knowledge15®
April 19–24, 2015 • Mandalay Bay • Las Vegas, NV

Lab Guide

Incident 201:

Major Incident and On-Call Rotation with Notify

Michael Ritchie & Chris McGrath

Login / Passwords:

admin / Knowledge15

itil / Knowledge15

employee / Knowledge15

This

Page

Intentionally

Left

Blank

Lab Goal

Twilio Account properties were automatically provisioned to your instance. This lab explains how to verify that your instance is communicating properly with Twilio. This is a pre-requisite for all other labs.

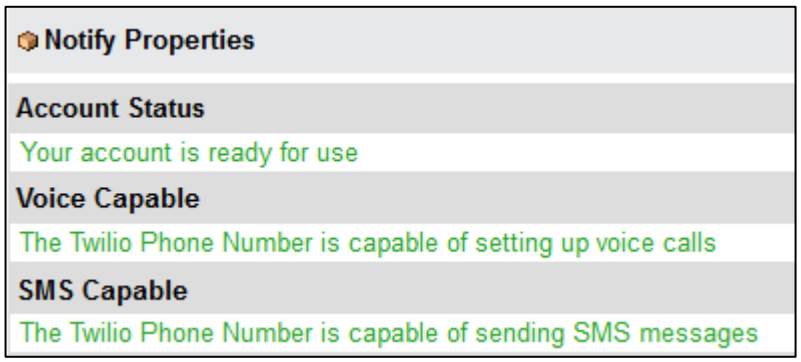
Lab 1
Verify
Notify
Properties

View the Notify Application Properties Page

1. In the Filter Text box, type **notify**.
2. Click **Properties** in the Notify Application under Administration.
3. View the messages at the top of the Notify Properties page.

Lab Success Verification

1. You should receive several messages confirming a successful setup. If not please raise your hand for help.



2. Navigate to **Lab Management > Report Lab Progress. Click I am done!**

©2015 ServiceNow, Inc. All rights reserved.

Lab Goal

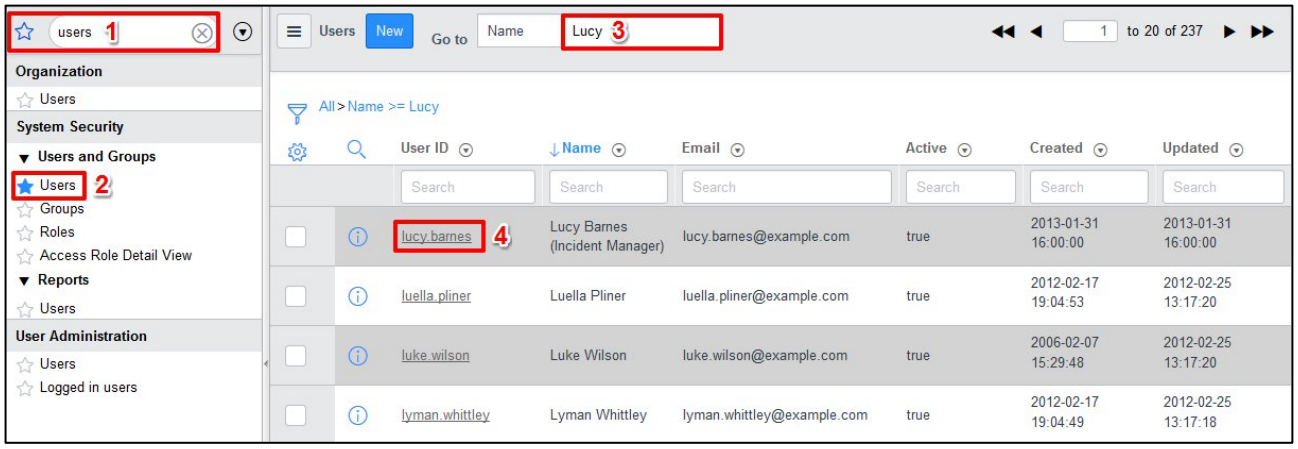
Notify requires that phone numbers be in E.164 format. The Dublin release introduced a new field type of Phone Number (E.164).

