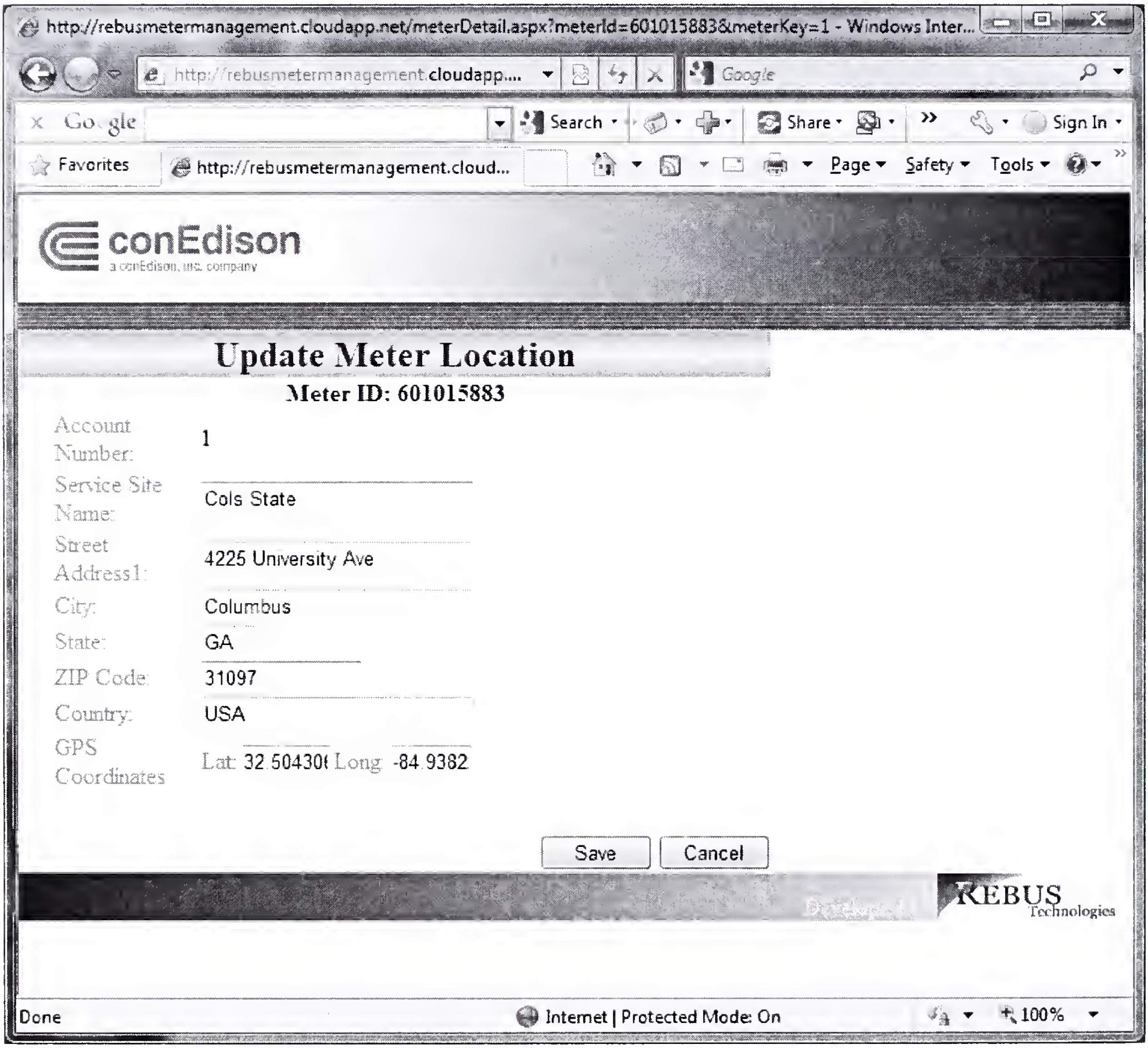
Meter ID	IP Address	Last Block Time	Last Export Time	Register Read	Status
601015383	601015883	10/18/2005 5:45:00 PM	10/17/2005 9:00:00 PM		✖
601017445	601017445	10/18/2005 5:30:00 PM	10/18/2005 7:15:00 PM		✖

Legend
✓ Enabled
▽ Disabled
⚠ Outage
✖ Error

Figure 5-3 Meter List

The second screen in the application is the Service Site screen. Clicking on a meter on the first screen takes the user to this screen, which will display information about the physical location of the meter selected. On this screen, the user may update the data associated with the service site location.

