Elastic Search Talk – Script

# PROACTIS

We have a bit of a problem at work. We have been successfully being buying up other companies around the world. This now means that we have 20+ products, written in several languages / frameworks spread across multiple global data centres.

To give a high level of service to our customers our ops team has work with the development teams to ensure

High Availability

Complicence (GDPR etc)

Developers need to know what the applications are doing, but don’t have access to the servers.

This means that we need a standard way of logging information from all our products/servers. Coupled with the fact that some of the products in the group are already using Elasticsearch it made sense to head in that direction.

Recently we’ve been looking at other solutions such as Elastic Search across our group in order to solve some of our performance issues.

Who has used Elastic Search before?

Who has used a NoSql database?

Who has used a relational database?

# Tonight

I’ve worked in the software industry for the past 20 years, coding in C# and mainly using Microsoft’s SQL Server for storage.

Tonight we will be looking at Elastic Search from the point of view of a dot net developer. If this was a Java meetup then this would be a different talk!

What issues do we have with SQLServer which we like to solve?

What is elastic search?

How do we get started/install/add data/query it?

Does it solve our problems?

We only have an hour so it’s more of a getting started guide!

# The Story

So you are doing some coding, getting stuck as usual which requires you to use **google** and **stackoverflow** to find various answers

However, then disaster strikes, you lose your internet connection

You can no longer use stack overflow.

# The Application

Being a developer you decided that you can solve this problem by….

1. Downloading a sql server database containing a 10GB of Stackoverflow Q+As
2. And writing a windows forms application to query it.

Open code and scope to project

Demo application, search for c#, sql and unicorn

Show number of results per page

Show wildcard – case insensitive search

# Architecture

Open Demo SQL.txt in SSMS

Look at the database in SSMS (StackOverflow2010)

Point out the size and main tables (Posts and Comments)

Our tables just have indexes on the ID

Do count(\*) to get the number of rows

select rowcnt from sysindexes where OBJECT\_NAME(id) = 'Posts'

We could create indexes, but on which columns, as we might want to search by any of them.

Even then we can’t create an index on the main content field.

We want to do a case-insensisitve search so need to convert ever thing to lower case. Also indexes can’t be used with do are doing a %wild card search

We could use full text search in SQLServer, but it’s limited in features and performance.

Finally, scale, the full stack overflow database is over 130GB, without any indexes. You can split sql server over multiple database but there are complications.

# The Problems

So to summarise, we have three problems

1. The ability to be able to search across multiple fields
2. The ability to be able to search within a block of text
3. The ability to scale out across multiple machines

# Introducing Elastic Search

The products are often referred to as either ELK Stack / Elastic Search

Elastic Search is built on top of lucence.

# E is for Elastic Search

Elastic search is the main query engine built on Lucence. It is typically accessed via REST calls.

L is Log Stash

This gives us an easy way to feed and transform log files into Elastic Search

K for Kibana

This is a visualisation tool build on top of elastic search

# Getting Started

docker pull sebp/elk

docker run –p 5601:5601 –p 9200:9200 –p 5044:4044 –it -–name elk sebp/elk

This starts running

Elastic Search (9200)

Log Stash (5044)

And Kibana (5601)

We will just look at Elastic Search for the moment.

Show http://localhost:9200/

We can see the version number and basic cluster info.

# REST

Open and scope to **ElasticSearchRESTDemo** tool

System Demos -> Overview

System Demos -> Cluster Health

Communication can be done via REST calls, for example to view the health of our 1 node cluster

<http://localhost:9200/_cluster/health>

Show code

Add document -> Describe URL format -> PUT method -> Schema less

Get Document -> Same URL -> GET method

Can re-add the first document -> version number increases

Add and Get a second document with a different URL/ID

Delete 2nd document – gets then returns a 404

If we wanted to store data about a different type of meetup, we might be tempted to use a different type within the index. However as you can see this fails. The advice about the use of indexes/types as changed. You shouldn’t think of types as tables, but rather a subset of rows in the same index

Delete entire index

Bulk Import action-data repeat – watch carriage returns – fails second time

Second bulk command – different actions. Note no data for delete, different format for update

LOAD STACKOVER FLOW DATA FROM COMMAND PROMPT

Searches

Show all data

Or single index

Can search on either Kevin or Kevin Smith or Kevin Betteridge(!) but we want to Kevin Smith to work but not David Betteridge. So the problem is that its analysing the text into separate words. Notice the score

Show mapping then

But can use speakers.keyword

# Elasticsearch.Net + click

This is a low level client, Abstracts the rest calls away and also supports things such as cluster failover

DEMO

Everything is loosely typed

Fiddler

# NEST

Higher level client built on top of ElasticSearch.net

This is strongly typed – show post class (case sensitive)

Change size

Fiddler

# Summary

So to summarise

The NEST Client is most developer friendly and is built using the ElastisSearch.Net client

The ElastisSearch.Net client can do everything that the REST client can do, built from it’s specification

# The Application Revisited

Show and scope to StackOverflowClientES

Now we know how to query data, we can update our application

Like so!

# Log Stash

**Centralize, Transform & Stash Your Data**

Logstash is an open source, server-side data processing pipeline that ingests data from a multitude of sources simultaneously, transforms it, and then sends it to your favorite “stas

Requires the Java runtime – might be an issue on dotnet only servers.

cd F:\logstash-6.2.2\bin

.\logstash -f ..\demo\_conf

Load data

Show config

Show config in iis manager

Show on map

# Beats

The libbeat library is written in GoLang

Demo WinLogBeat

Cd into C:\Code\Elastic Search Talk\Software\winlogbeat-6.2.4-windows-x86\_64

Show winlogbeat.yml (event logs + setup dashboard)

./winlogbeat.exe -c winlogbeat.yml

Show results in Kibana (watch out data range)

There is an IIS beat coming soon

# Kibana

What about if we want to better understand our data. This is where Kibana comes in.

<http://localhost:5601>

Create index over stackoverflow data

Show discover – watch time frame

Show total number of posts – save

Show posts by person – but filter for Questions only.

Tags – scale by log – save

Build Dashboard

# Grafana

An alternative to Kibana is Grafana.

This allows data to be read from alternative source, and can query multiple elastic search clusters.

However as it’s more generic, I don’t find it as good for exploring data.

Grafana has decent role based security built in, Kibana needs the x-pack, which until recently you had to pay for.

# Serilog

Show Program

Index.cs

Appsettings.json

Run website and refresh a few times

Kibana – create index (dotnetsheff)

Show data – structures v unstructured.

# Security

Which brings us on to security.

In the Microsoft world I was used to everything being secured by default. This isn’t the case with ELK stack.

Do you need security – well that depends on what you are using elastic search for. Backing the stackoverflow search client, then probably not. This is no different from having an internal sqlserver database.

If you are using it as a logging system, then yes. You don’t want people to be able to add/amend/delete logs.

When using Kibana, do you want everyone to see you data? You can use the Xpack to restrict what dashboards a person can see, and if they are allowed to edit them.

You could put an API gateway in front of Elastic search. So your application calls the API gateway, this verifies you are allowed to make the call, and the passes the class onto ES. You would configure ES so that it can only be called from the API gateway.

# Scaling

Nodes, Shards and replicas

Replicas can help read performance