This lab explains how to change the user record Mobile Phone field type to Phone Number (E.164) and update an out-of-the-box user with your mobile phone.

Lab 2
Update User

Update User Record Mobile Phone Field Type and Enter Mobile Phone

1. In the Filter Text box, type **users**.



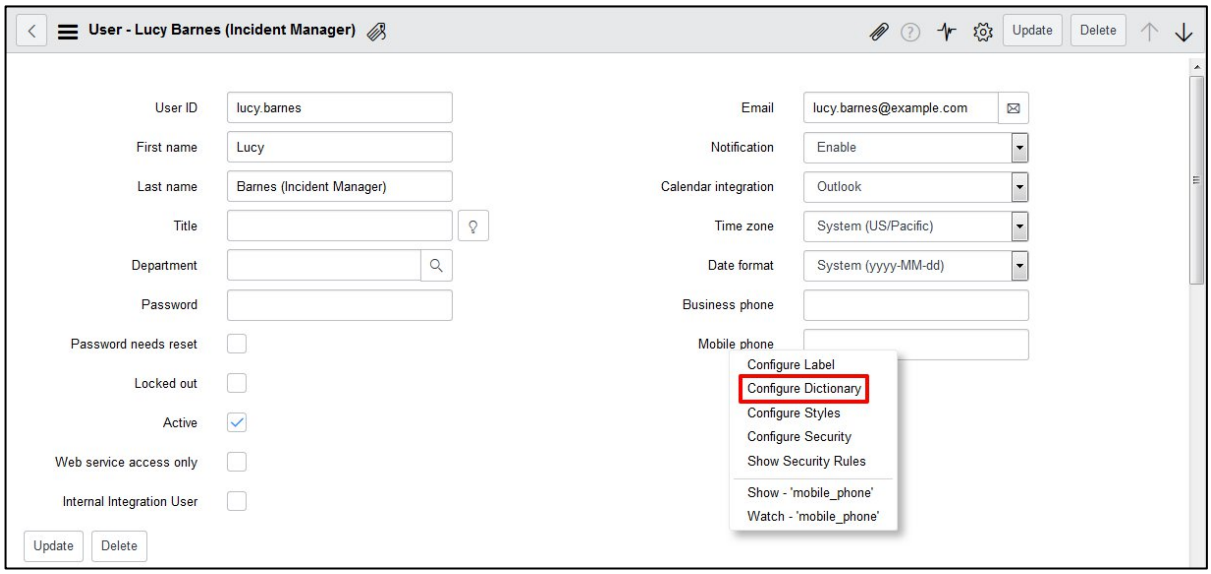
2. Select **Users** under the System Security application.

3. Using the list search at the top-middle of your screen, search for the Name **Lucy**.

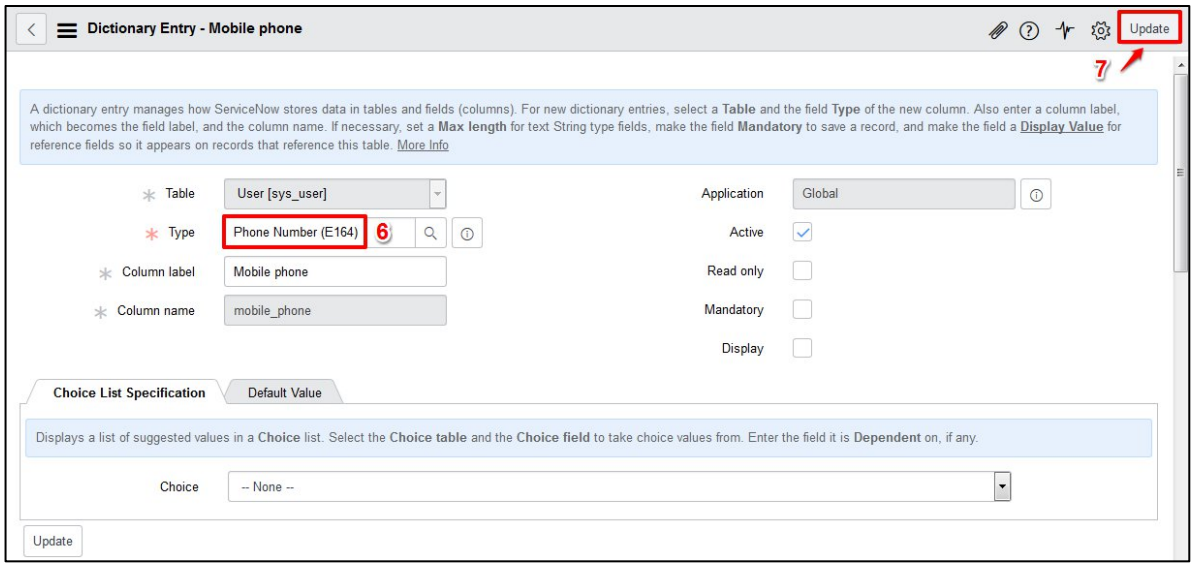
4. Click **lucy.barnes** in the list.

