Configure the Health Dashboard for large numbers of Applications

Summary: This section describes specific **configuration options and best practices** for one single instance of the **Health Dashboard** when you want to upload a **large number of Applications (200+)**.

Introduction

The Health Dashboard (HD) is currently supported and tested for a use case of 200 applications maximum (see **Standalone Health Dashboard deployment**). For customers that have more than 200 applications to consolidate into the dashboard, this is page describes various good practices and configuration options that could help to reduce the risk of performance issues.

The main areas that can impact performance are as follows:

- Infrastructure (the hardware used to run the dashboard)
- Number of users that are going to connect to the dashboard and the scope of applications they are authorised to access
- Number of Applications that have been consolidated
- Number of snapshots
- Number of tiles configured for display in the homepage

Deployment considerations

With regard to the deployment data sizing, please refer to **Deployment - sizing** to evaluate the appropriate configuration based on the number of Applications you will need to consolidated.

CAST Storage Service / PostgreSQL installation

Install on Linux

To improve the performance of CAST Storage Service/PostgreSQL, consider installing it on a **Linux operating system** instead of on Windows. See **PostgreSQL for Linux or Docker** for more information.

Isolate Measurement schema

If the Health Dashboard is going to be accessed by a **high number of users**, good practice is to **isolate the Measurement schema** on a dedicated CAST Storage Service/PostgreSQL instance instead of installing it on the same instance used to host the Dashboard schema. This configuration provides improved performance with regard to the number of connections to be configured between the Health Dashboard web application on Apache Tomcat and the CAST Storage Service/PostgreSQL instance (see section below for more information about Tomcat configuration).

Run optimization procedures on CAST Storage Service/PostgreSQL

It is good practice to run optimization procedures on the CAST Storage Service/PostgreSQL instance that is hosting your Measurement schema on a regular basis or at least each time a Measurement schemas is deleted and restored in order to recover disk space used by deleted data. Please run the following commands:

```
VACUUM FULL DSS_METRIC_RESULTS;
VACUUM FULL DSS_OBJECTS;
REINDEX TABLE DSS_METRIC_RESULTS;
REINDEX TABLE DSS_OBJECTS;
ANALYZE DSS_METRIC_RESULTS;
ANALYZE DSS_OBJECTS;
```

- You can use the built in **CSSOptimize** utility to run analyze/vacuum operations against a specific database.
- Optimization is run automatically immediately on completion of a **data upload to the Measure schema**.

Apache Tomcat configuration

Deployment on Linux

To improve performance, consider deploying Apache Tomcat on a **Linux operating system** instead of on Windows. Currently CAST does not provide any documentation around this, but Linux has been used successfully to host Apache Tomcat for use with the Health Dashboard.

Optimize context.xml for 1.x WAR files

The **context.xml** file has been removed in 2.x WAR files.

When deploying the Health Dashboard consider modifying the default configuration in the **CATALINA_HOME\webapps\CAST-Health\META-INF\context.xml** file to manage the number of connections between the web application and the CAST Storage Service instance. Below is the default configuration provided "out-of-the-box" in the **context.xml** file for both **Tomcat 7** and **8/8.5/9**:

```
<!-- TOMCAT 7 -->
<Resource name="jdbc/domains/AAD" url="jdbc:postgresql://localhost:2280/postgres"
initConnectionSqls="SET search_path TO [Measure Schema];"
username="operator" password="CastAIP"
auth="Container" type="javax.sql.DataSource" driverClassName="org.postgresql.Driver"
validationQuery="select 1"
initialSize="5" maxActive="20" maxIdle="10" maxWait="-1"/>

<!-- TOMCAT 8 / 8.5 -->
<Resource name="jdbc/domains/${domainName}"
url="jdbc:postgresql://${host}:${port}/postgres"
connectionInitSqls="SET search_path TO ${schema};"
username="${user}" password="${password}"
auth="Container" type="javax.sql.DataSource" driverClassName="org.postgresql.Driver"
validationQuery="select 1" initialSize="5" maxTotal="20" maxIdle="10" maxWaitMillis="-1"/>
```

Please optimize the following parameters depending on the number of users and user connection profile:

Tomcat 7	**Tomcat 8/8.5/9**	**Description**
initialSize	initialSize	(int) The initial number of connections that are created when the pool is started. Default value is 5.
maxActive	maxTotal	(int) The maximum number of active connections that can be allocated from this pool at the same time. The default value is 20.
maxIdle	maxIdle	(int) The maximum number of connections that should be kept in the pool at all times. Default value is 10. Idle connections are checked periodically (if enabled).
maxWait	maxWaitMillis	(int) The maximum number of milliseconds that the pool will wait (when there are no available connections) for a connection to be returned before throwing an exception. Default value is -1 (i.e. the pool will wait indefinitely).

