knowledge14®

April 27–May 1, 2014 • Moscone Center • San Francisco, CA

# Lab Guide

## Debugging 301: Problem Diagnosis and Prevention Techniques

Chris Maloy & Oliver Schmitt

Lab instance: https://**<<name>>**.lab.service-now.com

Login / Passwords:

admin / Knowledge14

itil / Knowledge14

employee / Knowledge14

# Lab Goal

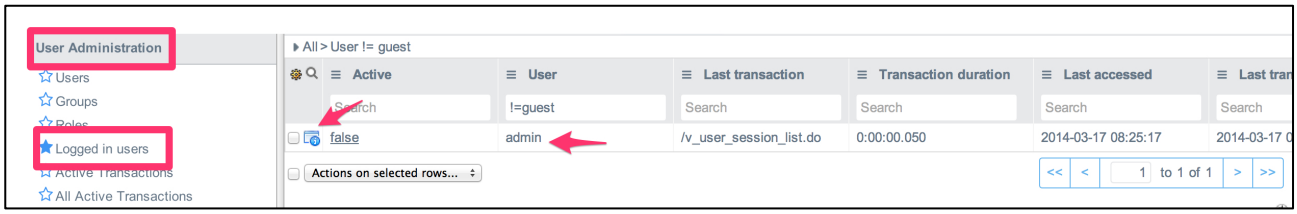
To use the Log File Download and Log File Browser to detect slow and recursive business rules effecting performance issues.

**Lab 1**
**Using App Server Logs to Find Issues**

## Using the Log File Utilities and App Log

1. Login to the instance URL on the cover page using the provided **admin** credentials.

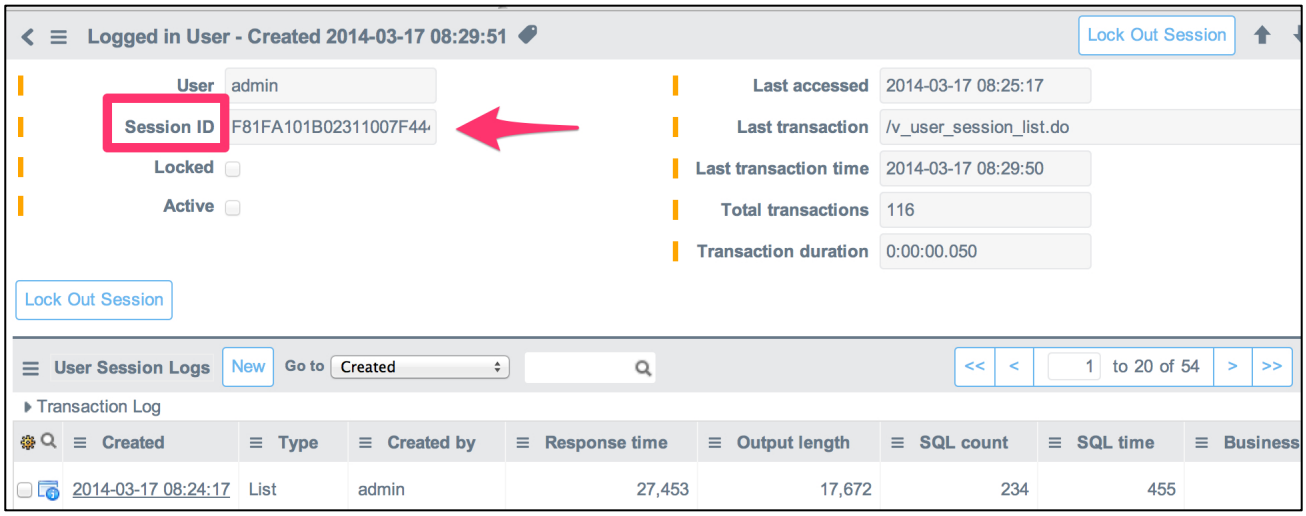
2. Navigate to **User Administration > Logged in users**.




3. Click the user record for the user you are logged in as.

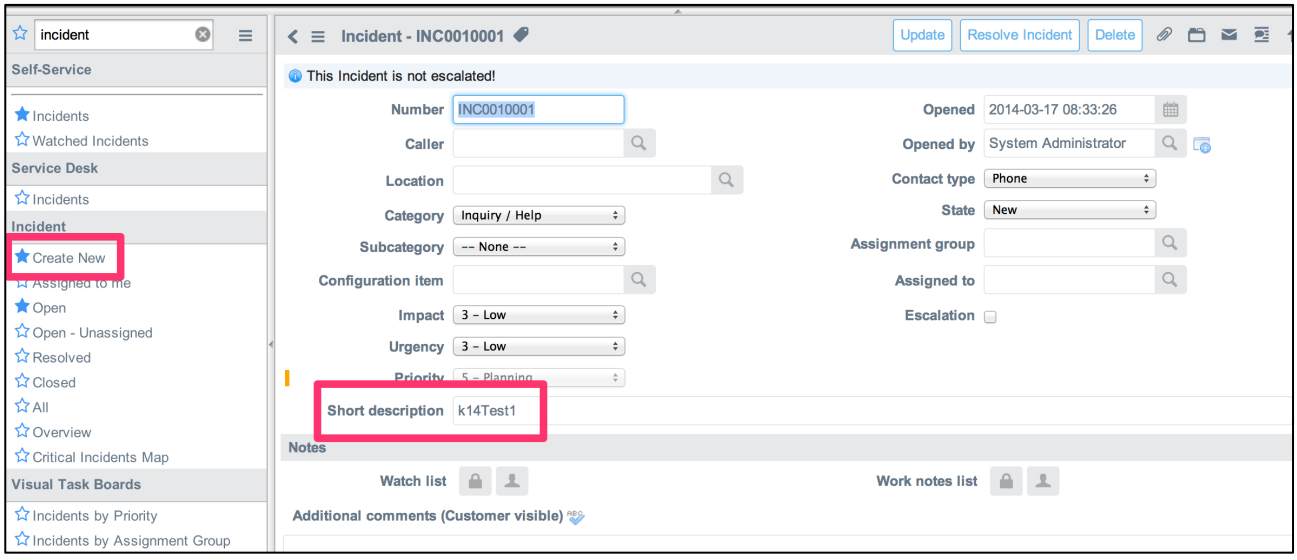
©2014 ServiceNow, Inc. All rights reserved.

4. Once you are in the logged in user record copy the Session ID. Keep it in your copy paste buffer until needed later on in this lab.




5. Navigate to **Incident > Create New**.

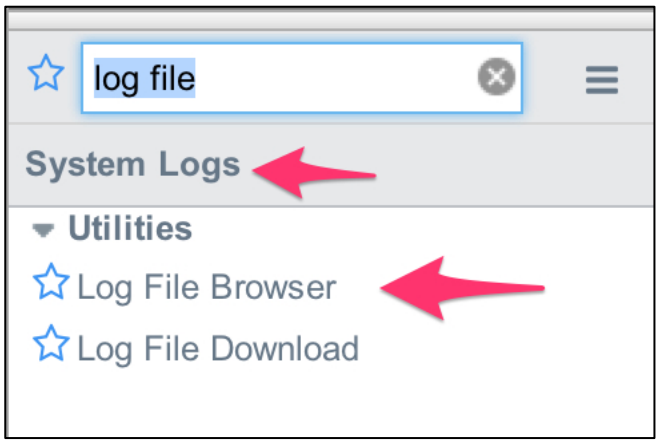
6. Fill out the **Short Description** field on the form. We are not doing anything fancy here. Click **Submit**.



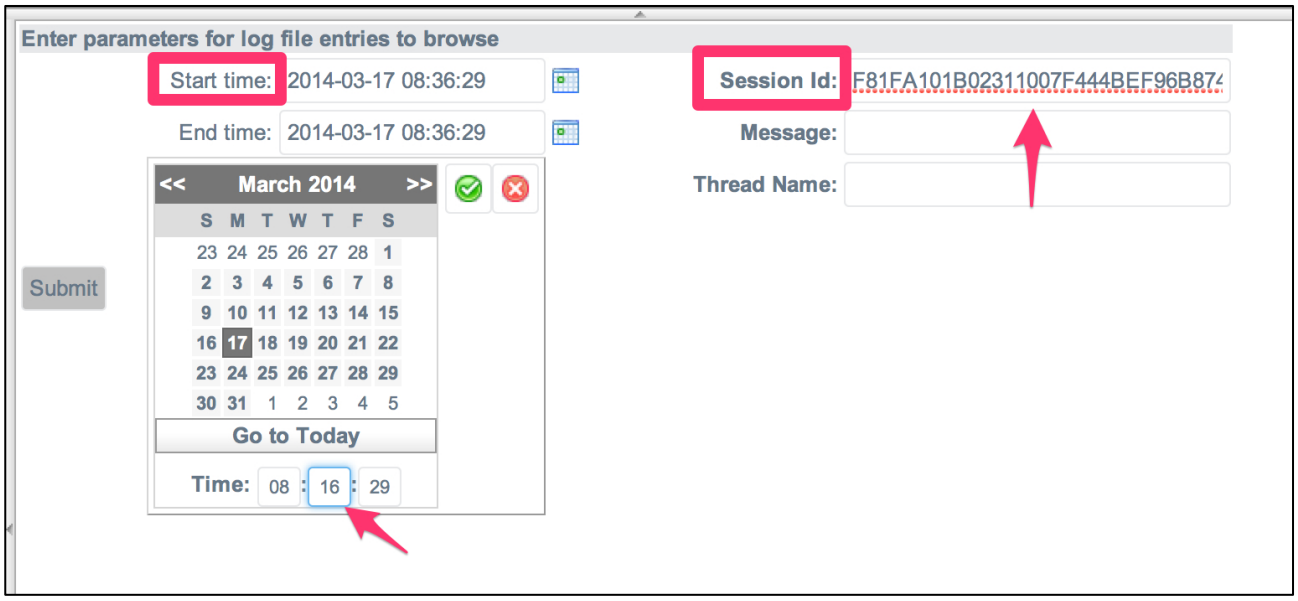

©2014 ServiceNow, Inc. All rights reserved.

