Chapter 1. OpenNMS Surveillance View

When networks are larger and contain devices of different priority, it becomes interesting to show at a glance how the “whole system” is working. The surveillance view aims to do that. By using categories, you can define a matrix which allows to aggregate monitoring results. Imagine you have 10 servers with 10 internet connections and some 5 PCs with DSL lines:

<table>
<thead>
<tr>
<th>Category</th>
<th>Servers</th>
<th>Internet Connections</th>
</tr>
</thead>
<tbody>
<tr>
<td>Super important</td>
<td>1 of 10</td>
<td>0 of 10</td>
</tr>
<tr>
<td>Slightly important</td>
<td>0 of 10</td>
<td>0 of 10</td>
</tr>
<tr>
<td>Vanity</td>
<td>4 of 10</td>
<td>0 of 10</td>
</tr>
</tbody>
</table>

The whole idea is to give somebody at a glance a hint on where the trouble is. The matrix-type of display allows a significantly higher aggregation than the simple list. In addition, the surveillance view shows nodes rather than services - an important tidbit of information when you look at categories. At a glance, you want to know how many of my servers have an issue rather than how many services in this category have an issue.

The visual indication for outages in the surveillance view cells is defined as the following:

- No services down: green as normal
- One (1) service down: yellow as warning
- More than one (1) services down: red as critical

This Surveillance View model also builds the foundation of the Dashboard View.

1.1. Default Surveillance View Configuration

Surveillance Views are defined in the surveillance-views.xml file. This file resides in the OpenNMS etc directory.

**NOTE**

This file can be modified in a text editor and is reread every time the Surveillance View page is loaded. Thus, changes to this file do not require OpenNMS to be restarted.

The default configuration looks like this:
1.2. Configuring Surveillance Views

The Surveillance View configuration can also be modified using the Surveillance View Configurations editor on the OpenNMS Admin page.

![Surveillance View Configurations UI](image)

Figure 2. The Surveillance View Configurations UI

This page gives an overview of the configured Surveillance Views and allows the user to edit, remove or even preview the defined Surveillance View. Furthermore, the default Surveillance View can be selected using the checkbox in the DEFAULT column.

When editing a Surveillance View the user has to define the view's title and the time in seconds between successive refreshes. On the left side of this dialog the defined rows, on the right side the defined columns are listed. Beside adding new entries an user can modify or delete existing entries. Furthermore, the position of an entry can be modified using the up/down buttons.

WARNING | Please note, that the old report-category attribute is deprecated and is no longer supported.
Editing row or column definitions require to choose an unique label for this entry and at least one OpenNMS category. When finished you can hit the Save button to persist your modified configuration or Cancel to close this dialog.

### 1.3. Categorizing Nodes

In order to categorize nodes in the Surveillance View, choose a node and click Edit beside Surveillance Category Memberships. Recalling from your Surveillance View, choose two categories that represent a column and a row, for example, Servers and Test, then click Add.

### 1.4. Creating Views for Users and Groups

You can use user and group names for Surveillance Views. When the Surveillance View page is invoked the following criteria selects the proper Surveillance View to be displayed. The first matching item wins:

1. Surveillance View name equal to the user name they used when logging into OpenNMS.
2. Surveillance View name equal to the user’s assigned OpenNMS group name
Chapter 2. OpenNMS Dashboard

In Network Operation Centers NOC an overview about issues in the network is important and often described as Dashboards. Large networks have people (Operator) with different responsibilities and the Dashboard should show only information for a given monitoring context. Network or Server operator have a need to customize or filter information on the Dashboard. A Dashboard as an At-a-glance overview is also often used to give an entry point for more detailed diagnosis through the information provided by the monitoring system. The Surveillance View allows to reduce the visible information by selecting rows, columns and cells to quickly limit the amount of information to navigate through.

2.1. Dashboard Components

The Dashboard is built with five components:

- **Surveillance View**: Allows to model a monitoring context for the Dashboard.
- **Alarms**: Shows unacknowledged Alarms which should be escalated by an Operator.
- **Notifications**: Shows outstanding and unacknowledged notifications sent to Engineers.
- **Node Status**: Shows all ongoing network Outages.
- **Resource Graph Viewer**: Shows performance time series reports for performance diagnosis.

The following screenshot shows a configured Dashboard and which information are displayed in the components.

![Dashboard with configured surveillance view and current outage](image)

The following section describe the information shown in each component. All other components display information based on the Surveillance View.

2.1.1. Surveillance View

The Surveillance View has multiple functions.

- Allows to model the monitoring context and shows service and node Outages in compact matrix view.
- Allows to limit the number of information in the Dashboard by selecting rows, columns and cells.

