ATSPM TRAIN THE TRAINER

Derek Lowe
Utah Department of Technology Services
Lead Developer for In-House Development
dlowe@utah.gov

ATSPM History

- We were not aware of the growth this system would require
- Started as a PowerShell script a database and an ASP.Net Web Form web page
- The previous configuration tool was more like a spreadsheet that had difficulty keeping up with changes to the system
  - This also made it difficult to create the new increasingly complex charts
- As features were added it drove some of our architecture DL5
  - Stored Procedures vs ORM (Object/Relational Mapping)
  - Common Library
  - WCF Services
  - Multi Threaded Processing
  - Partitioned Database

Slide 2

DL5 All of the recent changes that were made we were planning on implementing before the decision was made to go open source, however open source helped us to prioritize these changes

Derek Lowe, 1/12/2017

ATSPM History Continued

- Some features have been scrapped while some have been improved and others we plan on reintroducing
  - Cached Reports
  - Data Export
  - Executive Reports
  - System Health
  - Data Archiver

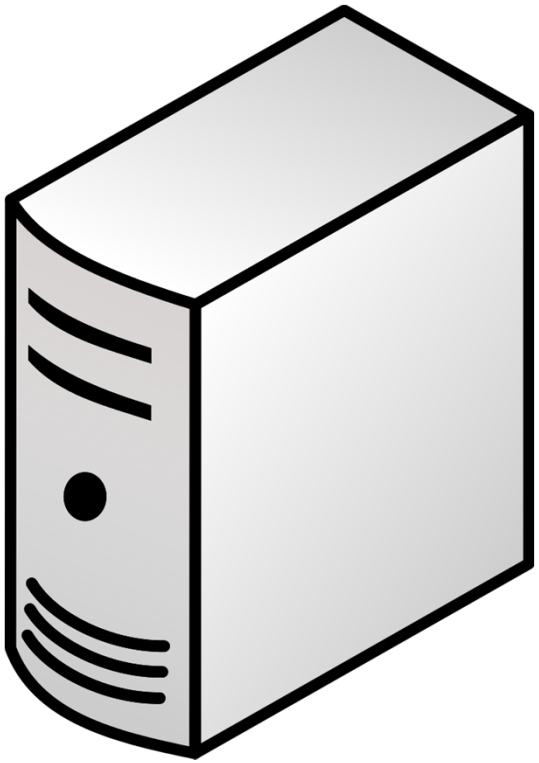
Current Code Architecture

Front End	Middle Tier	Back End
ASP.Net MVC	WCF Services	Microsoft Sql Server
Bootstrap	Entity Framework	
JQuery	Scheduled Tasks	

Why a Multi Tier System?

- UDOT's ATSPM System piggy backs on the main TOC Website so it is essential to not interfere with existing processes
  - Tasks that require extensive processing can be offloaded to other servers
- Much of the speed we see in our system is due to a robust Database server that does not need to compete for resources
- Scalability
- Security

Basic Setup Configuration



- Email Server (Optional)
- Service Host for Speed Listener (Optional)
- Application Server to run scheduled tasks
- Share Drive to store chart images
- Web Server for website and WCF services
- Database