7. Allow for some time to pass before the incident is created (5-10 seconds). I have intentionally added some business rules to slow this down.

8. Navigate to **System Logs > Log File Browser**.




9. Copy the Session ID in your buffer from step 5 above and paste it into **the Session Id** field. You will also want to change the Start time field. Change the minutes to reflect a time before creating the incident in step 6. Leave the rest default.



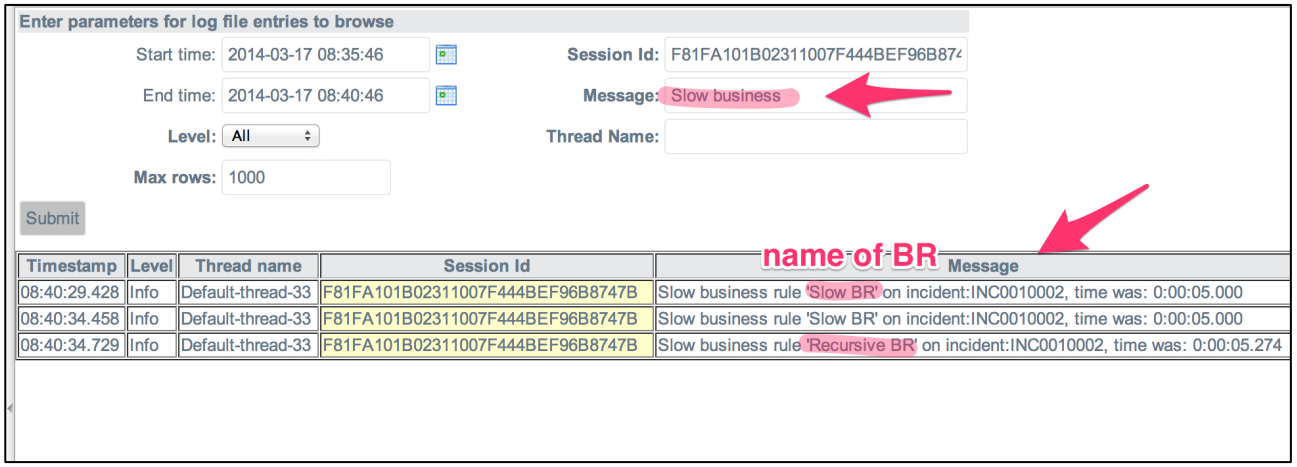

©2014 ServiceNow, Inc. All rights reserved.

10. You will be given a log file dump for a given user session. Either search with your Browser search feature or filter further using the **Message** field for the key words we talked about previously in the presentation, **Slow business rule** or **Recursive business rule**.

11. Go and find those business rules from your incident form and disable them.

## Lab Success Verification

1. Your screen should look like this:




## Challenge

1. Attempt to download the log file locally and either write a script or grep for the key words mentioned during the presentation. This could be something you do as a pre-production check on new code you introduce or if you seeing performance issues.

©2014 ServiceNow, Inc. All rights reserved.

# Lab Goal

To use the **IOStats** module to identify a script is making too many SQL inserts.

**Lab 2**
**Using**
**IOStats**

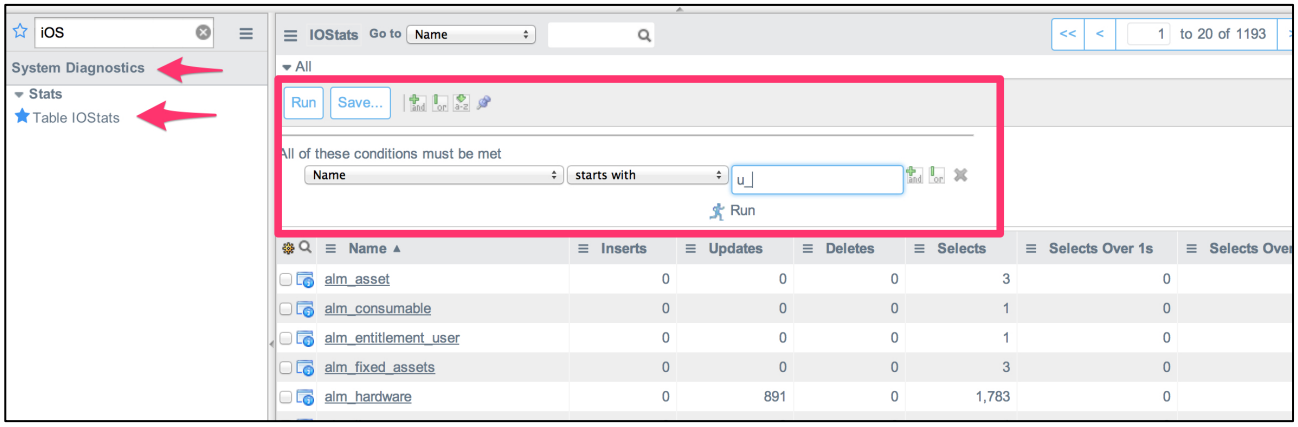
## Using the IOStats module

1. Instructor provides you with your own instance URL.

2. Log on with **admin/admin** credentials.

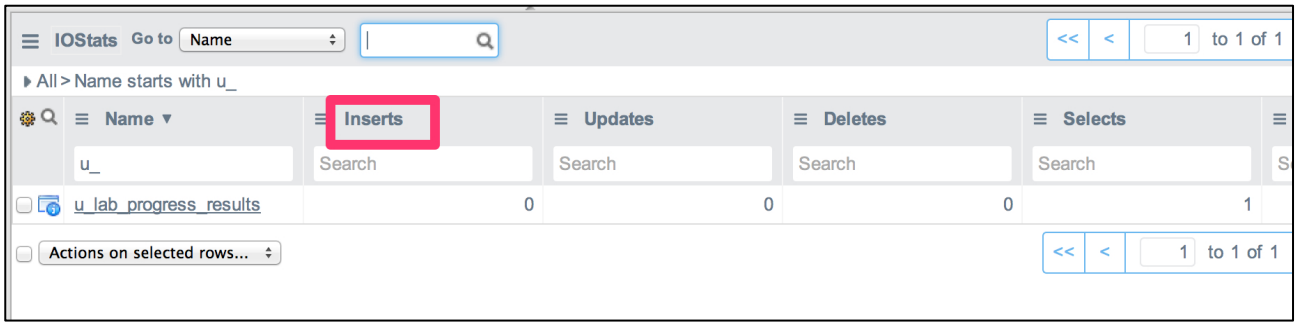
3. Navigate to **System Diagnostics > Table IOStats**.

4. Change the filter to look for custom tables by searching for table names that start with **u_** (user defined table).



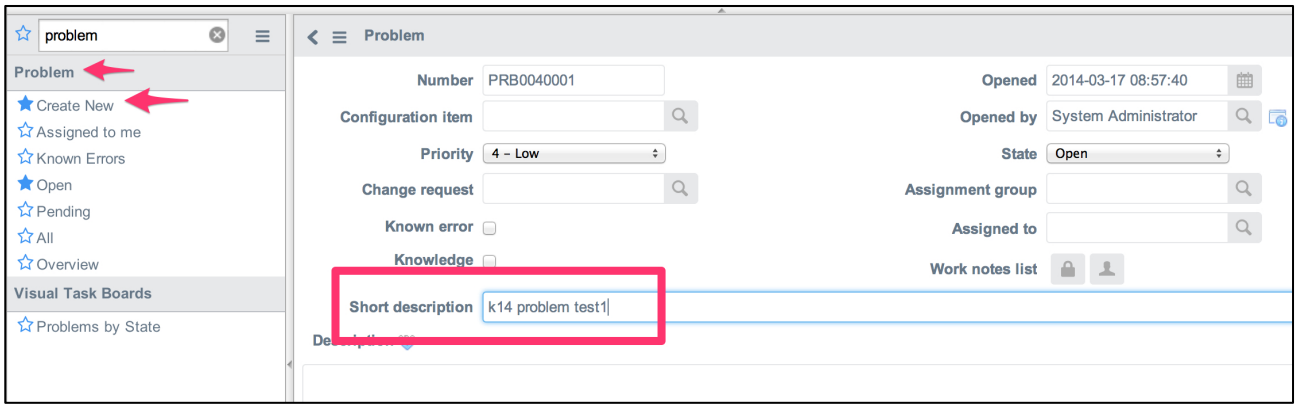

©2014 ServiceNow, Inc. All rights reserved.