You can select columns, rows, single cells and of course all entries in a Surveillance View. Please refer to the Surveillance View Section for details on how to configure Surveillance Views.
2.1.2. Alarms

The Alarms component gives an overview about all unacknowledged Alarms with a severity higher than Normal(1). Acknowledged Alarms will be removed from the responsibility of the Operator. The following information are shown in:

![Figure 6. Information displayed in the Alarms component](image)

1. **Node**: Node label of the node the Alarm is associated.
2. **Log Msg**: The log message from the Event which is the source for this Alarm. It is specified in the event configuration file in `<logmsg />`
3. **Count**: Number of Alarms deduplicated by the reduction key of the Alarm.
4. **First Time**: Time for the first occurrence of the Alarm.
5. **Last Time**: Time for the last occurrence of the Alarm.

The Alarms component shows the most recent Alarms and allows the user to scroll through the last 100 Alarms.

2.1.3. Notifications

To inform people on a duty schedule notifications are used and force action to fix or reconfigure systems immediately. In OpenNMS it is possible to acknowledge notifications to see who is working on a specific issue. The Dashboard should show outstanding notifications in the NOC to provide an overview and give the possibility for intervention.

![Figure 7. Information displayed in the Notifications component](image)

1. **Node**: Label of the monitored node the notification is associated with.
2. **Service**: Name of the service the notification is associated with.
3. **Message**: Message of the notification.
4. **Responder**: User name who acknowledged the notification
5. **Response Time**: Time when the user acknowledged the notification

The Notifications component shows the most recent unacknowledged notifications and allows the user to scroll through the last 100 Notifications.
2.1.4. Node Status

An acknowledged Alarm doesn’t mean necessarily the outage is solved. To give an overview information about ongoing Outages in the network, the Dashboard shows an outage list in the Node Status component.

Figure 8. Information displayed in the Node Status component

1. Node: Label of the monitored node with ongoing outages.

2. Current Outages: Number of services on the node with outages and total number of monitored services, e.g. with the natural meaning of "3 of 3 services are affected".

3. 24 Hour Availability: Availability of all services provided by the node calculated by the last 24 hours.

2.1.5. Resource Graph Viewer

To give a quick entry point diagnose performance issues a Resource Graph Viewer allows to navigate to time series data reports which are filtered in the context of the Surveillance View.

Figure 9. Show time series based performance with the Resource Graph Viewer

It allows to navigate sequentially through resource graphs provided by nodes filtered by the Surveillance View context and selection and shows one graph report at a time.

2.2. Advanced configuration

The Surveillance View component allows to model multiple views for different monitoring contexts. It gives the possibility to create special view as example for network operators or server operators. The Dashboard shows only one configured Surveillance View. To give different users the possibility using their Surveillance View fitting their requirements it is possible to map a logged in user to a given Surveillance View used in the Dashboard.

The selected nodes from the Surveillance View are also aware of User Restriction Filter. If you have a group of users, which should see just a subset of nodes the Surveillance View will filter nodes which are not related to the assigned user group.

The Dashboard is designed to focus, and therefore also restrict, a user’s view to devices of their interest. To do this, a new role was added that can be assigned to a user that restricts them to viewing only the Dashboard if that is intended.

2.2.1. Using the Dashboard role

The following example illustrates how this Dashboard role can be used. For instance the user drv4doe is assigned the dashboard role. So, when logging in as drv4doe, the user is taking directly to the Dashboard page and is presented with a custom Dashboard based on the drv4doe Surveillance View definition.

Step 1: Create an user

The following example assigns a Dashboard to the user "drv4doe" (a router and switch jockey) and restricts the user for navigation to any other link in the OpenNMS WebUI.
Step 2: Change magic-users.properties

Now, edit the magic-users.properties file in the `/opt/opennms/etc` directory and set `drv4doe` as a dashboard user.

```plaintext
role.dashboard.name=OpenNMS Dashboard User
role.dashboard.users=drv4doe
role.dashboard.notInDefaultGroup=true
```

Step 3: Define Surveillance View

Edit the `$OPENNMS_HOME/etc/surveilliance-view.xml` file to add a definition for the user `drv4doe`, which you created in step 1.
This configuration and proper assignment of node categories will produce a default Dashboard for all users, other than drv4doe.

**TIP**
You can hide the upper navigation on any page by specifying `?quiet=true`; adding it to the end of the OpenNMS URL. This is very handy when using the dashboard on a large monitor or TV screen for office wide viewing.

However, when logging in as drv4doe, the user is taking directly to the Dashboard page and is presented with a Dashboard
based on the custom Surveillance View definition.

**NOTE**
The drv4doe user is not allowed to navigate to URLs other than the dashboard.jsp URL. Doing so will result in an Access Denied error.