©2015 ServiceNow, Inc. All rights reserved.

5. Twilio requires that phone numbers be in E.164 format. When Lucy Barnes' user record appears, right-click the Mobile phone label and choose **Configure Dictionary**.



6. Change the type of this field from Phone Number to **Phone Number (E164)**.



7. Click **Update**.

®2015 ServiceNow, Inc. All rights reserved.

8. Now that the Mobile phone field is of type E.164, a new select box containing regions appears next to the Mobile phone label. Choose your personal region and enter your mobile number in the field beside it. Format does not matter; ServiceNow automatically corrects it to the proper format with next step.

   Note: Out of the box, only a few regions appear. If you do not find your region in the dropdown list, please raise your hand for help enabling your region.

9. After you click out of the **mobile phone** field, ServiceNow converts it to the proper format.

10. Click **Update** to save Lucy's user record.

11. Update one more user for later use in the Lab. Use the list search at the top middle of your screen to search for the Name **Beth Anglin**.

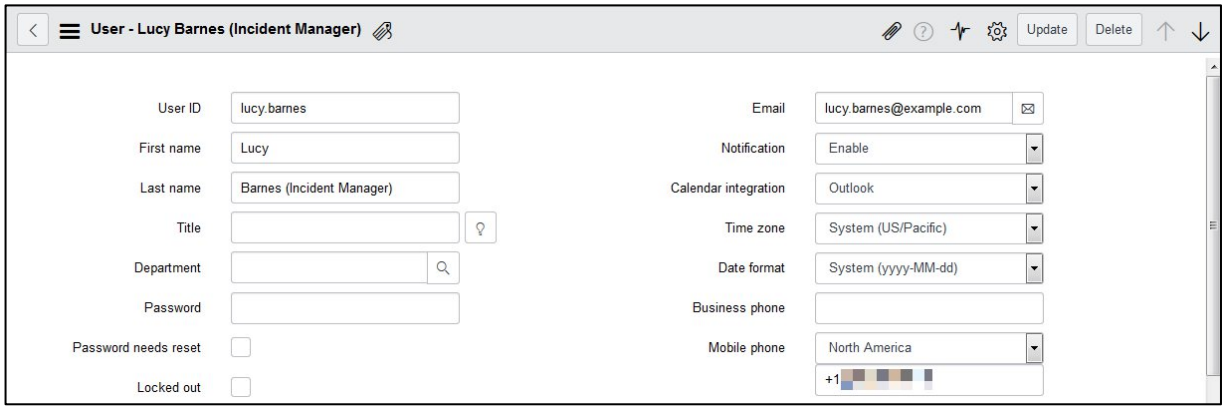
12. Click **beth.anglin** in the list.

13. In the Mobile phone field, choose your personal region and enter your mobile number in the field beside it. After you click out of the mobile number field, ServiceNow converts it to the proper format.

14. Click **Update** to save Beth's user record.

Lab Success Verification

1. Lucy Barnes' and Beth Anglin's user records have been updated with your mobile phone.



2. Navigate to **Lab Management > Report Lab Progress**. **Click I am done!**

©2015 ServiceNow, Inc. All rights reserved.