5. Take note of the number of inserts that have occurred on these custom tables. Your values may differ than what I am displaying here, but we are interested in the change for this exercise.




6. Navigate to **Problem > Create New**.

7. Fill out the **Short Description** field on the form. We are not doing anything fancy here. Click **Submit**.




8. Repeat **Steps 6 – 8** two times changing the **Short Description** each time.

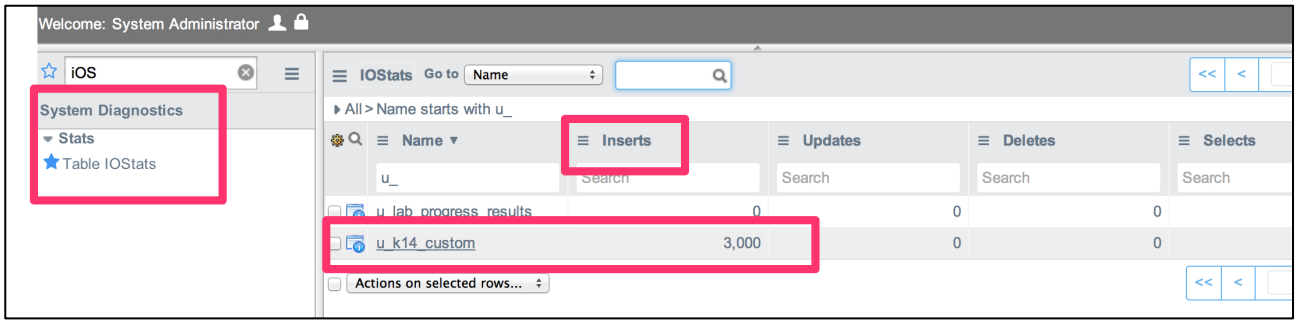
9. Allow for some time to pass before the incident is created (5-10 seconds). I have intentionally added some business rules to slow this down.

10. Navigate **to System Diagnostics > Table IOStats**.

©2014 ServiceNow, Inc. All rights reserved.

11. Change the filter to look for custom tables by searching for table names that start with **u_** (user defined table).

12. You can see that the number of inserts went way up when a problem was created.

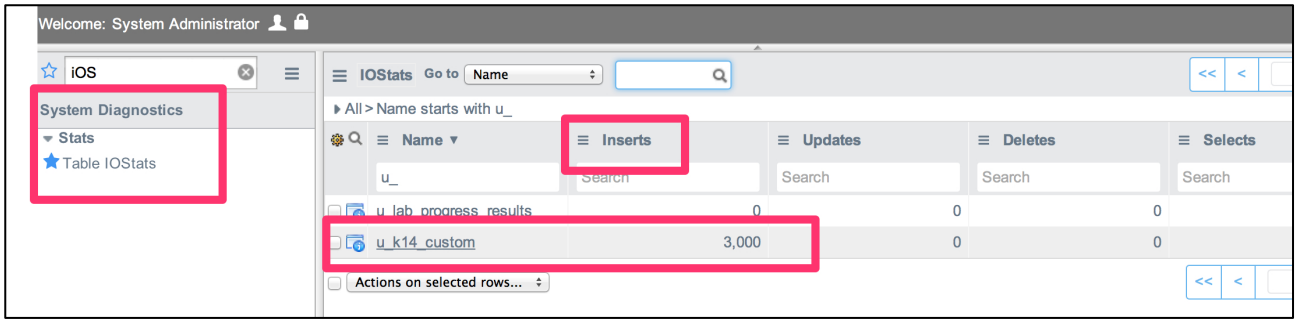



13. This is a simple lab, but the idea here is to keep an eye on the count of DB transactions made (either over time or after an action). It is also helpful to monitor the selects over 1 second and 10 second. You can identify slow running DB queries that way and use tricks like **setLimit()** when getting data.

14. Go and find those business rules from your problem table and disable it.

## Lab Success Verification

1. Your screen should look like this:




## Challenge

1. Navigate to the Slow Queries stats in the System Diagnostics Application. You are on a pretty clean instance, but familiarize yourself with the columns you can sort on. If you have a development or test instance you can use instead try using this tool to identify slow queries.

©2014 ServiceNow, Inc. All rights reserved.

# Lab Goal

To use the Event Log to find performance issues. Events are very powerful, but they are sneaky places that can cause performance issues.

**Lab 3**
**Event Log**

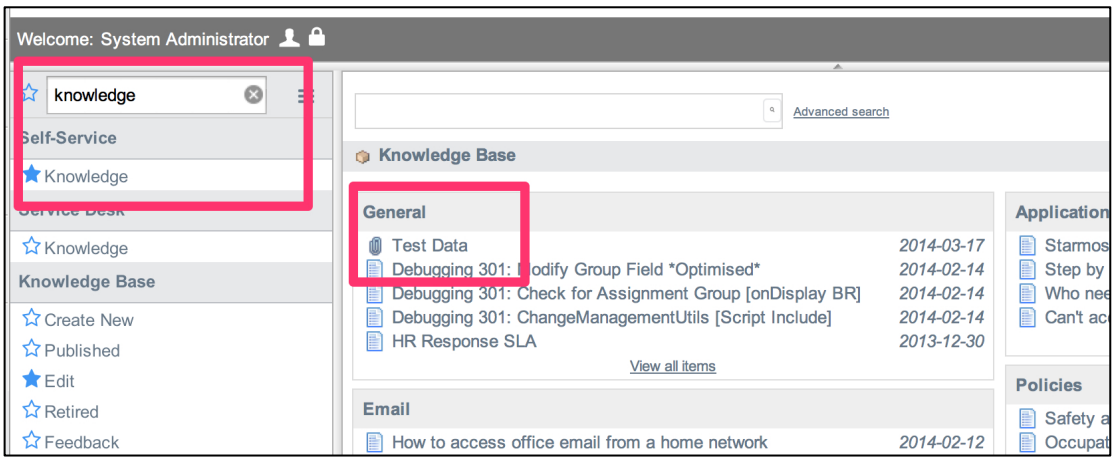
## Using the Event Log module

1. Instructor will provide you with your own instance URL.

2. Log on with admin/admin credentials.

3. Navigate to **Self-Service > Knowledge**.

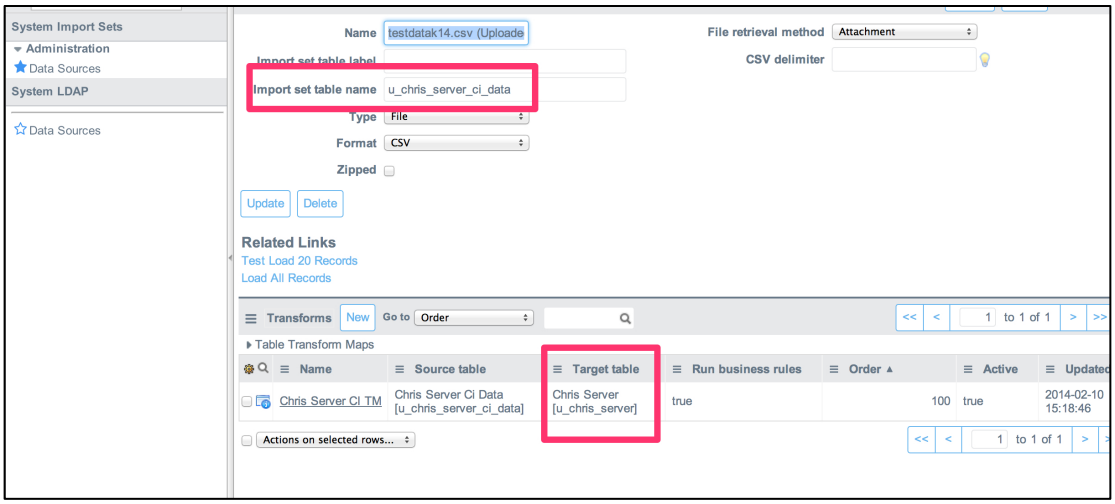
4. Under the **General** section click on the **TestDataK14.csv**. This will download the file to your local machine. We will need to attach this to the data source in the next steps.