Slide 6

DL1 Derek Lowe, 1/11/2017

UDOT System Configuration

Database Server

Web Server for WCF Services

Web Server for Web Site

Web Server for Web Site

Load Balancer for Web Sites

Email Server

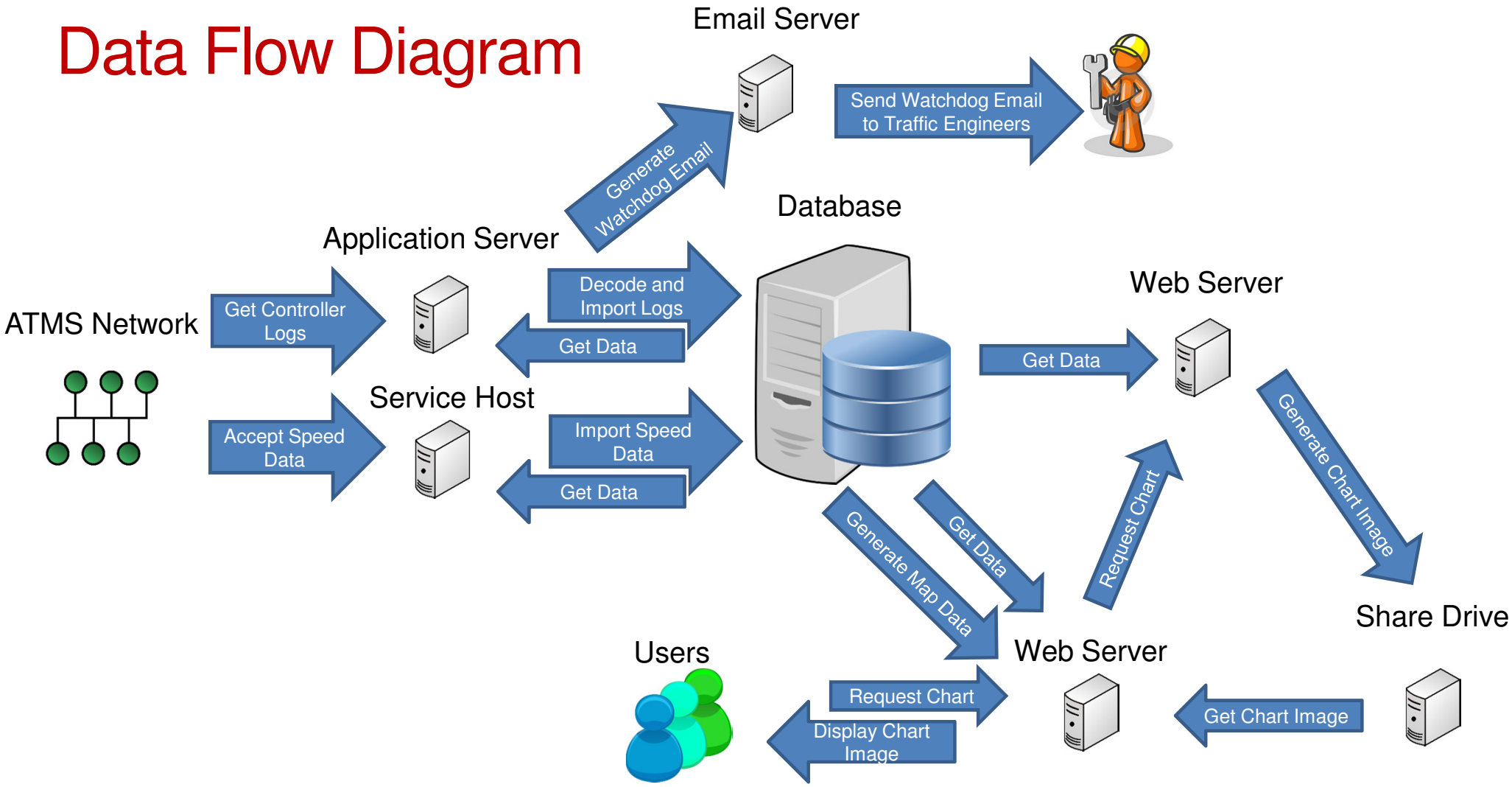
SAN for Database

Host for Speed Listener Service

Application Server for Scheduled Tasks

Share for Chart Images

Data Flow Diagram



Code Design Techniques

- Factory Pattern
  - Used in Repositories
- Repository Pattern
  - Used in the business layer to access the database and to allow unit testing
- Model View Controller
- Entity Framework
  - Code First allows us to spend less time managing database changes
- Inheritance
  - Our signal charts all have the same base object to capture parameters necessary to create the chart
- Class Libraries
  - This facilitates code reuse

ATSPM System Security

- ATSPM Website login uses OWIN Identity by Microsoft
  - Microsoft recommends this run in conjunction with SSL on your site
- ASP.NET MVC leverages anti-forgery tokens for post backs to prohibit Cross-site request forgery (also known as XSRF or CSRF)
- ATSPM code base uses Entity Framework which is a technology recommended to prevent SQL injection

Steps UDOT Takes to enhance security

- UDOT Limits its database login to table specific elevated privileges
- A majority of the UDOT system is behind a firewall
  - Only the website is publicly available
- UDOT uses HP Fortify to test the code base for security vulnerabilities
- The State of Utah performs occasional penetration tests on all state public sites
- Make sure your servers are patched with the latest Microsoft updates
- There is no such thing as a completely secure system however we do our best to follow best practices to ensure a high level of confidence in our system

Overview of OSADP

- OSADP created a private GitHub repository for the ATSPM project
- To collaborate on the ATSPM Project
  - Make a request on the OSADP Site with your proposed changes
  - UDOT will review the request for approval
  - A GitHub branch will be created for your changes
  - Changes will be reviewed and merged with the code trunk

How to Add a New Metric to the Default Charts Page

1. Add an entry into the database in the MetricTypes table with the name of your new chart.
2. Create a class in MOE.Common -> Business -> WCFServiceLibrary that inherits from the MetricOptions class.
3. Add the [DataContract] attribute to the class
4. Mark any property required to build your class with the [DataMember] attribute so that it can be serialized and sent to the WCF service.
5. Create your business logic objects. Make sure that any new tables are created through Entity Framework. All new tables should have corresponding repositories found in MOE.Common -> Models -> Repositories. If new tables are created a new Entity Framework migration will need to be created.
6. From the CreateMetric function in the MetricOptions class you created in step 2 call base.CreateMetric(); Then load your business objects. This function should return a string array with the location of the images through IIS.