This screen contains an ObjectDataSource, which is configured to call the GetSiteLocationData method, of the Meter object, in the client-side class library. In addition, the same ObjectDataSource is configured to call the UpdateSiteLocation method of the Meter object, when an update occurs. This screen is shown in figure 5-4.


http://rebusmetermanagement.cloudapp.net/meterDetail.aspx?meterId=601015883&meterKey=1 - Windows Inter...
http://rebusmetermanagement.cloudapp....
Google
Google
Search
Share
Sign In
Favorites
http://rebusmetermanagement.cloud...
Page
Safety
Tools
conEdison
a conEdison, inc. company
Save
Cancel
REBUS
Technologies
Done
Internet | Protected Mode: On
100%

Update Meter Location

Meter ID: 601015883

Account Number:	1
Service Site Name:	Cols State
Street Address1:	4225 University Ave
City:	Columbus
State:	GA
ZIP Code:	31097
Country:	USA
GPS Coordinates	Lat: 32.50430(Long: -84.9382

Figure 5-4 Site Location

MeterManagementCommon

This project is used for the common Interface that will be used by both the client side library and server-side library projects. This allows both class libraries to implement the same Interface, ensuring that the same calls exist on both the client and server side. The methods of the Interface are shown in the table.

Table 5-1 Interface Methods

Interface	Method	Description
IMeter	GetMeterData	Used to return a list of meters based on the input selection criteria.
IMeter	GetSiteLocationData	Used to return the service site based on the meterKey.
IMeter	UpdateSiteLocation	Used to update the service site.

MeterManagementDataClient

This client side library contains the Meter object, which implements the IMeter Interface from the common library. In addition, the Channel class, in this project, is used for setting up the channel to the Service Bus. These objects are called by the cloud-based Web application and are used to manipulate application data. This subsystem is responsible for communicating with the Service Bus on behalf of the MeterManagement Web application.

MeterManagementDataServer

This server side library contains a server-side object that implements the IMeter Interface defined in the common library. The signatures of these objects are the same as the corresponding object in the client side library (as required by the

Interface). The server side library is responsible for the interaction with the database. The data will flow to and from these objects using the Azure Service Bus. This library is used by a server-side Windows service (described below) so it will always be running. It must be on an on-site machine that has access to the database.

MeterManagementService

This is a Windows service running on the database server. A Windows service is a long-running application that does not generally require user intervention (44). In this case, the service is an application that is always running and provides access to the server side library. This service is not required to run on the same machine as the database, but it must have access to the database. This Windows service will manage the connection to the Azure Service Bus and will use the MeterManagementDataServer project to access the database.

Data Flow

The ASP.Net Web application will be running inside the Windows Azure cloud. This application will drive the data flow by making request to the Azure Service Bus. The Windows service, running the server-side libraries, will listen from requests from the Service Bus, fetch any data required, and return the data to the caller using the Service Bus. All communication with the Service Bus will occur on the standard HTTP port, 80. Figure 5-5 illustrates this process.