### 2.2.2. Anonymous dashboards

You can modify the configuration files for the security framework to give you access to one or more dashboards without logging in. At the end you'll be able to point a browser at a special URL like http://opennms/dashboard1 or http://opennms/dashboard2 and see a dashboard without any authentication. First, configure surveillance views and create dashboard users as above. For example, make two dashboards and two users called dashboard1 and dashboard2. Test that you can log in as each of the new users and see the correct dashboard. Now create some aliases you can use to distinguish between dashboards. In /opt/opennms/jetty-webapps/opennms/WEB-INF, edit web.xml. Just before the first `<servlet-mapping>` tag, add the following servlet entries:

```xml
<servlet>
    <servlet-name>dashboard1</servlet-name>
    <jsp-file>/dashboard.jsp</jsp-file>
</servlet>

<servlet>
    <servlet-name>dashboard2</servlet-name>
    <jsp-file>/dashboard.jsp</jsp-file>
</servlet>
```

Just before the first `<error-page>` tag, add the following servlet-mapping entries:

```xml
<servlet-mapping>
    <servlet-name>dashboard1</servlet-name>
    <url-pattern>/dashboard1</url-pattern>
</servlet-mapping>

<servlet-mapping>
    <servlet-name>dashboard2</servlet-name>
    <url-pattern>/dashboard2</url-pattern>
</servlet-mapping>
```

After the last `<filter-mapping>` tag, add the following filter-mapping entries:

```xml
<filter-mapping>
    <filter-name>AddRefreshHeader-120</filter-name>
    <url-pattern>/dashboard.jsp</url-pattern>
</filter-mapping>

<filter-mapping>
    <filter-name>AddRefreshHeader-120</filter-name>
    <url-pattern>/dashboard1</url-pattern>
</filter-mapping>

<filter-mapping>
    <filter-name>AddRefreshHeader-120</filter-name>
    <url-pattern>/dashboard2</url-pattern>
</filter-mapping>
```

Next edit applicationContext-acegi-security.xml to enable anonymous authentication for the `/dashboard1` and `/dashboard2` aliases. Near the top of the file, find `<bean id="filterChainProxy" />`. Below the entry for `/rss.jsp*`, add an entry for each
of the dashboard aliases:

```xml
<bean id="filterChainProxy" class="org.acegisecurity.util.FilterChainProxy">
    <property name="filterInvocationDefinitionSource">
        <value>
            CONVERT_URL_TO_LOWERCASE_BEFORE_COMPARISON
            PATTERN_TYPE_APACHE_ANT
            /rss.jsp=httpSessionContextIntegrationFilter,logoutFilter,authenticationProcessingFilter,basicProcessingFilter,securityContextHolderAwareRequestFilter,anonymousProcessingFilter,basicExceptionTranslationFilter,filterInvocationInterceptor
            /dashboard1=httpSessionContextIntegrationFilter,logoutFilter,securityContextHolderAwareRequestFilter,dash1AnonymousProcessingFilter,filterInvocationInterceptor
            /dashboard2=httpSessionContextIntegrationFilter,logoutFilter,securityContextHolderAwareRequestFilter,dash2AnonymousProcessingFilter,filterInvocationInterceptor
            /**=httpSessionContextIntegrationFilter,logoutFilter,authenticationProcessingFilter,basicProcessingFilter,securityContextHolderAwareRequestFilter,anonymousProcessingFilter,exceptionTranslationFilter,filterInvocationInterceptor
        </value>
    </property>
</bean>
```

About halfway through the file, look for `<bean id="filterInvocationInterceptor"` >. Below the entry for `/dashboard.jsp`, add an entry for each of the aliases:

```xml
<bean id="filterInvocationInterceptor" class="org.acegisecurity.intercept.web.FilterSecurityInterceptor">
    ... /frontpage.htm=ROLE_USER,ROLE_DASHBOARD
    /dashboard.jsp=ROLE_USER,ROLE_DASHBOARD
    /dashboard1=ROLE_USER,ROLE_DASHBOARD
    /dashboard2=ROLE_USER,ROLE_DASHBOARD
    /gwt.js=ROLE_USER,ROLE_DASHBOARD
    ...
</bean>
```

Finally, near the bottom of the page, add a new instance of `AnonymousProcessingFilter` for each alias.

```xml
<!-- Set the anonymous username to dashboard1 so the dashboard page can match it to a surveillance view of the same name. -->
<bean id="dash1AnonymousProcessingFilter" class="org.acegisecurity.providers.anonymous.AnonymousProcessingFilter">
    <property name="key"><value>foobar</value></property>
    <property name="userAttribute"><value>dashboard1,ROLE_DASHBOARD</value></property>
</bean>

<bean id="dash2AnonymousProcessingFilter" class="org.acegisecurity.providers.anonymous.AnonymousProcessingFilter">
    <property name="key"><value>foobar</value></property>
    <property name="userAttribute"><value>dashboard2,ROLE_DASHBOARD</value></property>
</bean>
```

Restart OpenNMS and you should bring up a dashboard at http://…/opennms/dashboard1 without logging in.

**WARNING** There's no way to switch dashboards without closing the browser (or deleting the JSESSIONID session cookie).
If you accidentally click a link that requires full user privileges (e.g. Node List), you’ll be given a login form. Once you get to the login form, there’s no going back to the dashboard without restarting the browser. If this problem bothers you, you can set `ROLE_USER` in addition to `ROLE_DASHBOARD` in your `userAttribute` property. However this will give full user access to anonymous browsers.