Slide 13

DL2 Inheritance, JQuery, CSHTML, Contoller, UDOT Common Library, Database

Derek Lowe, 1/11/2017

How to Add a New Metric to the Default Charts Page (Continued)

7. In the MOEWcfServiceLibrary project open the IMetricGenerator class and add the following for your new MetricOptions class: [ServiceKnownType(typeof(MOE.Common.Business.WCFServiceLibrary.YourNewMetricOptions))]
8. In the SPM project add a new partial view that takes your new MetricOptions class as the model.
9. In the SPM project DefaultChartsController class add a line in the MetricOptions(int id) function that returns your new partial view.
10. In the SPM project DefaultChartsController class add a function that accepts your new MetricOptions class as an argument. This class should open a new client to the metric generator service pass your metric options to it and get back a string array of image locations. It should then return the MetricResults partial view with your populated string array as the model.

How to Add a New Metric to the Default Charts Page (Continued)

11. Modify Index.cshtml file found at SPM->Views->DefaultCharts. Add a line in the scripts section that uses razor to build the url to the DefaultChartsController. This will ensure the url is correct wherever it is installed.
12. Modify the GetMetric.js script found at SPM->Scripts. Add a function that collects the options the user has selected from the web page and passes them to the DefaultChartsController function you created in step 11. Use the GetCommonValues() add the base MetricOptions properties to the collection. Use the GetMetric() function to make the call to the DefaultChartsController.
13. Modify the GetMetric.js script found at SPM->Scripts. Change the $('#CreateMetric').click() function to call the function you created in step 12.

ATSPM TRAIN THE TRAINER

Shane Johnson
Utah Department of Technology Services
Signal System Slave
Code Monkey
shanejohnson@utah.gov

DATA RETRIEVAL USING SQL SCRIPTS

What this means

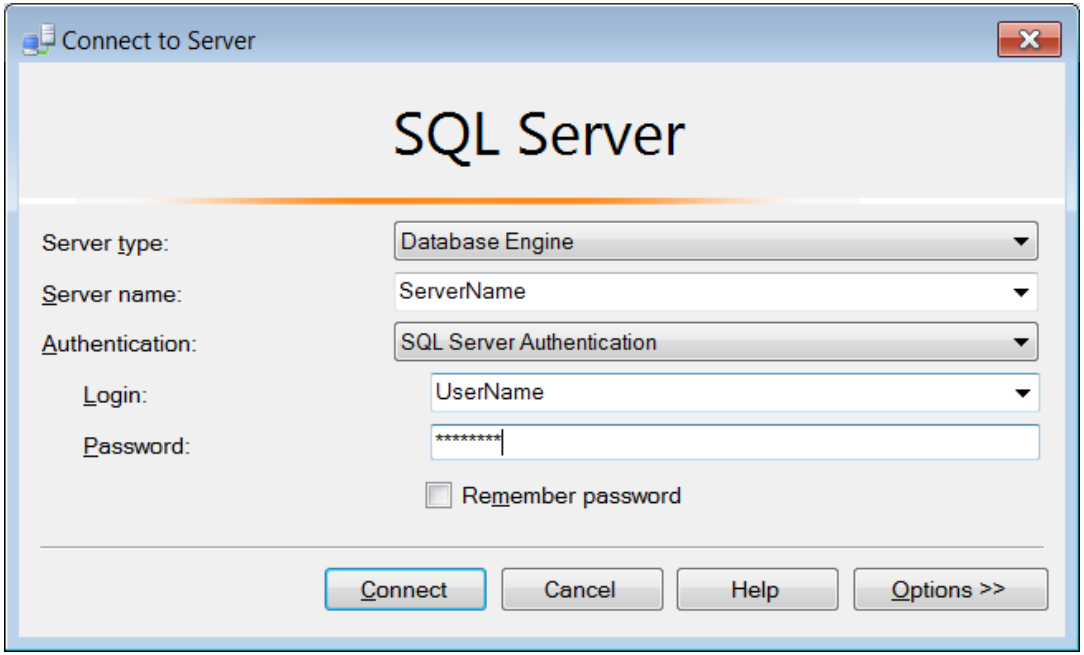
- You can get to the stored records directly without using the SPM webpage
- You must use an SQL query client that works with MSSQL
- The **Microsoft SQL Server Management Studio** is a popular one
- It can be downloaded from the Microsoft website