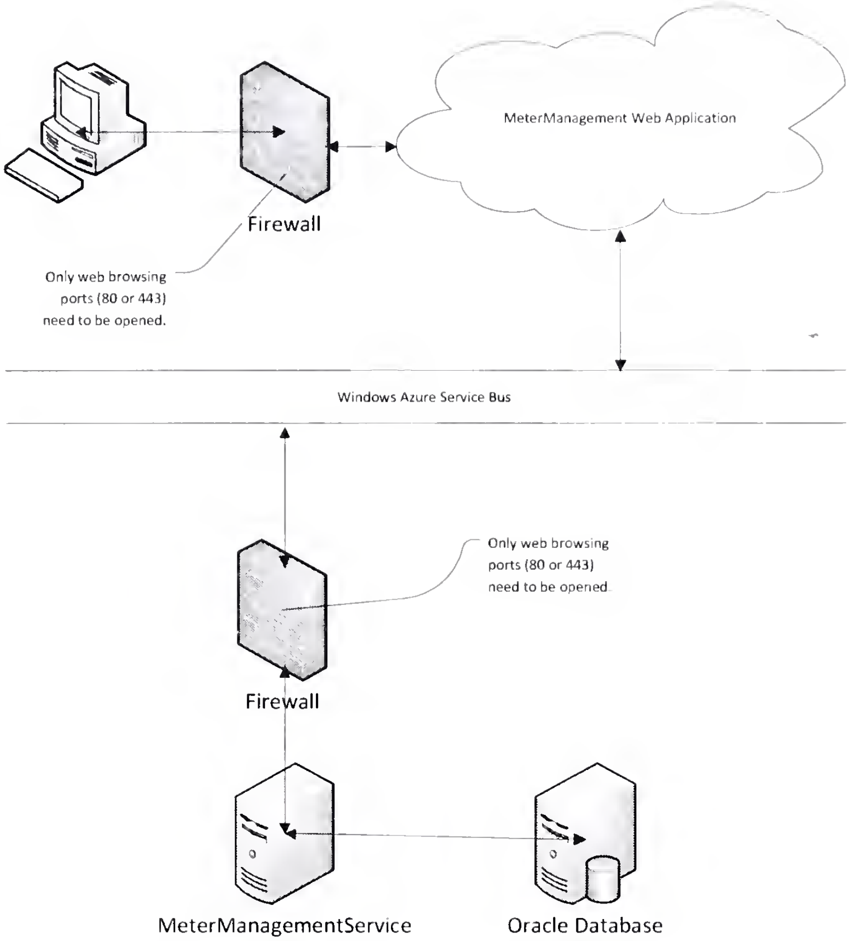

MeterManagement Web Application
Firewall
Only web browsing ports (80 or 443) need to be opened.
Windows Azure Service Bus
Only web browsing ports (80 or 443) need to be opened.
Firewall
MeterManagementService
Oracle Database

Figure 5-5 Data Flow

Process Flow

This section will describe the process flow of the Meter List Web page. First, the user requests the home page of the application, which will display the meter list. The ObjectDataSource on this Web page makes a call the GetMeterData method of the MeterManagementDataClient. The MeterManagementDataClient opens a channel to the Service Bus, which is returned as an IMeterChannel object (defined in the MeterManagementCommon project). The GetMeterData method, of this channel object is invoked which make a request to the Azure Service Bus.

On the server, the MeterManagementService application has already connected to the Service Bus and is listening for requests. When a request is received, the GetMeterData method in the MeterManagementDataServer is invoked. This method retrieves the data from the Oracle database and returns a serializable DataTable, which is returned to the client using the Azure Service Bus. A sequence diagram describing this process is shown in figure 5-6.