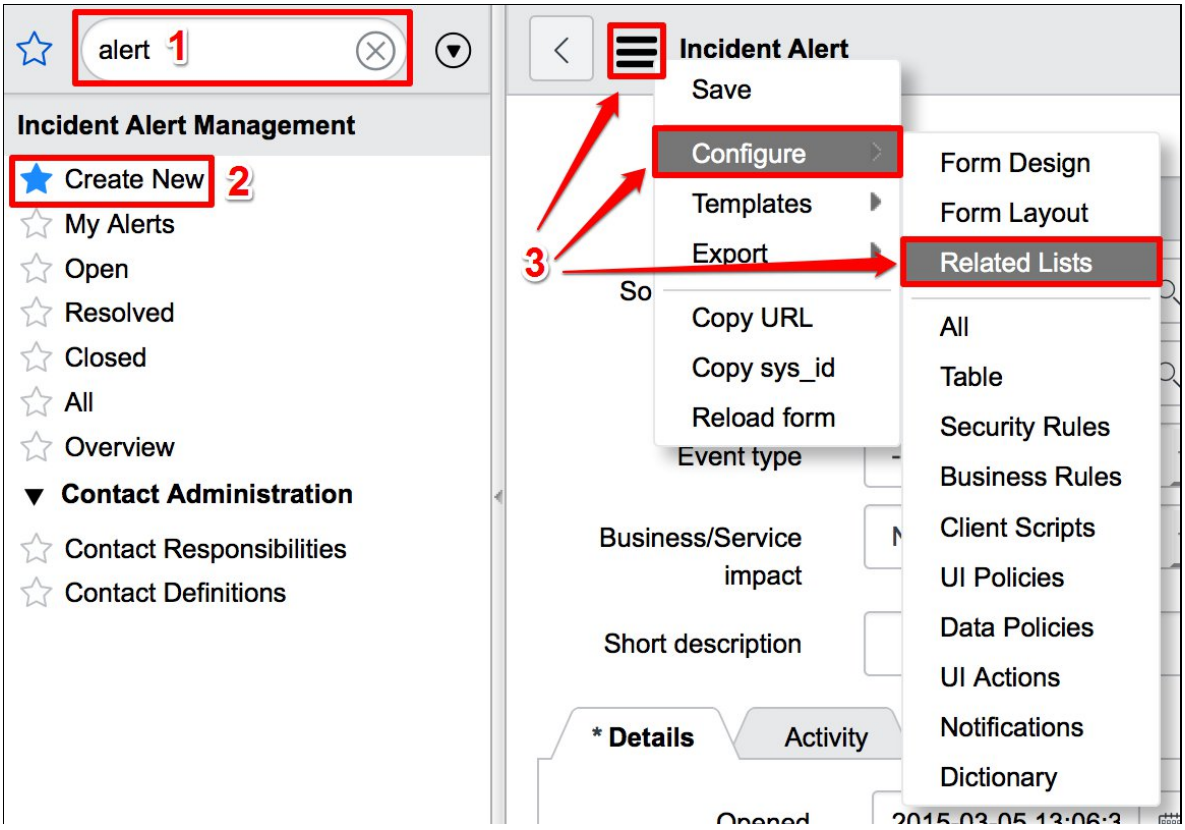
Lab Goal

This lab explains how to Add two related lists to the bottom of the Incident Alert form, which allows you to view SMS and Voice Calls associated with an Incident Alert Record.