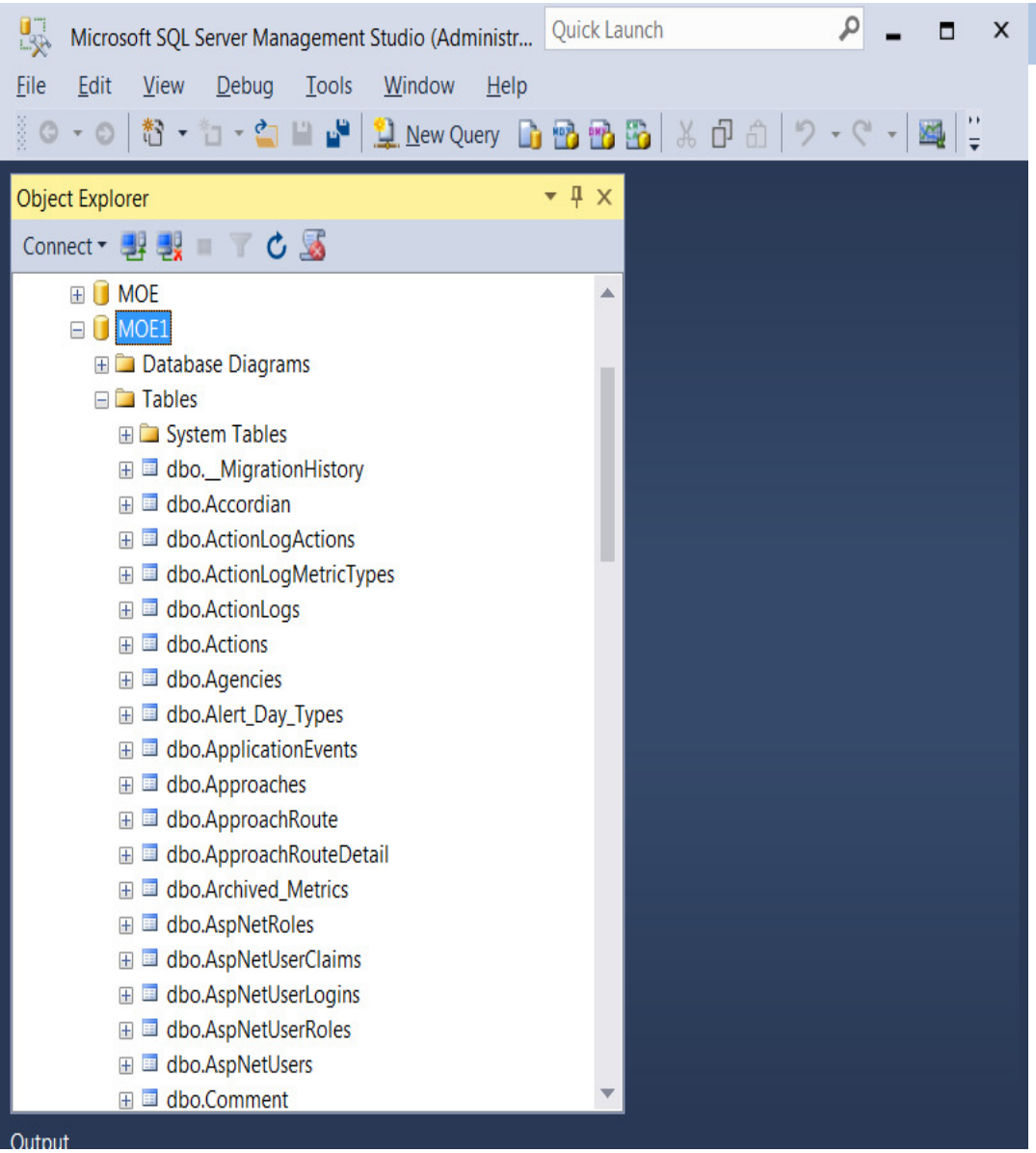
Why you would do this

- Troubleshooting. It can be helpful to know if the records are in the database at all
- Quality assurance. The charts must accurately reflect the records
- Respond to public information requests
- Get information without creating a chart for it