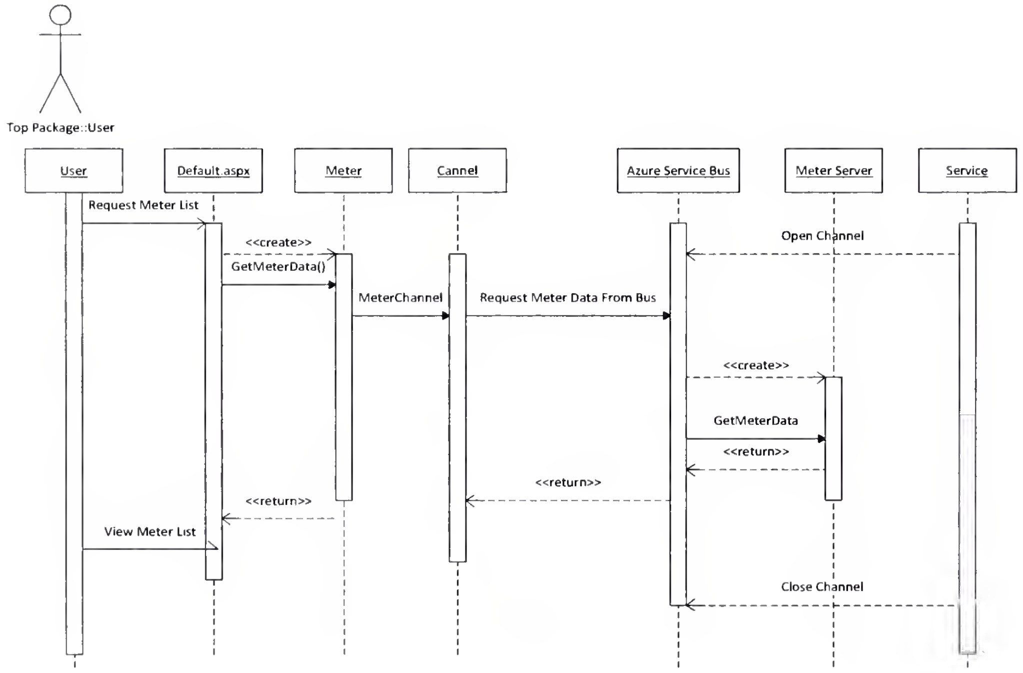


Figure 5-6 Sequence Diagram

Summary

This chapter proposes a solution to the data security problem identified in Chapter 2. This solution calls for implementing a service that connects to the Azure

Service Bus using the BasicHttpRelayBinding, which only requires the standard HTTP ports for communication. This service will run outside of the cloud and will have access to the data source using any method and ports required. The Web application, running in the cloud, will then interact with the Service Bus for all data interaction.

This chapter then describes an implementation of the solution to the problem. The implementation uses a window service that continuously runs and is connected to the Service Bus. This service, which is outside of the cloud, is running on the same server as an Oracle database storing the data. A Web application was created that runs in the Windows Azure cloud and uses the Service Bus for all data interactions.

Chapter 6: Evaluation

Chapter 2 presented possible solutions for making data available to a Cloud Computing application. These solutions included strategies for protecting data stored in the cloud as well as methods for storing data outside of the cloud. This section explains why the solution described in Chapter 4 may be a better solution.

Data in the Cloud

In the current research section of Chapter 2, four methods were identified for securing data stored in the cloud. With the data stored in the cloud, there is a high probability that performance will be enhanced. However, when the data is stored in the cloud many of the security and legal problems, mentioned in Chapter 2 exist. The data remains outside of the control of the cloud consumer, and ownership and jurisdiction issues still exist. An organization must trust a cloud provider to implement the security layer it claims to use. In addition, in many cases the cloud provider itself can circumvent the security procedures.

One method involved encrypting data, which could prevent the cloud provider from accessing the data. However, encrypting data stored in the cloud presents problems when searching or indexing the data. Once the data is encrypted, it must be decrypted before it can be searched or indexed (26). When searching large amounts of data, the overhead to decrypt the data can be prohibitive.

The proposed solution does not store data in the cloud. The data is stored in the consumer's data center. This leaves the data security responsibilities with the consumer organization. This approach alleviates many of the legal and compliance related issue. It also makes clear who has responsibility for data security.

Data External from the Cloud

Storing data outside of the cloud solves many of the problems presented in Chapter 2. However, this approach exposes a new problem: how does the cloud-based application access the data outside of the cloud?

One method for accessing data outside of the cloud is to connect to the server housing the data, in the same manner a non cloud-based application would. This is a straightforward solution but it requires opening firewall ports to allow connectivity between the data source and the cloud-based application. Some examples of this are opening port 1521 for the Oracle listener or enabling file sharing when data resides on a mapped drive. This approach is fraught with security dangers and may lead to network vulnerabilities. In addition, many network administrators disapprove of opening ports (33). The solution presented in this thesis is superior because it only required the standard HTTP/HTTPS ports to be open, which are typically open anyway (45).