The configuration of these parameters is easier to understand if you think of them as exactly corresponding to the **number of cash registers you need to open to avoid having too many customers waiting in your shop to buy articles**. Depending on the number of customers and period in the day, you will need to adapt the number of opened cash registers. In addition to that, these cash registers can be used for other shops in parallel. It is therefore very important to know the users' profile and when they are going to connect to your dashboard to be able to adapt these parameters: with a lot of potential users that will connect to a unique Measurement Service schema, this default configuration should be good.

Optimize application.properties for 2.x WAR and ZIP files

When deploying the Health Dashboard consider modifying the default configuration in the following file to manage the number of connections between the web application and the CAST Storage Service/PostgreSQL instance:

```
WAR  2.x
CATALINA_HOME\webapps\<deployed_war>\WEB-INF\classes\application.properties

ZIP  2.x
<unpacked_zip>\configurations\application.properties
```

Below is the default configuration provided "out-of-the-box" in the **application.properties** file:

```
## DATASOURCE
# Resource1 is the datasource name used in domains.properties
# Adapt server name (localhost) and port (2282) if required
# You can add multiple datasources if you want to connect to multiple CSS Servers. Datasource name must be
unique
# You have to configure your domains names and relative schema names in domains.properties
restapi.datasource[0].url=jdbc:postgresql://localhost:2282/postgres
restapi.datasource[0].username=operator
restapi.datasource[0].password=CastAIP
restapi.datasource[0].poolname=Resource1
restapi.datasource[0].minimumIdle=10
restapi.datasource[0].maximumPoolSize=20
```

Optimize the following parameters depending on the number of users and user connection profile:

minimumIdle	The minimum number of connections that should be kept in the pool at all times (even if there is no traffic). Default value is 10. Idle connections are checked periodically.
maximumPoolSize	The maximum number of active connections that can be allocated from this pool at the same time. The default value is 20.

Using ping to check for performance issues

If you are facing performance issues, you can run a "ping" test to check the connection between the server hosting Apache Tomcat and the server hosting CSS/Postgres. This may help you determine whether your performance issues are due to a bottleneck on your internal network. Please proceed as follows:

- Open a command line window (cmd) from the server hosting Apache Tomcat
- Run the following command: ping <name_of_CSS_or_postgres_server>

If the response time returned during the ping test is high, the bottleneck/issue is likely to be located on the internal network. Please contact your network administrator in this case.

Applications and Snapshots

The **number of Applications** consolidated into **one single Measurement Service schema** can impact the performance of your dashboard. CAST generally considers **more than 200 Applications** to be a "large" deployment where performance issues may start to appear, but this figure is not set in stone and depending on other criteria (such as the number of snapshots per Application) you may see a performance hit with a lower number of Applications (for example 20 Applications with a total of 2000 snapshots).

The **number of snapshots** per Application is also key as mentioned already. You may only have 20 Applications, but if the total number of snapshots in these 20 Applications is 2000 (for example) then you may see a performance hit.