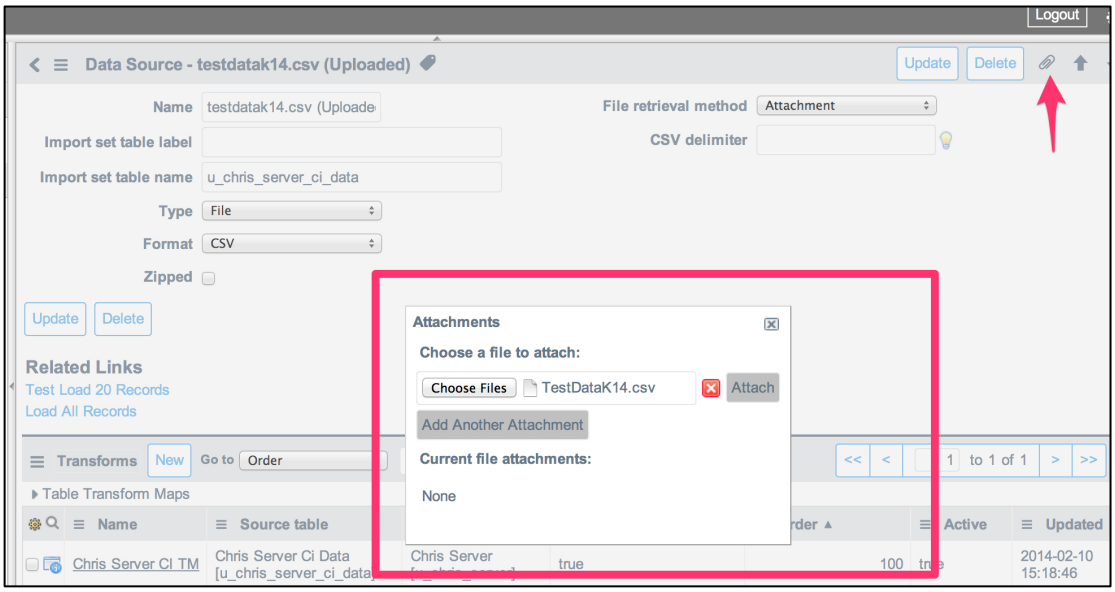

5. Navigate to **System Import Sets > Data Sources**.

©2014 ServiceNow, Inc. All rights reserved.

6. Click on **the testdatak14.csv** data source. The import set table name should be **u_chris_server_ci_data** and the target table should be **u_chris_server**.

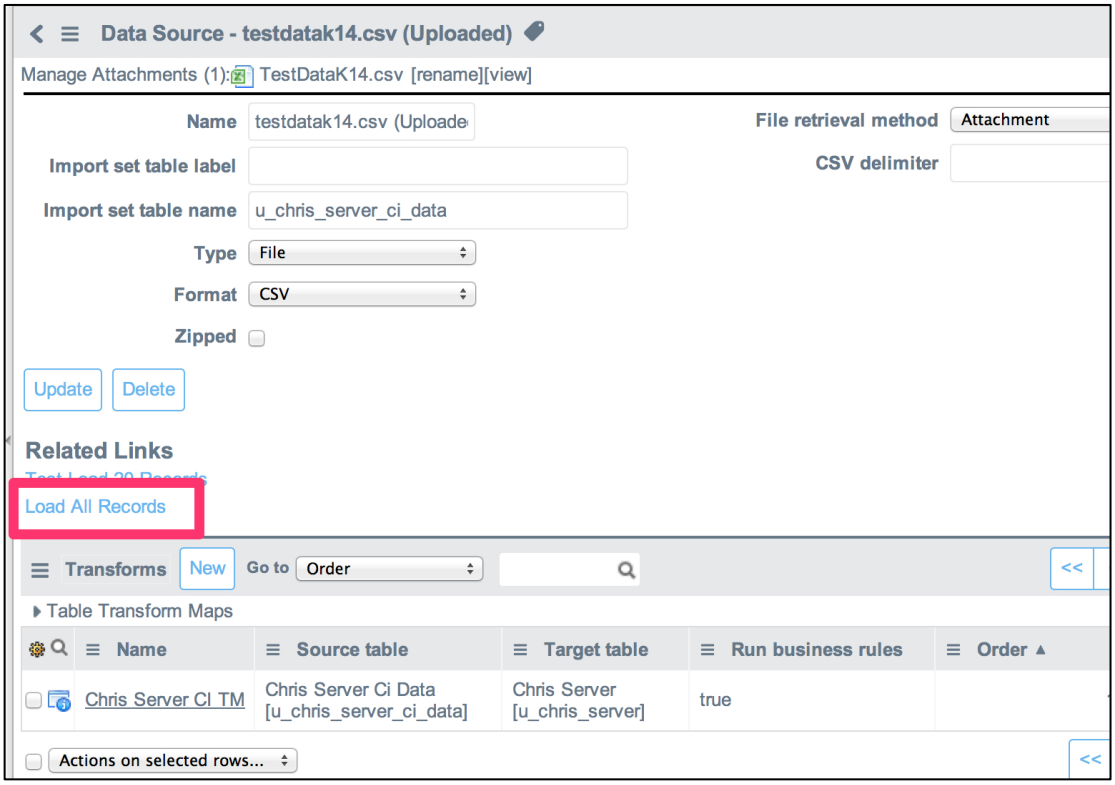



7. Click on the attachment icon and attach the file you just downloaded from the Knowledge page.

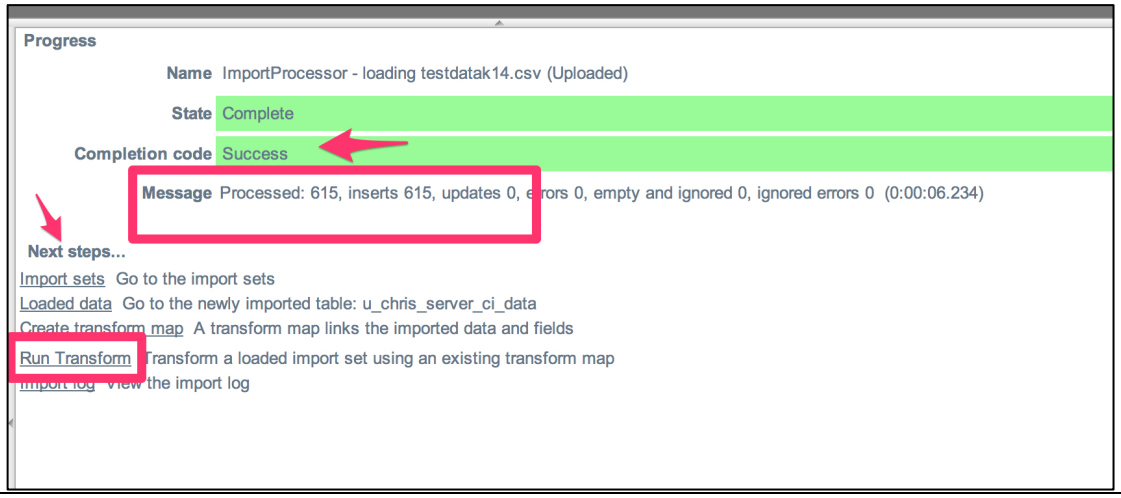



®2014 ServiceNow, Inc. All rights reserved.

8. Click the **Load All Records** related link.

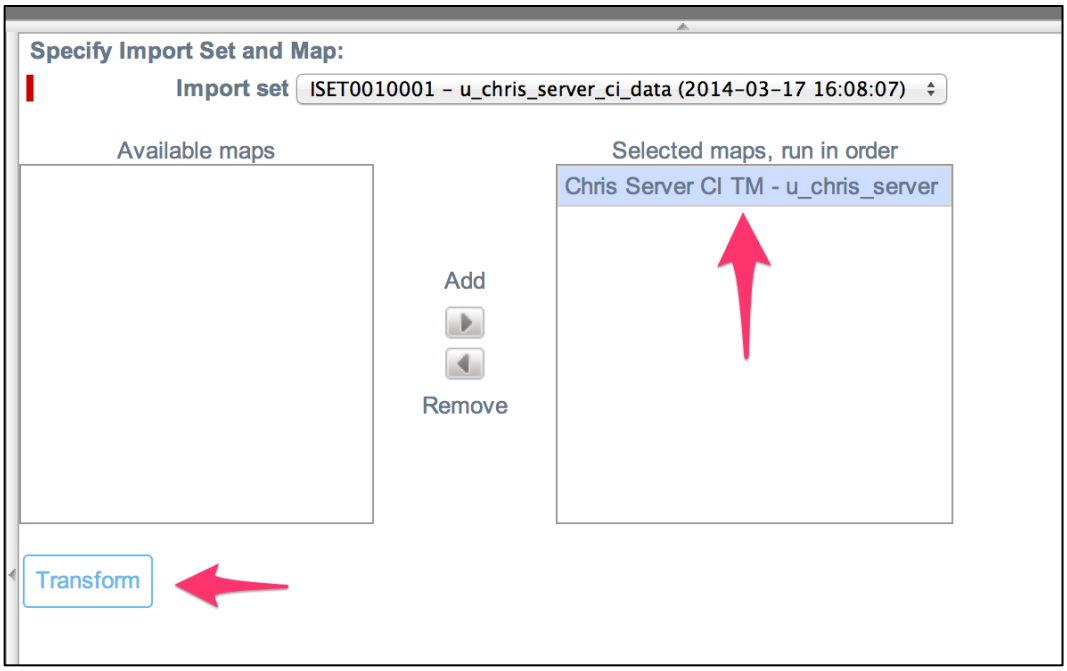



9. Make sure that **615** record inserts where made. Then click the **Run Transform** link in the **Next steps…** section.