A solution to opening ports would be to use a Web-service to access the data. The Web Service can be configured to use only the HTTP/HTTPS ports, which overcomes the problems with firewall exceptions. This overcomes the problem with firewall exceptions, but it presents the problem of how clients find the endpoints they need to connect to the Web Service. For a small project, this may not be an issue, but for a large organization with many cloud-based applications, this could result in duplication of Web-services. A user or developer could no longer inspect a shared drive or query the data structure looking for data that is needed. Some type of registry would need to be implemented that would help clients look-up the available Web Services (33). That registry is the Windows Azure Service Bus. Endpoints on the Service Bus can be made discoverable making it easy for client applications to find and connect to them. The solution in this thesis uses the Service Bus; therefore, discoverable endpoints become an option when they are needed.

Summary

Cloud Computing brings a new economic and programming paradigm to the world of software development. These cloud based applications reduce the need for on-site hardware as well as full-time IT support staff. In addition, the Cloud Computing architecture moves processing power into the cloud, which allows powerful applications to run on a myriad of mobile devices regardless of their operating system (46).

The solution presented in the thesis has several advantages over existing solutions. However, other solutions may be better depending on the needs of an application. For example, storing data in the cloud may be the only viable solution for some data intensive applications. In addition, with large data sets, the overhead associated with the Service Bus, WCF, and SOAP may prove to be too costly. A thorough evaluation of the specific problem, combined with knowledge of a range of potential solutions, will help guide the technical decision makers to a wise decision.

Works Cited

1. Schneier, Bruce. Cloud Computing. *Schneier on Security.* [Online] Jun 4, 2009. [Cited: Apr 20, 2010.] http://www.schneier.com/blog/archives/2009/06/cloud_computing.html.

2. Knorr, Eric and Gruman, Galen. What Cloud Computing Really Means. *InfoWorld.* [Online] [Cited: March 12, 2010.] http://www.infoworld.com/d/cloud-computing/what-cloud-computing-really-means-031.

3. Stolvoort, Anette. Utility Computing. *IBM Technology Update.* [Online] Jun 6, 2007. [Cited: Apr 20, 2010.] http://www-05.ibm.com/nl/events/presentations/Utility_Computing.pdf.

4. Ivanov, Ivan. Utility Computing: Reality and Beyond. [book auth.] Joaquim and Obaidat, Mohammad Filipe. *E-business and Telecommunications: 4th International Conference.* Barcelona, Spain : Springe, 2008, pp. 16-29.

5. Perry, Geva. How Cloud & Utility Computing Are Different. *The Apple Blog.* [Online] Fenruary 28, 2008. [Cited: March 31, 2010.] http://gigaom.com/2008/02/28/how-cloud-utility-computing-are-different/.

6. Danieldon, Krissi. Distinguishing Cloud Computing from Utility Computing. *Ebiz.* [Online] March 26, 2008. [Cited: March 12, 2010.] http://www.ebizq.net/blogs/saasweek/2008/03/distinguishing_cloud_computing/

7. Litoiu, M. and Iszlai, G. *2009. 3rd International Workshop on Cloud Computing. In Proceedings of the 2009 Conference of the Center For Advanced Studies on Collaborative Research (Ontario, Canada, November 02 - 05, 2009). P. Martin, A. W. Kark, and D. Stewart, Eds. CASCON '09. ACM, New York, NY, 303-303. DOI= http://doi.acm.org/10.1145/1723028.1723068*

8. Chong, Frederick and Carraro, Gianpaolo. Architecture Strategies for Catching the Long Tail. *MSDN.* [Online] Apr 2006. [Cited: Apr 6, 2010.] http://msdn.microsoft.com/en-us/library/aa479069.aspx.

9. Reese, George. The Economics of Cloud Computing. *O'Reilly Community.* [Online] Oct 24, 2008. [Cited: March 29, 2010.] http://broadcast.oreilly.com/2008/10/the-economics-of-cloud-c.html.