Users

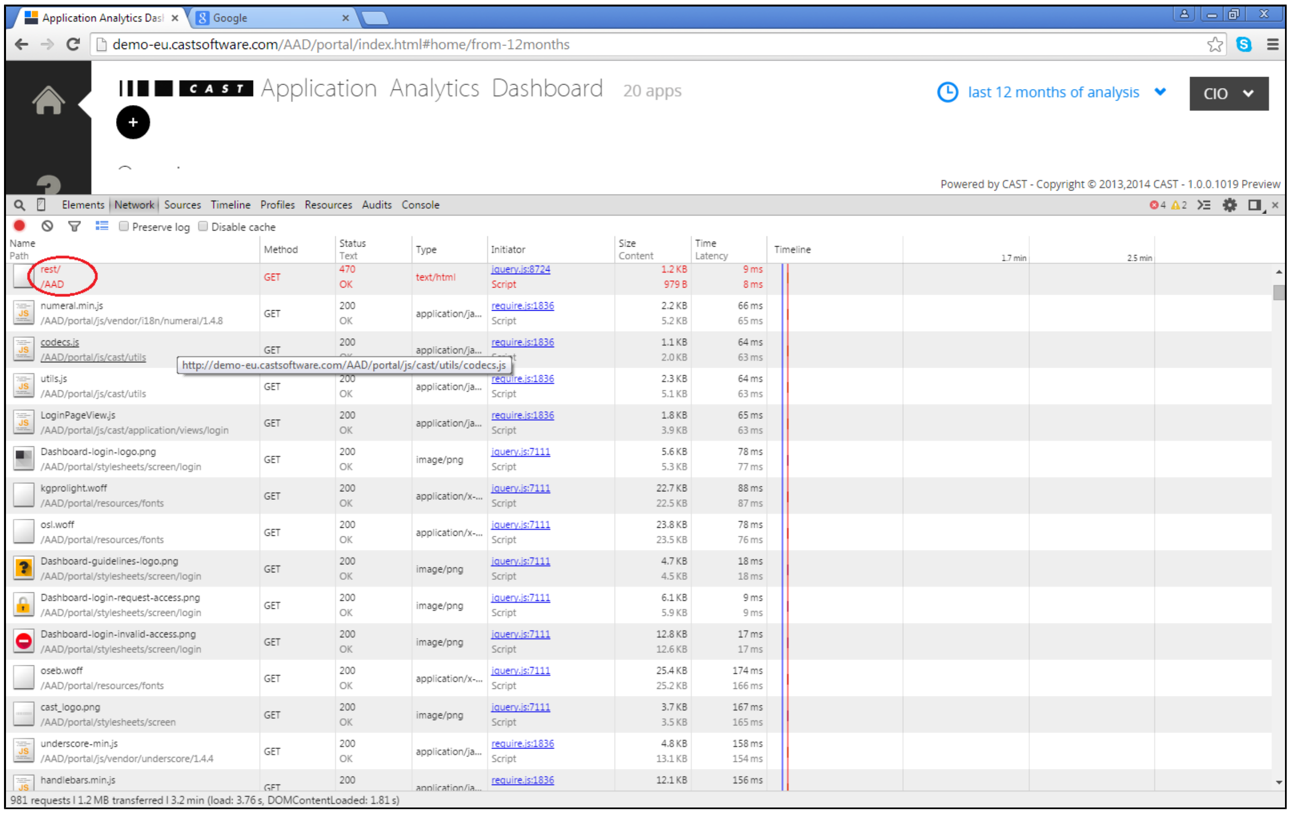
The number of users that are connecting to your dashboard can impact performance:

- If you are using Active Directory/LDAP authentication, performance of the dashboard can be adversely affected by the response time of your Active Directory/LDAP server.
- If all users are authorized to view all Applications, performance can also be impacted

Checking Active Directory/LDAP authentication response time

If you are using Active Directory/LDAP authentication, performance of the dashboard can be adversely affected by the response time of your Active Directory/LDAP server. To check whether this is the case, please proceed as follows:

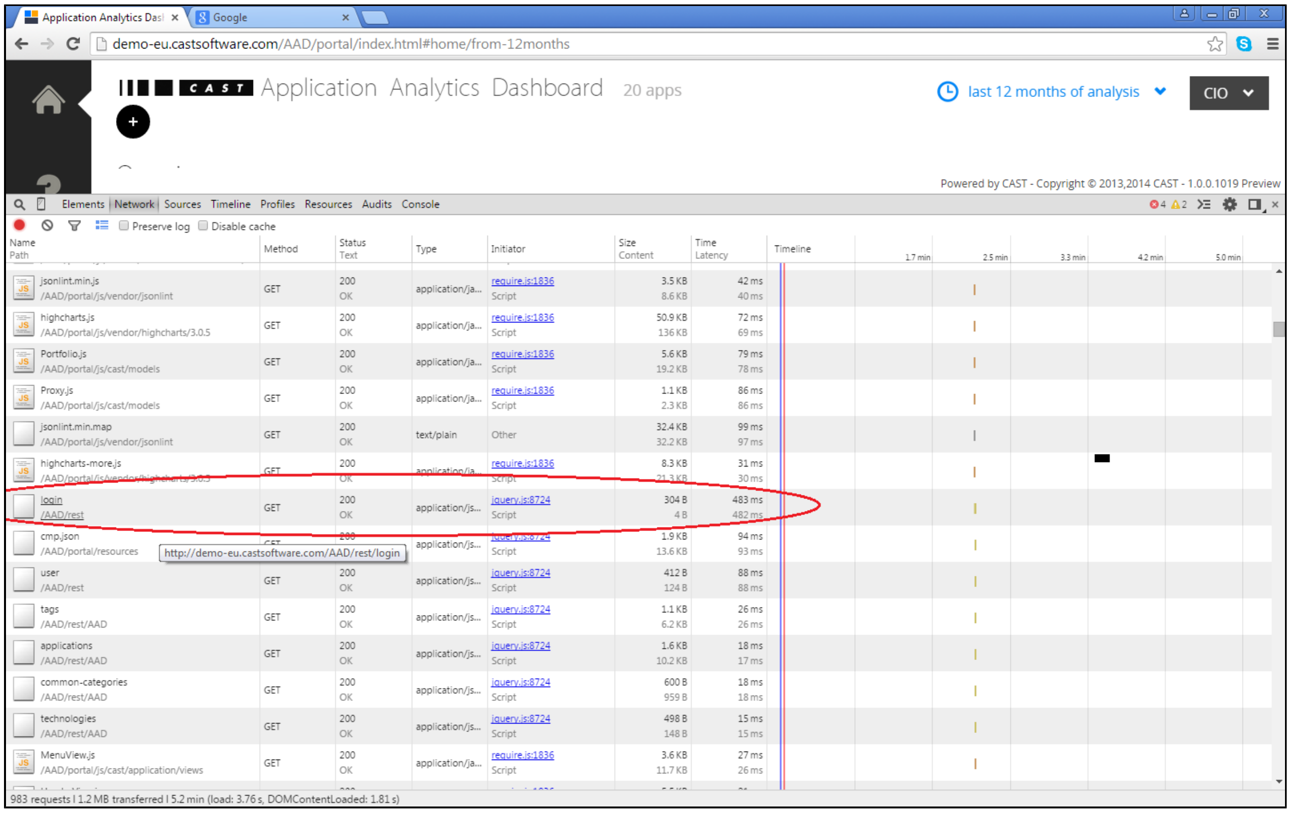
- If you already have a HAR (HTTP Archive) trace file, you can use the HAR files viewer with Google Chrome: http://ericduran.github.io/chromeHAR/
- If you do not have a HAR (HTTP Archive) trace file, you can view response times by:
  - connecting to the Health Dashboard using Google Chrome, tapping F12 and then reproducing the issue. You will get a HAR trace at the bottom of the page in the "Network" tab.
  - In the HAR trace look for the following as shown in the images below:
    - rest/
    - login
  - The value listed in "ms" in the "Time" column indicates how login the process has taken. If the value is high, please contact your network administrator in this case as this delay is being caused by something other than the CAST dashboard. For example: the total login time to the dashboard is less than 6 seconds but you observe that the "rest/ login" time is equal to 5 seconds. In this case, the issue is located on your Active Directory/LDAP server.


Application Analytics Das
Google
demo-eu.castsoftware.com/AAD/portal/index.html#home/from-12months
CAST
Application Analytics Dashboard
20 apps
last 12 months of analysis
CIO
Powered by CAST - Copyright © 2013,2014 CAST - 1.0.0.1019 Preview
Elements
Network
Sources
Timeline
Profiles
Resources
Audits
Console
Preserve log
Disable cache
Timeline
1.7 min
2.5 min