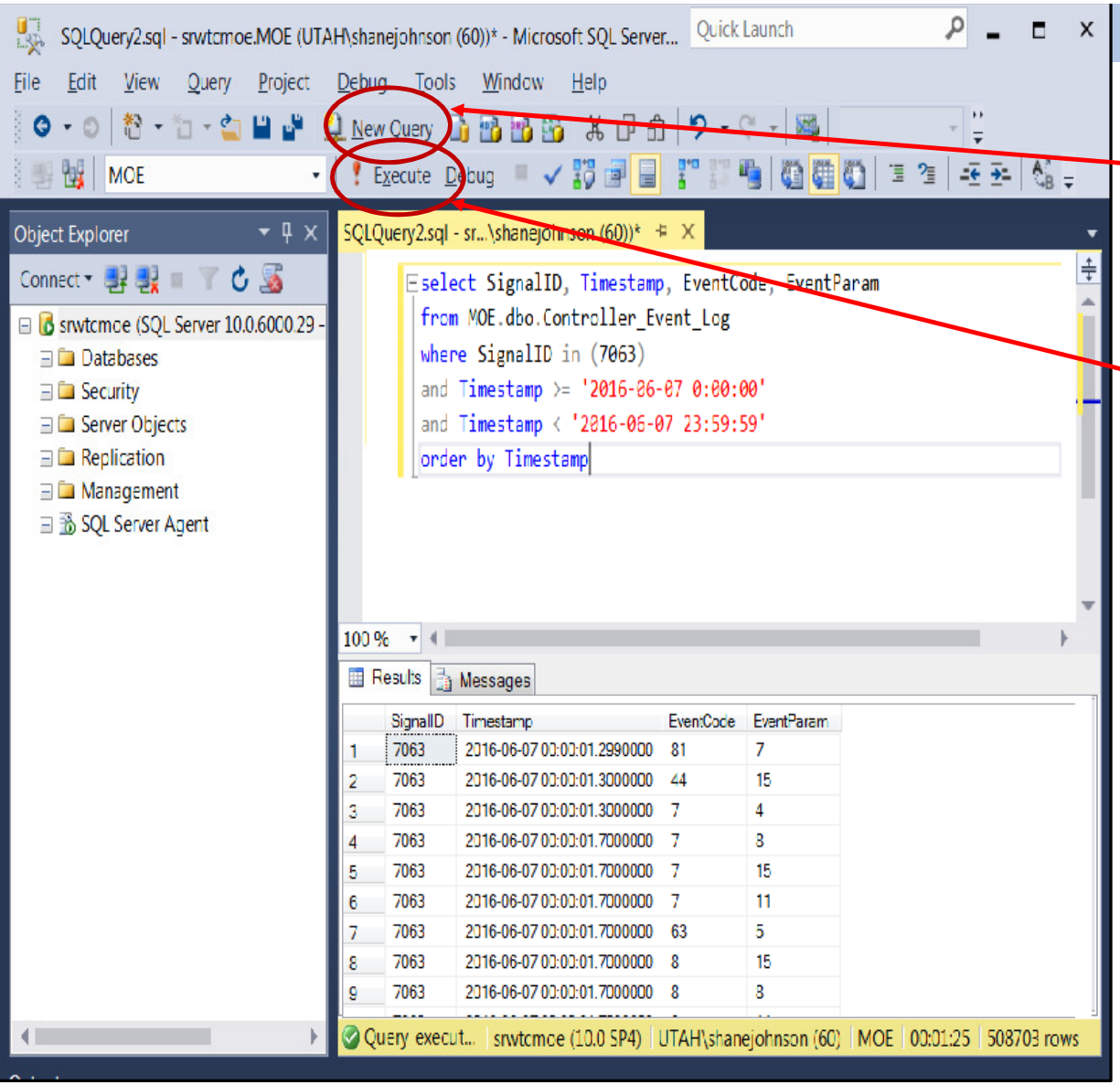
How you would do this

- Open the SQL client and connect to the server.
- You must provide the server name and account credentials





- Select the database from the “Object Explorer” pane on the left
- You can use the selection tree to view the tables and other components of the database from here
- This client will default to the “Master” system database, which won’t do us much good.



- Click the “New Query” button at the top
- Type in your query in the right pane
- Click the “execute” button to get the results

Some Useful Queries

```
select SignalID, Timestamp, EventCode, EventParam
from MOE.dbo.Controller_Event_Log
where SignalID in (7063)
and Timestamp >= '2016-06-07 0:00:00'
and Timestamp < '2016-06-07 23:59:59'
order by Timestamp
```

This one will get all of the records for SignalID 7063 Between '2016-06-07 0:00:00‘ and '2016-06-07 23:59:59'

It is useful for public information requests

Some Useful Queries

```
select SignalID, Timestamp, EventCode, EventParam
from MOE.dbo.Controller_Event_Log
where SignalID in (7055)
and Timestamp >= '7/16/2014 14:30'
and Timestamp < '7/16/2014 18:45'
AND EventCode in (82)
-- and (EventParam in (34,35,36,40,41,42,43,46,47,48,49,10,11))
order by Timestamp
```

This query get all of the detector activations (event 82) for the signal 7055 on the detector channels listed