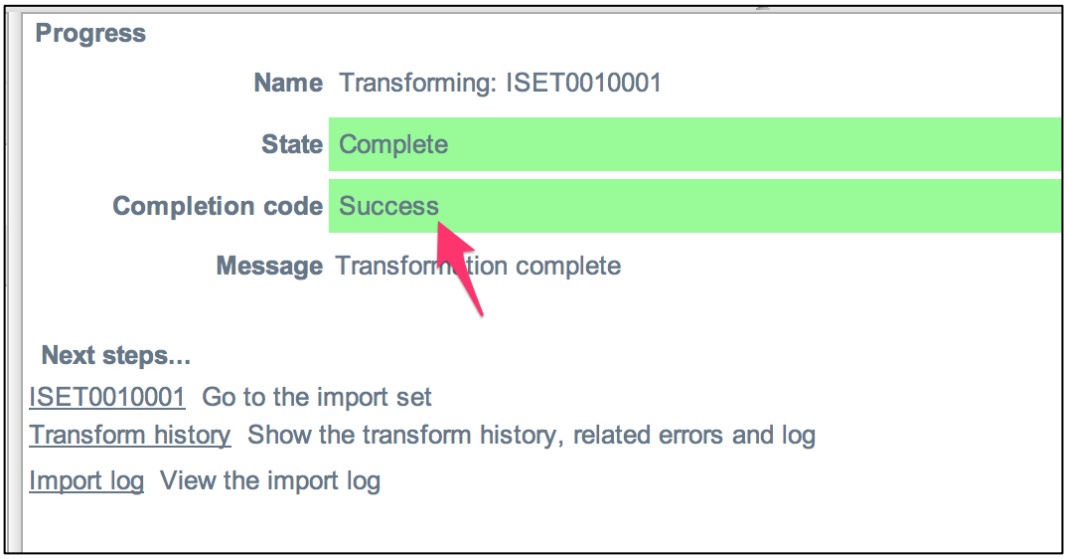



©2014 ServiceNow, Inc. All rights reserved.

10. The next screen will ask for you to select a transform map to use. There should only be one. Make sure it is on the **Selected** maps side of the slush bucket and click **Transform**.



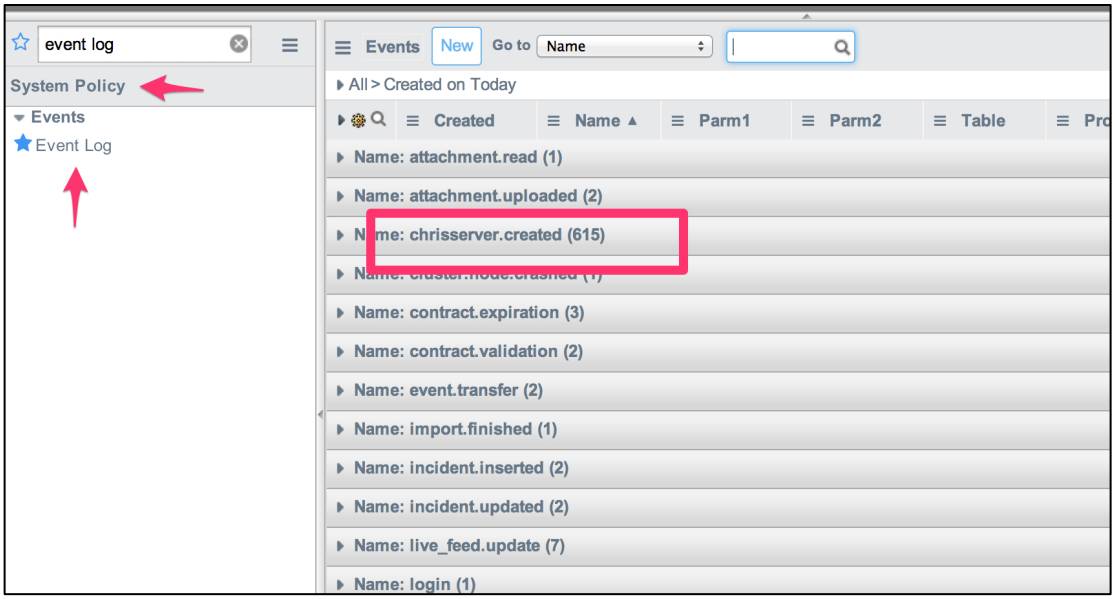

11. When the transform is complete you should see a success.




©2014 ServiceNow, Inc. All rights reserved.

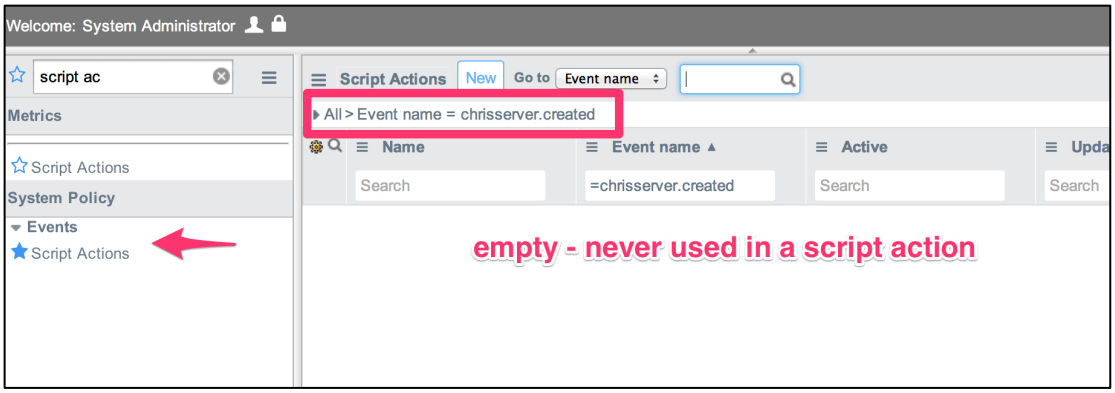
12. Navigate to **System Policy > Event Log**.

13. Right-click on the name column of the Event Log and select **Group-By Name**.



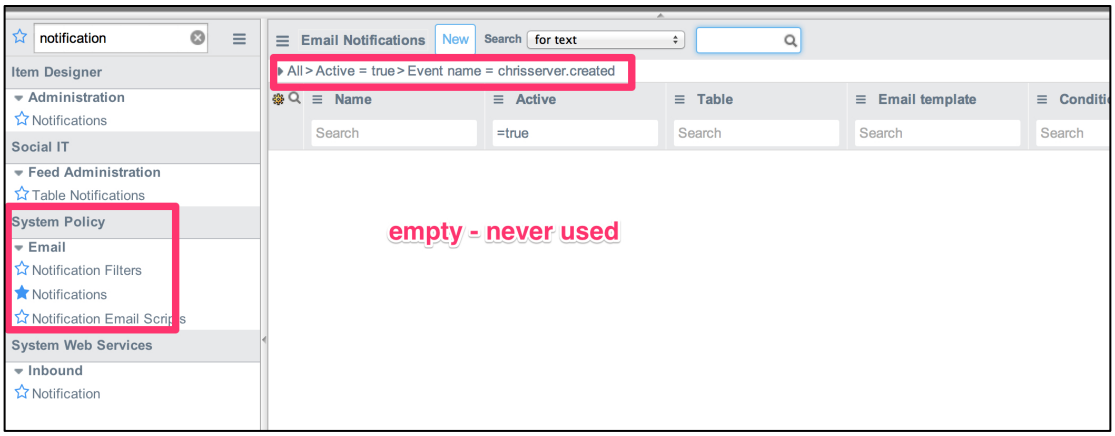

14. You can identify clearly that there is an event firing quite a lot. Let's go figure out what it does. Take note of the name **chrisserver.created**.

15. There are two main places that events are handled. First the Script Action module under **System Policy > Events**. Filter search where event name is **chrisserver.created**.




©2014 ServiceNow, Inc. All rights reserved.

16. The second place an event can be handled is the email notification. Navigate to **System Policy > Notifications** and filter search for active notifications and Event name is **chrisserver.created**.




17. We have identified an event that is firing when we upload data and is filling up the event backlog, but not doing anything. Go deactivate the business rule firing this event.

## Lab Success Verification

You have successfully completed this lab if you can identify the business rule that is firing the event and you have deactivated it.

©2014 ServiceNow, Inc. All rights reserved.

# Lab Goal

Refactor a client script to make it maintainable, more readable and better performing.