Name Path	Method	Status Text	Type	Initiator	Size Content	Time Latency
rest/ /AAD	GET	470 OK	text/html	jquery.js:8724 Script	1.2 KB 979 B	9 ms 8 ms
numeral.min.js /AAD/portal/js/vendor/i18n/numeral/1.4.8	GET	200 OK	application/ja...	require.js:1836 Script	2.2 KB 5.2 KB	66 ms 65 ms
codecs.js /AAD/portal/js/cast/utils	GET	200 OK	application/ja...	require.js:1836 Script	1.1 KB 2.0 KB	64 ms 63 ms
utils.js /AAD/portal/js/cast/utils	GET	200 OK	application/ja...	require.js:1836 Script	2.3 KB 5.1 KB	64 ms 63 ms
LoginPageView.js /AAD/portal/js/cast/application/views/login	GET	200 OK	application/ja...	require.js:1836 Script	1.8 KB 3.9 KB	65 ms 63 ms
Dashboard-login-logo.png /AAD/portal/stylesheets/screen/login	GET	200 OK	image/png	jquery.js:7111 Script	5.6 KB 5.3 KB	78 ms 77 ms
kgprolight.woff /AAD/portal/resources/fonts	GET	200 OK	application/x-...	jquery.js:7111 Script	22.7 KB 22.5 KB	88 ms 87 ms
osl.woff /AAD/portal/resources/fonts	GET	200 OK	application/x-...	jquery.js:7111 Script	23.8 KB 23.5 KB	78 ms 76 ms
Dashboard-guidelines-logo.png /AAD/portal/stylesheets/screen/login	GET	200 OK	image/png	jquery.js:7111 Script	4.7 KB 4.5 KB	18 ms 18 ms
Dashboard-login-request-access.png /AAD/portal/stylesheets/screen/login	GET	200 OK	image/png	jquery.js:7111 Script	6.1 KB 5.9 KB	9 ms 9 ms
Dashboard-login-invalid-access.png /AAD/portal/stylesheets/screen/login	GET	200 OK	image/png	jquery.js:7111 Script	12.8 KB 12.6 KB	17 ms 17 ms
oseb.woff /AAD/portal/resources/fonts	GET	200 OK	application/x-...	jquery.js:7111 Script	25.4 KB 25.2 KB	174 ms 166 ms
cast_logo.png /AAD/portal/stylesheets/screen	GET	200 OK	image/png	jquery.js:7111 Script	3.7 KB 3.5 KB	167 ms 165 ms
underscore-min.js /AAD/portal/js/vendor/underscore/1.4.4	GET	200 OK	application/ja...	require.js:1836 Script	4.8 KB 13.1 KB	158 ms 154 ms
handlebars.min.js	GET	200	application/ja...	require.js:1836	12.1 KB	156 ms

http://demo-eu.castsoftware.com/AAD/portal/js/cast/utils/codecs.js

981 requests | 1.2 MB transferred | 3.2 min (load: 3.76 s, DOMContentLoaded: 1.81 s)



demo-eu.castsoftware.com/AAD/portal/index.html#home/from-12months

CAST Application Analytics Dashboard 20 apps

last 12 months of analysis

CIO

Powered by CAST - Copyright © 2013,2014 CAST - 1.0.0.1019 Preview

Name / Path	Method	Status / Text	Type	Initiator	Size / Content	Time / Latency
jsonlint.min.js /AAD/portal/js/vendor/jsonlint	GET	200 OK	application/ja...	require.js:1836 Script	3.5 KB 8.6 KB	42 ms 40 ms
highcharts.js /AAD/portal/js/vendor/highcharts/3.0.5	GET	200 OK	application/ja...	require.js:1836 Script	50.9 KB 136 KB	72 ms 69 ms
Portfolio.js /AAD/portal/js/cast/models	GET	200 OK	application/ja...	require.js:1836 Script	5.6 KB 19.2 KB	79 ms 78 ms
Proxy.js /AAD/portal/js/cast/models	GET	200 OK	application/ja...	require.js:1836 Script	1.1 KB 2.3 KB	86 ms 86 ms
jsonlint.min.map /AAD/portal/js/vendor/jsonlint	GET	200 OK	text/plain	Other	32.4 KB 32.2 KB	99 ms 97 ms
highcharts-more.js /AAD/portal/js/vendor/highcharts/3.0.5	GET	200 OK	application/ja...	require.js:1836 Script	8.3 KB 21.3 KB	31 ms 30 ms
login /AAD/rest	GET	200 OK	application/js...	jquery.js:8724 Script	304 B 4 B	483 ms 482 ms
cmp.json /AAD/portal/resources	GET	200 OK	application/js...	jquery.js:8724 Script	1.9 KB 13.6 KB	94 ms 93 ms
user /AAD/rest	GET	200 OK	application/js...	jquery.js:8724 Script	412 B 124 B	88 ms 88 ms
tags /AAD/rest/AAD	GET	200 OK	application/js...	jquery.js:8724 Script	1.1 KB 6.2 KB	26 ms 26 ms
applications /AAD/rest/AAD	GET	200 OK	application/js...	jquery.js:8724 Script	1.6 KB 10.2 KB	18 ms 17 ms
common-categories /AAD/rest/AAD	GET	200 OK	application/js...	jquery.js:8724 Script	600 B 959 B	18 ms 18 ms
technologies /AAD/rest/AAD	GET	200 OK	application/js...	jquery.js:8724 Script	498 B 148 B	15 ms 15 ms
MenuView.js /AAD/portal/js/cast/application/views	GET	200 OK	application/ja...	require.js:1836 Script	3.6 KB 11.7 KB	27 ms 26 ms