You can use it to get detector counts for specific channels

Other hints

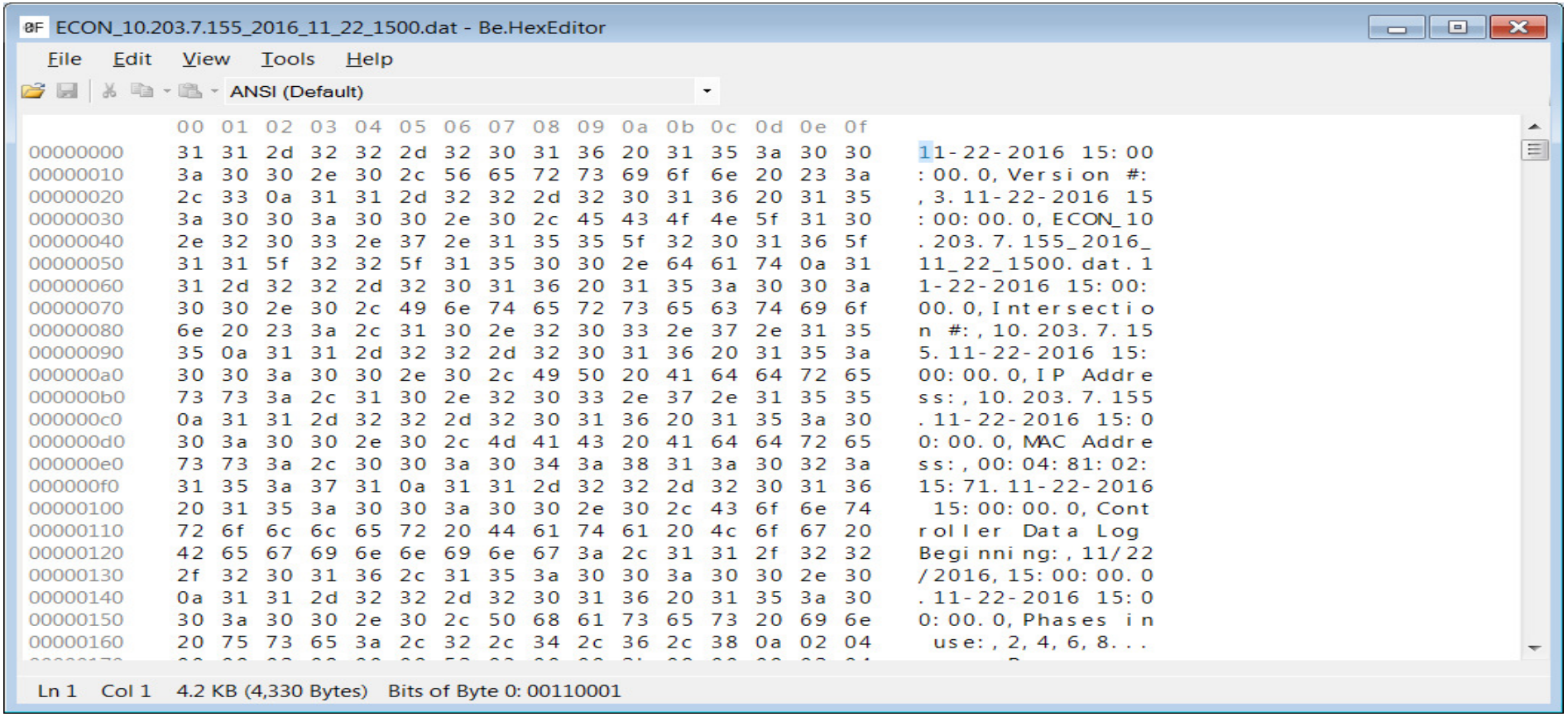
- “Controller_Event_Log” has all of the event data
- SignalIDs and DetectorIDs are strings, even if they look like numbers
- That means they should be enclosed like *‘SignalID’* in a query
- Signals can have more than one approach, but an approach can only belong to one signal
- Approaches can have more than one detector, but a detector can only have one approach
- A phase number and direction is assigned to every approach

DATA LOG TRANSLATORS

What they are:

- Software utilities written by the controller manufacturers
- They all use a command line interface, which make them easy to use in scripts

Why we need them:

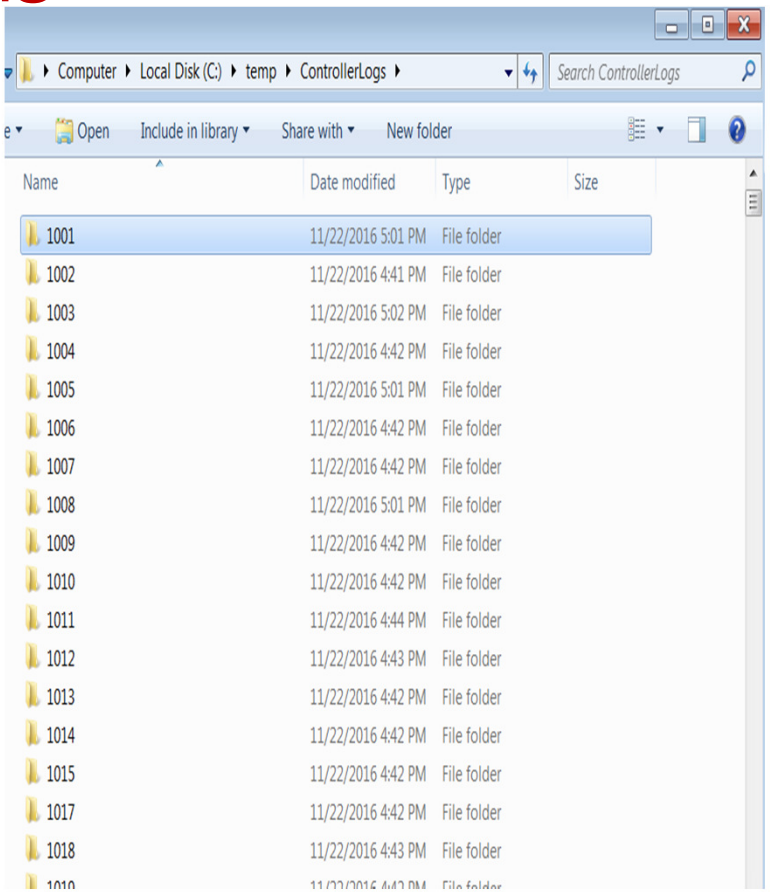


Where to get them

- We do not put the translator programs on the OSADP website, as they are not ours to give out
- Contact the controller manufacture to obtain their translator program