Lab 3 Modify Incident Alert Form

Modify Incident Alert-Related Lists

1. In the Filter Text box, type **alert**.



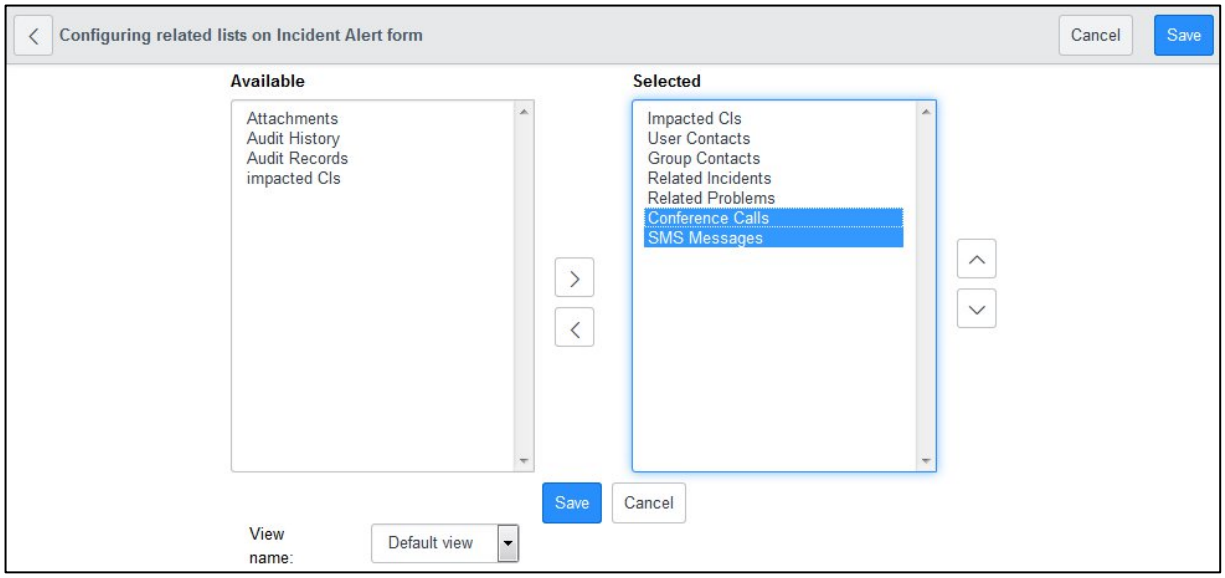
2. Click **Create New** in the Incident Alert Management pane.

©2015 ServiceNow, Inc. All rights reserved.

3. Click the **Incident Alerts** icon, then **Configure**, and finally **Related Lists**.

4. When the slush bucket appears, double-click **Conference Calls** on the **Available** list (left side) to move it to the **Selected** list (right side). Repeat for **SMS Messages**.

5. Click **Save**.



Lab Success Verification

1. After clicking **Save**, the screen should return to a blank Incident Alert Record.

2. Navigate to **Lab Management > Report Lab Progress. Click I am done!**

©2015 ServiceNow, Inc. All rights reserved.

Lab Goal

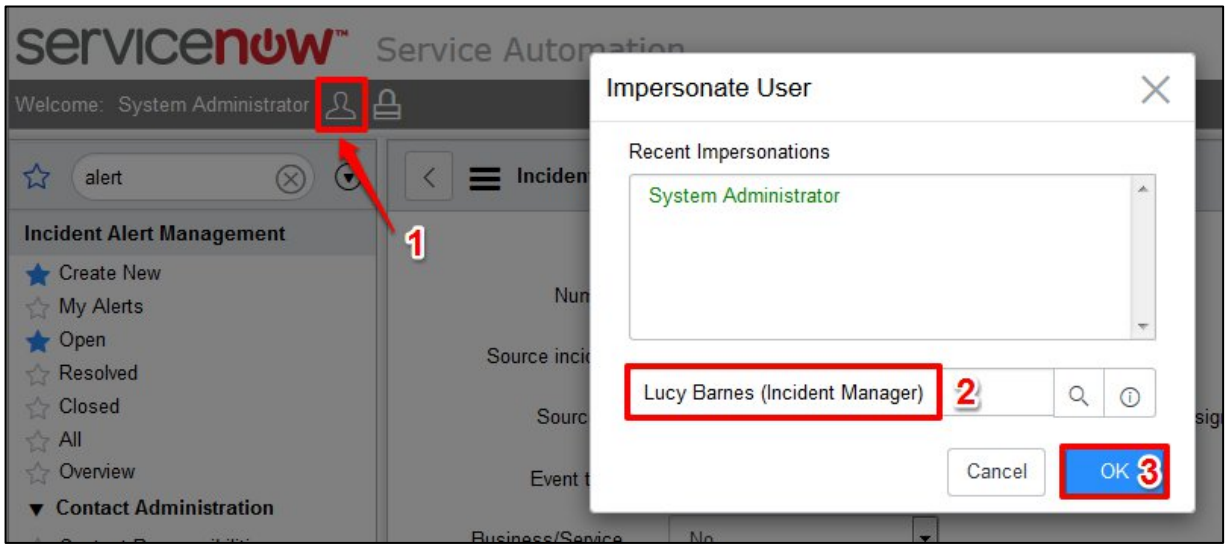
This lab explains how to create an **Incident Alert**.