http://demo-eu.castsoftware.com/AAD/rest/login

983 requests l 1.2 MB transferred l 5.2 min (load: 3.76 s, DOMContentLoaded: 1.81 s)

Authorization

When you are consolidating a large number of Applications, you should always actively use the data authorization feature to prevent users from viewing Applications/data they do not need to, therefore improving performance.

Benefits:

- When a user is logged in, if authorization mapping between user and Application has been implemented, the user will access only the Applications he/she is authorized to access. As such the number of Applications that are accessed will be highly reduced and performance will improve (less Applications to display, less calculations to provide in the homepage, less data to load in graphs)
- Using data authorization via tags is one way to avoid having to duplicated Health Dashboard instances (database + war).

Homepage configuration

Each tile that is configured for display in the dashboard home page will take time to render as calculations will have to be done based on the number of applications and snapshots (e.g. to display TQI score evolution tile, the average of all snapshots to all applications must first be calculated and then the data must be displayed graphically).

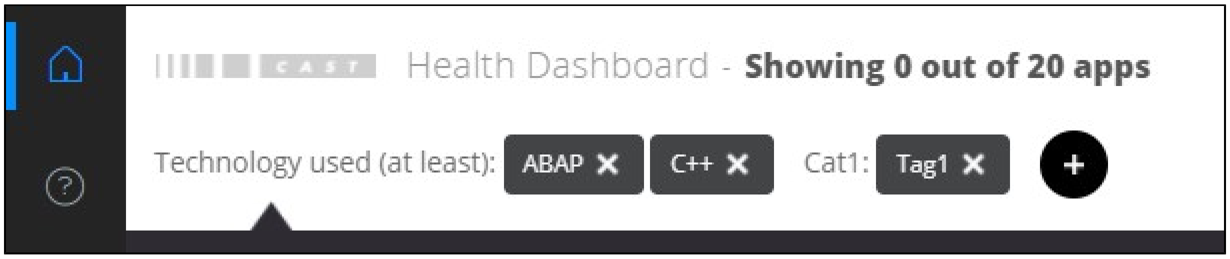
Reduce the number of tiles displayed in the homepage (cmp.json)

Some tiles can be time consuming to load. Please use the following as guide to limit the number tiles in your homepage:

- Use a maximum of one tile for benchmarking (eg bubble graph tile or treemap tile). Avoid using this type of tile if you can as it is time consuming to load.
- Don't use a tile that lists rules across all Applications
- Remove or reduce the number of evolution tiles

Provide a direct filtered URL to reduce number of Applications that load in the home page

You can provide a direct URL that points to a filter (i.e. a tag or a specific time filter). This will improve performance as not all the Applications in the portfolio need to be displayed:


Health Dashboard - Showing 0 out of 20 apps
Technology used (at least):
ABAP
C++
Cat1:
Tag1

To share a filtered URL simply browse to the filtered data you want to share and copy the URL from your browser's address bar, e.g.:

- http://<webapplication-server>/CAST-Health/portal/index.html#home/tags-1/from-12months
- http://<webapplication-server>/CAST-Health/portal/index.html#home/tags-46/from-30days

Performance test results

Some performance tests have been completed based on specific scenarios for the Health Dashboard. These tests provide some insight into the limit of responses time depending on the number of simultaneously connected users.

- See Last Performance analysis report with JMeter on the Health Dashboard for CAST AIP 7.3: AAD Performance Analysis (June 2014).pptx
- See Performance Analysis report with JMeter on the Health Dashboard for two different Measurement Service schemas: AAD Performance Analysis (May 2016).pptx