**Lab 4**
**Optimizing the client side**

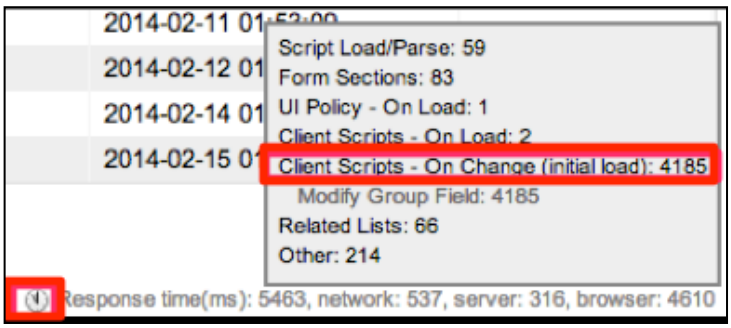
## Using GlideAjax, g_sratchpad and GSLog

1. Instructor will provide you with your own instance URL.

2. Log on with admin/admin credentials.

3. Navigate to **Change > Open > CHG0000012**.

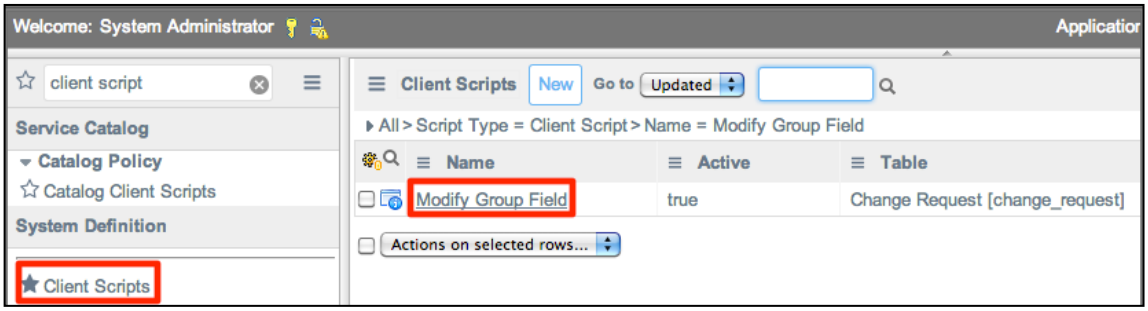
4. Scroll to the bottom of the page and investigate the Client transaction timings by clicking on the little clock ( ).




**Results**: The script **Modify Group Field** requires a lot of time**.**

©2014 ServiceNow, Inc. All rights reserved.

5. Navigate to **System Definition > Client Scripts**.




©2014 ServiceNow, Inc. All rights reserved.

6. Open the script **Modify Group Field** on the table **change_request** and replace it with this:

```
function onChange(control, oldValue, newValue, isLoading, isTemplate)
{
   //do nothing if the form is loading and the group is empty
   if ((isLoading && oldValue=="") || newValue == '') {
      return;
   }
   //if the group is filled onLoad, fill the member field
   else if(isLoading && oldValue!="")
   {
         g_form.clearOptions("u_group_member");
         fillGroupMembers(g_scratchpad.chg_groupmembers);
   }
   else //retrieve the value from
   {
      var ga = new GlideAjax('ChangeManagementUtils');
      ga.addParam('sysparm_name','getGroupMembersOnDemand');
      ga.addParam('sysparm_assignement_group', newValue);
      ga.getXML(processResponse);
      g_form.clearOptions("u_group_member");
   }
}
//process the ajax call
function processResponse(response)
{
   var answer =
response.responseXML.documentElement.getAttribute('answer');
   if (answer)
   {
            fillGroupMembers(answer);
   }
}
```

```
//function fill the group members field with the options
function fillGroupMembers(members)
{
   members = members.evalJSON(true);
   for(member in members)
   {
         g_form.addOption("u_group_member", member, members[member]);
   }
}
```

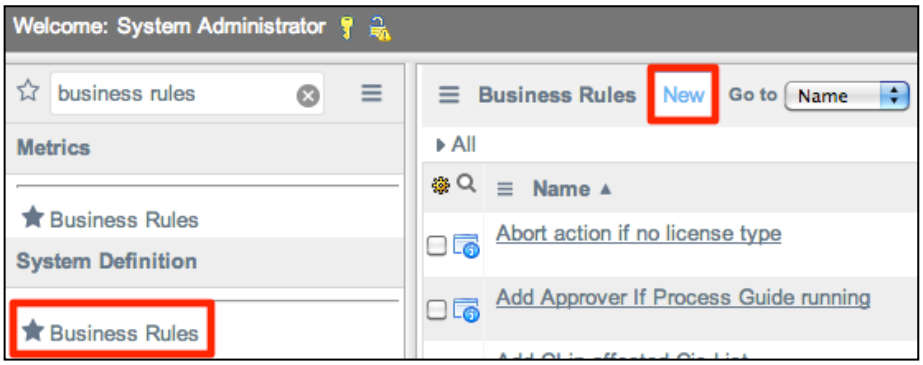
The script can be found on your instances in the Knowledge Base Article **KB0010002**.

©2014 ServiceNow, Inc. All rights reserved.

**Results**:

**OnLoad**: Only if the assignment group field is filled the script will be triggered and the values from the **onDisplay** business rule will be used.
**onChange**: A single round trip to the server has been realized by using **GlideAjax**.
The script is optimized to its best and well documented for further enhancements.

7. Navigate to **System Definition > Business Rules** and create a new one.




8. Set up the following:

Name: **Check for Assignment Group**
Table: **Change Request [change_request]**
When: **display**
Condition: **!current.assignment_group.nil()**
Script:

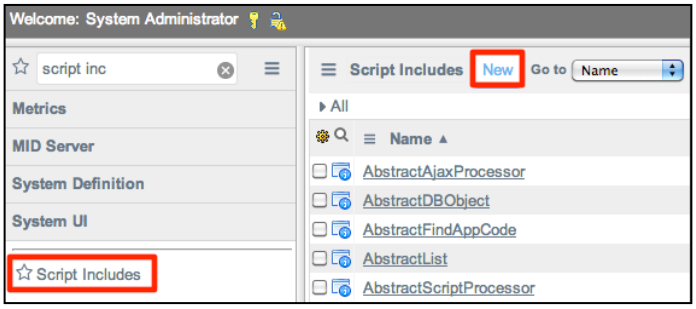
```
loadMembers();
function loadMembers()
{
	var utils = new ChangeManagementUtils();
	var json = utils.getGroupMembers(current.assignment_group);
	g_scratchpad.chg_groupmembers = json; //return array
}
```

The script can be found on your instances in the KnowledgeBase Article **KB0010003**.

**Result**: The values for the members will be pre-populated in the scratchpad and sent to the client but only in the event that an assignment group exists on the ticket.

©2014 ServiceNow, Inc. All rights reserved.

9. Navigate to **System Definition > Script Includes** and create a new one.




10. Set up the following:

    Name: **ChangeManagementUtils**
    Description: **Client & Server Side funtions for CHM**
    Client callable: **true**
    Script:

```
var ChangeManagementUtils = Class.create();

ChangeManagementUtils.prototype =
Object.extendsObject(AbstractAjaxProcessor, {

//wrapper for the AjaxCall
  getGroupMembersOnDemand: function ()
  {
     var group = this.getParameter("sysparm_assignement_group");
     return this.getGroupMembers(group);
  },
```

©2014 ServiceNow, Inc. All rights reserved.

```
//retrieve all member by name and sysid for a given group
  getGroupMembers: function (group_sysid)
  {
       var json = new JSON();
       var result = {};

      //get the relevant group member
      var grMember = new GlideRecord("sys_user_grmember");
      grMember.addQuery("group", group_sysid);
      grMember.query();

      //build or condition to get the names and ids of the others
      if (grMember.next())
      {
            var grUsers = new GlideRecord('sys_user');
            var qc = grUsers.addQuery('sys_id',
grMember.getValue("user"));
          //add the other users to the query
            while (grMember.next())
            {
                  qc.addOrCondition('sys_id',
grMember.getValue("user"));
            }
            grUsers.query();
            //build JSON and put it into the scratchpad
            while(grUsers.next())
            {
                  result[grUsers.getValue("sys_id")] =
grUsers.getValue("name");
            }
            return json.encode(result); //return array
      }
  },

//logging functionality for all your Ajax calls
  _log: function (msg)
  {
        if (typeof this.logger == "object") //if exists
        {
            this.logger.info(msg);
        }
        else //if not exists, init
        {
            this.logger = new GSLog("mycompany.process.ajax.logging",
"ChangeManagementUtils");
              this.logger.info(msg);
        }
  },

  type: "ChangeManagementUtils"
});
```

©2014 ServiceNow, Inc. All rights reserved.

The script can be found on your instances in the KnowledgeBase Article **KB0010003**.

**Results**: The script include reuses code and offers in addition a function to for logging. In case of an error the logging could be switched on and off easily without interfering with other parts of the system.

## Lab Success Verification

1. You have successfully completed this lab if you can identify the performance increase on when loading the Change Management form.

©2014 ServiceNow, Inc. All rights reserved.