Adding the SignalID to the records

- The SPM system requires controllers to be identified with a SignalID
- Each event record in the database must have a SignalID
- The controllers have NO IDEA what their SignalID is
- Therefore, that information is completely missing from the event logs.
- All of the “decoder” programs in the ATSPM project Get the SignalID from the directory name where the logs are stored.



Adding the SignalID to the records

TRAF_0018_2014_02_25_0900.csv

```
 7  2/25/2014 09:00:20.3, 7, 3
 8  2/25/2014 09:00:20.3, 7, 7
 9  2/25/2014 09:00:20.3, 8, 3
10  2/25/2014 09:00:20.3, 8, 7
11  2/25/2014 09:00:20.3, 5, 3
12  2/25/2014 09:00:20.3, 5, 7
13  2/25/2014 09:00:20.1, 2, 3
14  2/25/2014 09:00:20.1, 2, 7
15  2/25/2014 09:00:20.1, 43, 3
16  2/25/2014 09:00:20.1, 43, 7
17  2/25/2014 09:00:20.1, 63, 4
18  2/25/2014 09:00:20.1, 63, 8
19  2/25/2014 09:00:23.5, 9, 3
20  2/25/2014 09:00:23.5, 9, 7
21  2/25/2014 09:00:23.5, 10, 3
22  2/25/2014 09:00:23.5, 10, 7
23  2/25/2014 09:00:23.6, 64, 4
24  2/25/2014 09:00:23.6, 64, 8
25  2/25/2014 09:00:25.1, 65, 4
26  2/25/2014 09:00:25.1, 65, 8
27  2/25/2014 09:00:25.2, 12, 3
28  2/25/2014 09:00:25.2, 0, 4
```

Before the Decoder Program

5118.csv

```
1  5118,2015-03-12 05:50:46.1000000,81,11
2  5118,2015-03-12 05:50:46.1000000,82,12
3  5118,2015-03-12 05:50:46.2000000,81,12
4  5118,2015-03-12 05:50:46.3000000,0,4
5  5118,2015-03-12 05:50:46.5000000,31,1
6  5118,2015-03-12 05:50:46.5000000,33,1
7  5118,2015-03-12 05:50:46.5000000,65,1
8  5118,2015-03-12 05:50:46.5000000,11,2
9  5118,2015-03-12 05:50:46.5000000,12,2
10 5118,2015-03-12 05:50:46.5000000,61,2
11 5118,2015-03-12 05:50:46.5000000,32,3
12 5118,2015-03-12 05:50:46.5000000,65,3
13 5118,2015-03-12 05:50:46.5000000,1,4
14 5118,2015-03-12 05:50:46.5000000,61,4
15 5118,2015-03-12 05:50:46.5000000,2,6
16 5118,2015-03-12 05:50:46.5000000,11,6
17 5118,2015-03-12 05:50:46.5000000,12,6
18 5118,2015-03-12 05:50:46.5000000,0,8
19 5118,2015-03-12 05:50:46.5000000,1,8
20 5118,2015-03-12 05:50:46.5000000,82,21
21 5118,2015-03-12 05:50:46.6000000,82,11
22 5118,2015-03-12 05:50:47.1000000,82,10
23 5118,2015-03-12 05:50:47.3000000,44,6
24 5118,2015-03-12 05:50:47.3000000,81,10
25 5118,2015-03-12 05:50:47.3000000,81,15
```

The SignalID has been added to the table

After the Decoder Program

Intelight MaxTime

- Does not need a decoder.
- The records are retrieved by connecting to a webservice on the controller.
- The webservice delivers an XML document containing the events.
- The XML document need never be saved anywhere, it can be altered in memory and imported directly into the database.
- The SignalID is retrieved from the database and is matched to the proper controller by IP Address.

Econolite ASC3 & Cobalt

- UDOT has a lot of these controllers, so we wrote a custom decoder to handle them.
- Our decoder takes the binary .DAT files retrieved from the controller, translates them in memory and stores the records in the database without saving them to .CSV files first.
- By skipping the intermediary .CSV, the process is fast enough to store records from 1500 different controllers in about 5 minutes.