10. Marshall, David. Beaver, Stephen S. McCarty, Jason W. *VMware ESX Essentials in the Virtual Data Center.* Boca Raton, FL : Auerbach Publications, 2009.

11. Moreno-Vozmediano, R., Montero, R. S., and Llorente, I. M. 2009. Elastic management of cluster-based services in the cloud. In *Proceedings of the 1st Workshop on Automated Control For Datacenters and Clouds* (Barcelona, Spain, June 19 - 19, 2009). ACDC '09. ACM, New York, NY, 19-24. DOI= http://doi.acm.org/10.1145/1555271.1555277

12. Capital Investment. *Investor Words: The Biggest, Best Investing Glossary on the Web.* [Online] [Cited: Apr 20, 2010.] http://www.investorwords.com/712/capital_investment.html.

13. Current vs. Capital Expenses. *NOLO.* [Online] [Cited: April 12, 2010.] http://www.nolo.com/legal-encyclopedia/article-30223.html.

14. 10 Benefits of Cloud Computing for Small Businesses. *Pluging Into the Cloud.* [Online] Cloud Tweaks. [Cited: July 15, 2010.] http://www.cloudtweaks.com/2010/01/10-benefits-of-cloud-computing-for-small-businesses/.

15. Barriers To Entry. *Economics: Helping to Simplify Economics.* [Online] [Cited: Apr 20, 2010.] http://www.economicshelp.org/microessays/markets/barriers-entry.html.

16. Amazon Elastic Compute Cloud. *Amazon Web Services.* [Online] [Cited: March 31, 2010.] http://aws.amazon.com/ec2/.

17. Kommalapati, Hanu. Windows Azure Platform for Enterprises. *MSDN Magazine.* [Online] February 2010. [Cited: March 31, 2010.] http://msdn.microsoft.com/en-us/magazine/ee309870.aspx.

18. Pricing. *Windows Azure Platform.* [Online] [Cited: March 31, 2010.] http://www.microsoft.com/windowsazure/pricing/#windows.

19. Google Apps Reseller Program. *Google Apps.* [Online] Google, 2010. http://www.google.com/apps/intl/en/business/resellers/opportunity.html.

20. Reliable, secure online applications wherever you work. *Google Apps.* [Online] [Cited: March 31, 2010.] http://www.google.com/apps/intl/en/business/index.html.

21. 2007 Microsoft Office system pricing and upgrade information. *Office Online.* [Online] [Cited: March 31, 2010.] http://office.microsoft.com/en-us/suites/FX101754511033.aspx.

22. Wang, Chenxi. Forrester: A Close Look At Cloud Computing Security Issues. *CSO Security and Risk.* [Online] March 2009. [Cited: April 12, 2010.] http://www.csoonline.com/article/496388/Forrester_A_Close_Look_At_Cloud_Computing_Security_Issues?page=1.

23. Farrar, Lara. How safe is cloud computing? *Digital Biz.* [Online] Mar 12, 2010. [Cited: Apr 26, 2010.] http://www.cnn.com/2010/TECH/03/12/cloud.computing.security/index.html.

24. Lockheed Martin, LM Cyber Secuirty Alliance, Market Connections INC. Awareness Trust Security. *Lockheed Martin WebSite.* [Online] April 2010. [Cited: April 25, 2010.] http://www.lockheedmartin.com/data/assets/isgs/documents/CloudComputingWhitePaper.pdf.

25. Privacy Rights Clearinghouse / UCAN. The Privacy Implications of Cloud Computing. *Privacy Rights Clearinghouse.* [Online] March 2009. [Cited: April 12, 2010.] http://www.privacyrights.org/ar/cloud-computing.htm.

26. Chow, Richard and Masuoke, Ryusuke. *Controlling data in the cloud: outsourcing computation without outsourcing control.* New Youk, NY : ACM, 2009. pp. 85-90.

27. HIPAA Security Series. *U.S. Department of Health & Human Services.* [Online] December 10, 2007. [Cited: April 12, 2010.] http://www.hhs.gov/ocr/privacy/hipaa/administrative/securityrule/smallprovider.pdf.