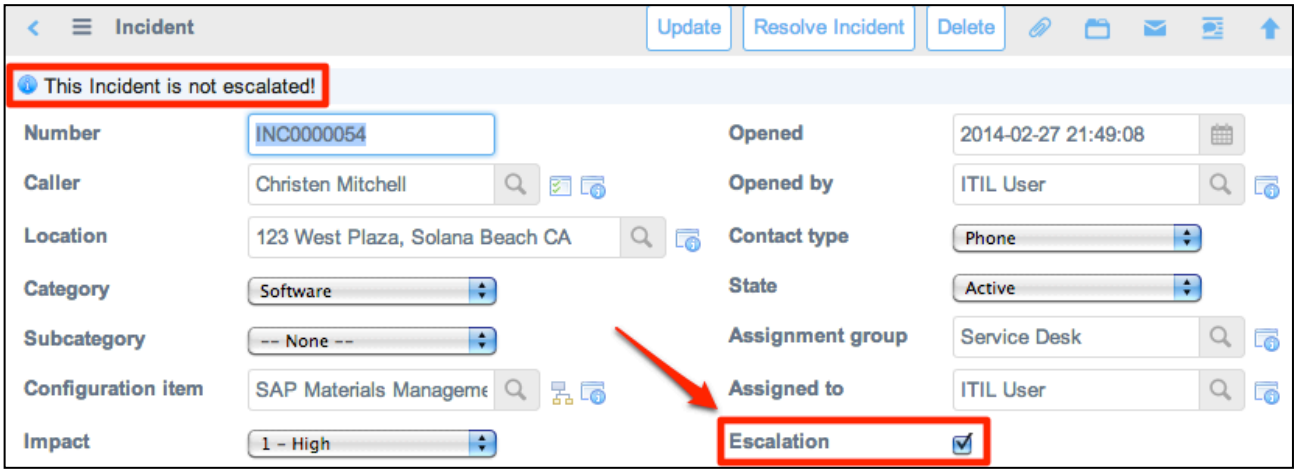
# Lab Goal

Use the browser build in debugger to fix a client script.

**Lab 5**
**Debugging the Client Side**

## Using a Debugger

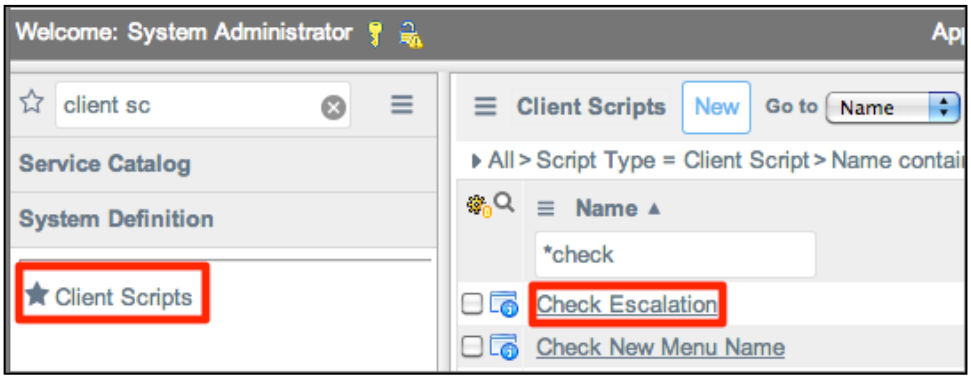
1. Instructor will provide you with your own instance URL.
2. **Use Chrome** and log on with admin/admin credentials.
3. Navigate to **Incident > Open > INC0000054**
4. Have a look at the top of the content frame and you will find a message appear.



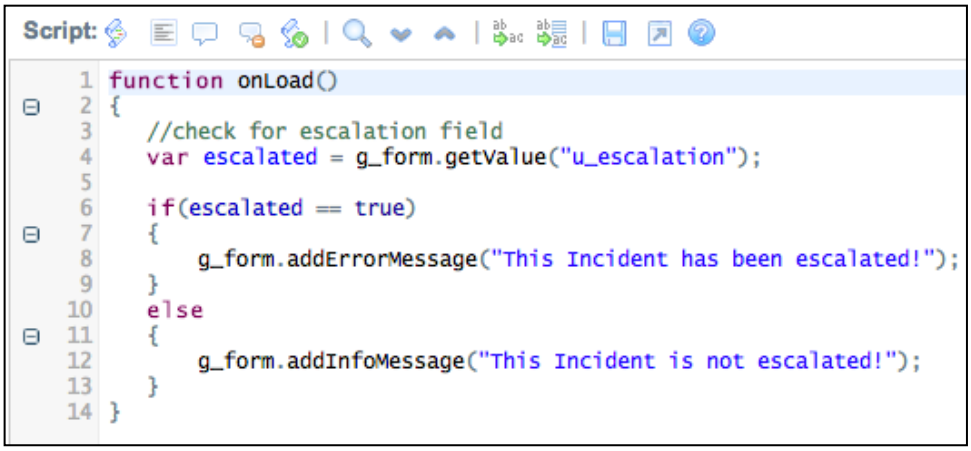

5. Someone escalated this Incident because the **Escalation** field is set to **true**. The message therefore is wrong and has to be fixed.

©2014 ServiceNow, Inc. All rights reserved.

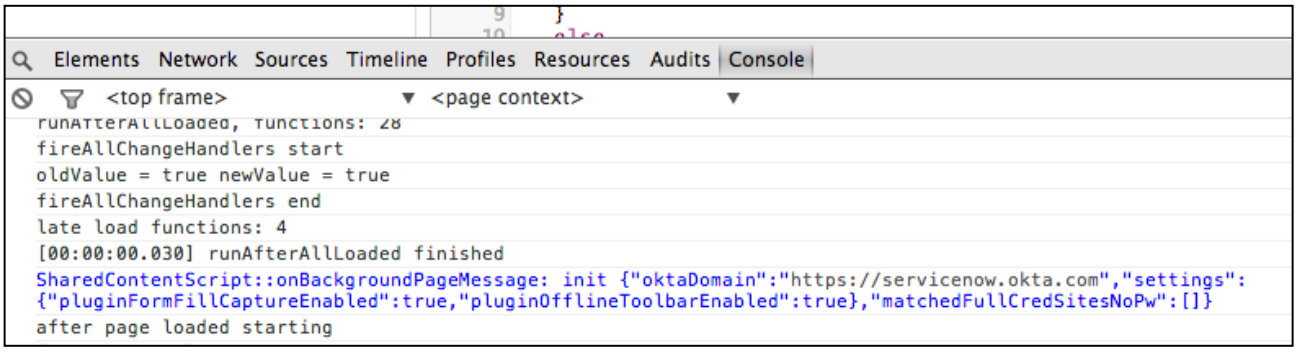
6. Navigate to **System Definition > Client Scripts** and open **Check Escalation** on the incident table




7. The script checks the **onLoad** for the value of the **Escalation** field and sets a message accordingly. For INC0000054 it should set the message **This incident has been escalated!**



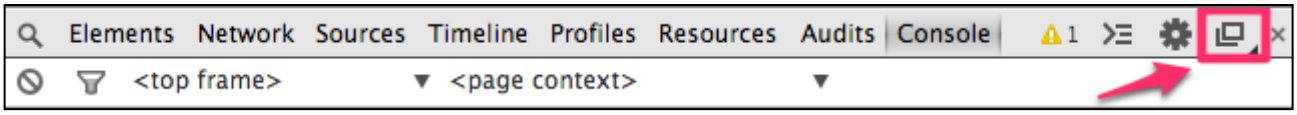

8. Open **Chrome DevTools** by clicking **CMD + Alt + i**. The DevTools will now appear on the bottom of your screen.



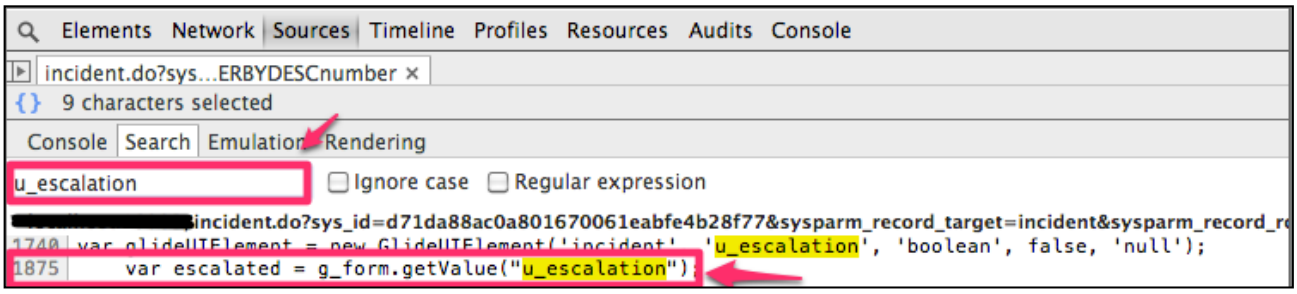

©2014 ServiceNow, Inc. All rights reserved.