Siemens and Trafficware

- These translators convert the .DAT files into readable .CSV files.
- The executables you will find in the ATSPM solution will use the translator to convert ALL of the files it can find
- It will read the .CSV files into memory, add a column for the signal ID and populate the column
- The records are then sent to the database

Peek

- The log files are stored on the controller as compressed .gz archives to save storage space
- The files, therefore, must be decompressed, then translated to .CSV
- This requires the use of a gzip program in addition to the translator program
- After that, the .CSV files are updated and imported into the database as normal

SIGNAL CONTROLLER SECURITY

The first step is to admit you have a problem

- The controllers are vulnerable, and have been for a long time
- There are multiple avenues for attack
- The consequences of a compromised system range from inconvenient to embarrassing to life threatening
- Most controller manufacturers refuse to take this problem seriously
- Even worse, most agencies don't take it seriously, either

We can not prevent every attack

We can only hope to make it so difficult that it is not worth the effort

Physical security

- If the bad guy can touch your hardware, you have lost
- The most straight-forward attack, simply flipping the breaker switches, can have a very large impact
- Cabinet keys are helpful, but the same key will open almost every cabinet in the country
- Most controllers will support a “Cabinet Door Open” alarm. We should be making better use of it

Network Security

- The signals network can be compromised in a lot of exciting ways
  - Cabinet switches
  - Conduit access vaults
  - Hub buildings
  - Wireless connections of any kind (puck detectors, SSR links, wireless radio)

- Once the network is compromised, there is a significant threat to your entire system.

What can be done

- Make sure the basic encryption and security setting are configured on your network devices
- Use MAC address filtering everywhere possible. It isn't foolproof, but it is better than nothing
- Turn off switch ports that are not supposed to be used
- Segment your network into subnets and VLANs. Keep the segments restricted by function.
- Use a network monitoring system with intrusion detection.
- Hide your IP addresses.

The controllers are vulnerable

- Most controllers have almost no security controls
- FTP usernames and passwords are sent over clear text.

  That means almost anyone with network access can read them.
- The FTP credentials, in most cases, provide root-level access to the controller file system.

  That means anything on the controller can be altered or deleted.
- The same credentials will give you access to every controller of that type, anywhere.
  - I know the password for the McCain controller on my desk. I can alter any McCain controller in the world if I can reach it. No one changes those passwords.

FTP isn't the only problem

- Many controllers use Telnet
  - Telnet has all the same problems as FTP, and it usually comes with a "reboot" command.
- NTCIP is a widely published protocol. It has no security controls
  - Pattern change commands, including FREE and FLASH can be issued over NTCIP.
- Many controllers can be configured using SNMP strings. The version of SNMP we use has no security controls
- Many controllers are still vulnerable to network packet storms that will cause the controller to fail into FLASH.

Think Big

A clever attack can:

- Radically disrupt traffic over your entire system
- Can disrupt your entire system from a single network access point
- Can reoccur, indefinitely (or at least until the vulnerabilities are fixed or the source is removed)
- Look like intermittent, perhaps even random, failures
- Lie dormant until the least convenient time (during a large event, for example).

The Human Factor

Proper security is expensive, inconvenient and time-consuming

- No agency wants to have multiple keys for their different cabinets
- No one wants to keep track of or change the passwords for all of their field devices
- Understanding the security settings of the network devices takes more time than most agencies have
- The more secure a system is, the more difficult it is to use.
- Building in security controls will be difficult for the controller manufacturers
- There have, so far, been no major successful attacks on a traffic signal system

I Know All Of This

It Does Not Matter

There is hope

- Effective security means making the payoff not worth the price
- We don’t have to do everything, all at once, in order to achieve a result
- Rapid response is good, prevention is better
- Get the low hanging fruit first.
  - Hide your IP Addresses
  - Secure your wireless devices
  - Change all the default passwords for your network devices
  - Segment your network into VLANS and subnets
  - All of this is under your control, and will have a minimal impact on operations

It won't change until we change it

- Stop buying controllers with obvious security vulnerabilities
- Change your procurement requirements to insist on secure controllers
- Pressure the standards committees to put security controls in the requirements

ADDING DATA LOGS TO THE SERVER

Why you would do this

- Someone sent you log files
- You have found log files being stored in the wrong place
- You have collected files from a Raspberry Pi device

Working with a Raspberry Pi

- Howell Li from Purdue has created a guide on how to configure a Raspberry Pi to collect log files from a controller and synchronize the controller clock with a GPS receiver
- The guide can be found online here: ftp://itsdev1.ecn.purdue.edu/RaspPi_Setup.pdf

Working with a Raspberry Pi

- To get to the log files on a Raspberry Pi, you must use an FTP or SSH client to connect to the device
- You can not take the SD card out of the Raspberry Pi and plug it into your laptop. You will not be able to get to the files that way
- My personal favorite client is the Filezilla FTP client. You can find it here: https://filezilla-project.org

Once you have the files…

- They still need to be translated, given a SignalID and imported into the database.
- The easiest way is to create a sub-directory with the proper SignalID in the same directory where your controller logs are already stored, and put the logs in there.
- The translator program will do all the work for you from there