28. Morris, Glen Emerson. Cloud Computing Has Dark Lining. *Advertising & Marketing Review.* Denver, CO : CSC Publishing, March 2009.

29. Ossian, Kathryn L. Cloud Computing: Forecasting a Storm of Potential Risks. *Everything Digital.* [Online] Apr 15, 2010. [Cited: April 26, 2010.] http://www.corpmagazine.com/SpecialInterests/EverythingDigital/tabid/805/itemid/1438/Default.aspx.

30. Brunete, Glenn. Cloud Saftey Box. *Projet Kenai.* [Online] Jun 11, 2008. [Cited: Apr 26, 2010.] http://kenai.com/projects/s3-crypto/pages/Home.

31. Tidmarsh, David. ITS delays switch to Gmail. *Yale Daily News.* [Online] March 30, 2010. [Cited: April 12, 2010.] http://www.yaledailynews.com/news/university-news/2010/03/30/its-delays-switch-gmail-community-input/.

32. Skonnard, Aaron and Brown, Keith. An Introduction to Windows Azure platform AppFabric for Developers. *Microsoft Azure Platform.* [Online] November 2009. [Cited: April 8, 2010.] http://www.microsoft.com/windowsazure/whitepapers/.

33. Chappell, David. Introducing the Windows Azure Platform. *Microsoft Azure Platform.* [Online] Decemeber 2009. [Cited: April 08, 2010.] http://www.microsoft.com/windowsazure/whitepapers/.

34. Fielding, R. T. *Architectural Styles and the Design of Network-Based Software Architectures. Doctoral Thesis.* Irvine, CA : University of California, Irvine, 2000.

35. Rodriguez, Alex. RESTful Web services: The basics. *IBM.* [Online] Nov 06, 2008. [Cited: July 15, 2010.] https://www.ibm.com/developerworks/webservices/library/ws-restful/.

36. Von Schweber, Erick and Von Schweber, Linda. Computing Fabrics. *PC Week Online.* Oct 26, 1998.

37. Chappell, David. Introduction to the Enterprise Service Bus. *Enterprise Service Bus: Theory in Practice.* s.l. : O'Reilly Media, 2008, p. 1.

38. Microsoft Corporation. Windows Communication Foundation Essentials. *MSDN.* [Online] 2010. [Cited: May 21, 2010.] http://msdn.microsoft.com/en-us/library/ee354181(v=MSDN.10).aspx.

39. Microsoft. Windows Communcation Foundation Bindings. *MSDN Library.* [Online] Microsoft, 06 09, 2010. [Cited: 06 17, 2010.] http://msdn.microsoft.com/en-us/library/ms733027.aspx.

40. Configuring System-Provided Bindings. *MSDN Library.* [Online] Microsoft. [Cited: 06 17, 2010.] http://msdn.microsoft.com/en-us/library/ms731092.aspx.

41. WCF Essentials: Contracts. *MSDN.* [Online] 2010. [Cited: July 15, 2010.] http://msdn.microsoft.com/library/ff183866.aspx.

42. IClientChannel Interface. *MSDN.* [Online] [Cited: July 15, 2010.] http://msdn.microsoft.com/en-us/library/system.servicemodel.iclientchannel.aspx.

43. Oracle. Managing Oracle Database Port Numbers. *Oracle Database Installation Guide.* [Online] 2007. [Cited: June 8, 2010.] http://download.oracle.com/docs/cd/B19306_01/install.102/b14316/ports.htm#BEHFDBEE.

44. Microsoft. Introduction to Windows Service Applications . *MSDN.* [Online] 2010. [Cited: July 16, 2010.] http://msdn.microsoft.com/en-us/library/d56de412(VS.80).aspx.

45. AngiSoft. Introduction to SOAP. *gnhiSoft.* [Online] AngiSoft. [Cited: June 18, 2010.] http://www.agnisoft.com/soap/soapintro.htm.

46. Dern, Daniel. Writing Small - Too many platforms can spoil the smartphone app. *IEEE Spectrum.* 2010, Vol. 47, 6.