9. Navigate back to **Incident > Open > INC0000054**.

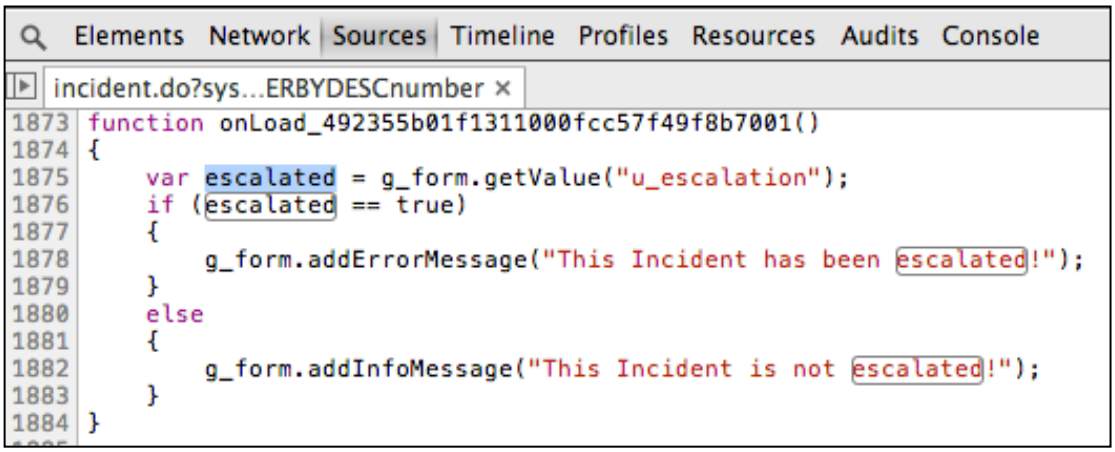
10. Press **ESC** or click on the icon ( ) on the upper right of the DevTools.




11. In the new bar that appears you can search the whole site for any content. In our case we want to find the onLoad client script to set breakpoint and debug it.
Click on **Search** and type in the field name **u_escalated**. There are two results returned and we will take a closer look to the second on, which seems to be the line of our client script.




12. Click on the second search result and you will see the correct script popping up in the background. Press **ESC** again to close the search screen.

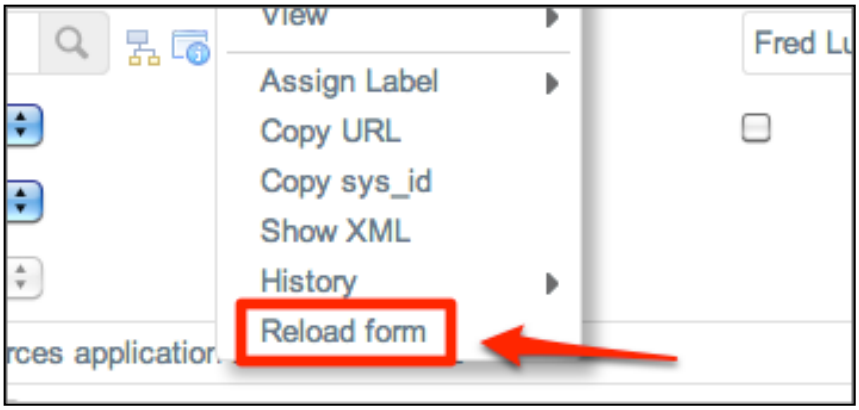



©2014 ServiceNow, Inc. All rights reserved.

13. At this point in time we know that the comparison (if-statement) goes wrong. Our assumption is that the variable “escalated” delivers the wrong values.
    The double check that we have to set a breakpoint at this line by clicking the line number in front of it. It is possible to set multiple breakpoints. The browser will stop executing at this point and the debugger will take over.

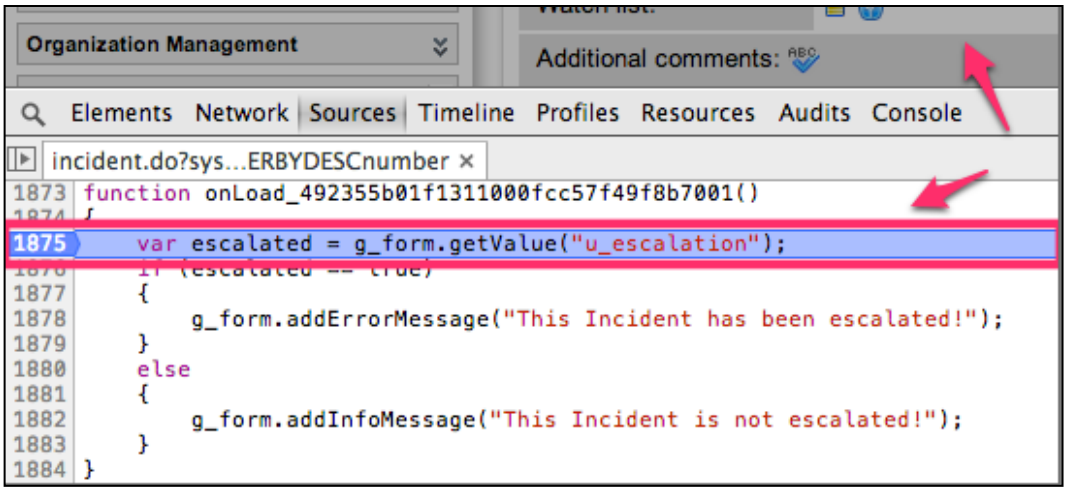



14. Now simply reload the INC by clicking the header bar and pressing **Reload Form**.

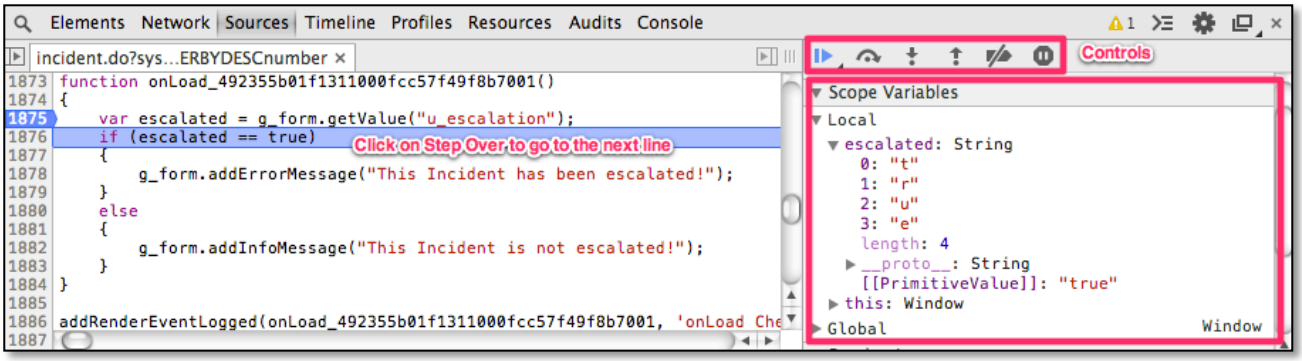



©2014 ServiceNow, Inc. All rights reserved.

15. The browser is now stopping at the breakpoint and we can control the behavior. An indicator is that the line is colored in blue and the screen locked in gray.



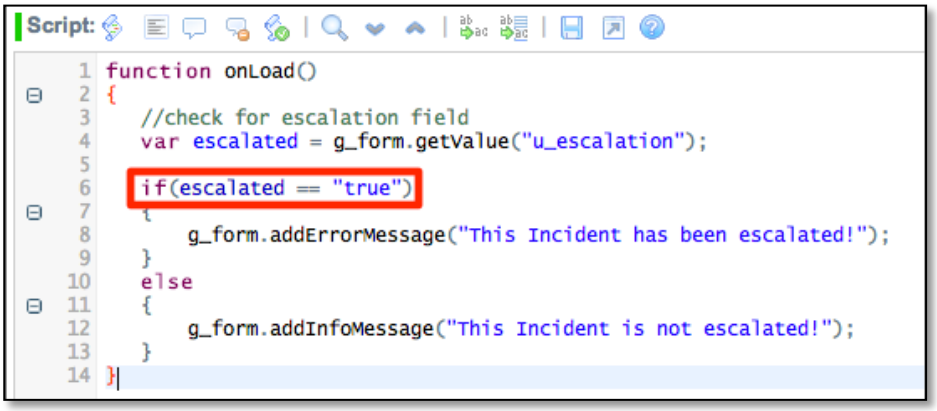

16. Press on **Step Over** ( ) to go to the next line in the code. On the top right side you will find the controls, which allow you to step over (evaluate) to the next line in the code, step in to function, step out of functions or stop the debugging.
In addition you can investigate below the **Scope Variables**. This box shows all the initiated and available variables in the scope of the function.




17. In the **Scope Variables** box we can see that our variable “escalated” will be filled by the getValue() function with the value “true” and is of type String.
Our if-statement checks for the Boolean value true, which is not the same as the string value “true”. That’s the cause why the script is failing.

©2014 ServiceNow, Inc. All rights reserved.

18. Fix the script by modifying the script like this:




**Results**: The script will now work properly because the if-statement can evaluate against the correct type of variable.

## Lab Success Verification

1. Open the Incident again and see the message properly changing according to the field **Escalation**.

©2014 ServiceNow, Inc. All rights reserved.