Lucy Barnes is an Incident Manager and she wants to review existing incidents to determine if any Major Incidents are occurring. She decides to open an Incident Alert to communicate the outage and get the right people involved.

Lab 4
Create Incident Alert

Create an Incident Alert from an Existing Incident

1. Click **Impersonate User** to log in as Lucy Barnes.



2. After the Impersonate User dialog appears, type **Lucy Barnes** and select her in the dropdown list.

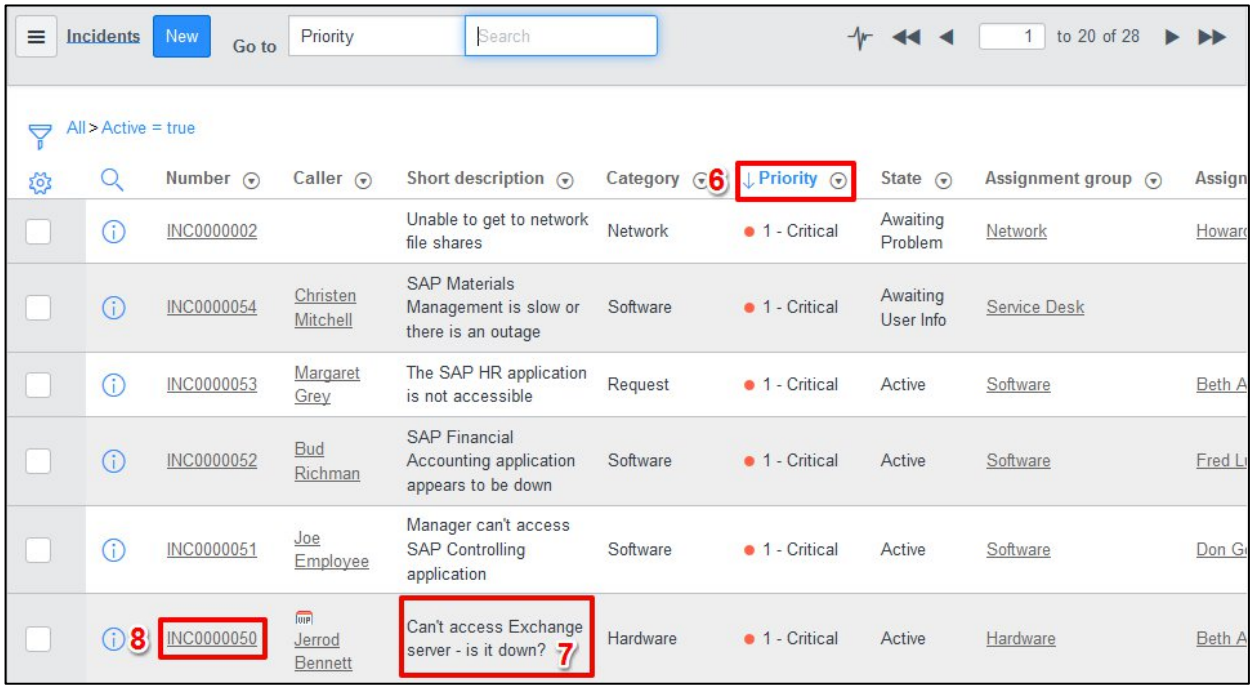
3. Click **OK**.

4. After you are logged in as Lucy, in the Filter Text box, type **incident**.

5. Click **Open** under the Incident Application.

©2015 ServiceNow, Inc. All rights reserved.

6. Lucy wants to focus on Critical Priority incidents. This can be done by several methods in ServiceNow. For this lab, click the **Priority** column heading to sort the list with **1-Critical Priority** incidents at the top of the list.

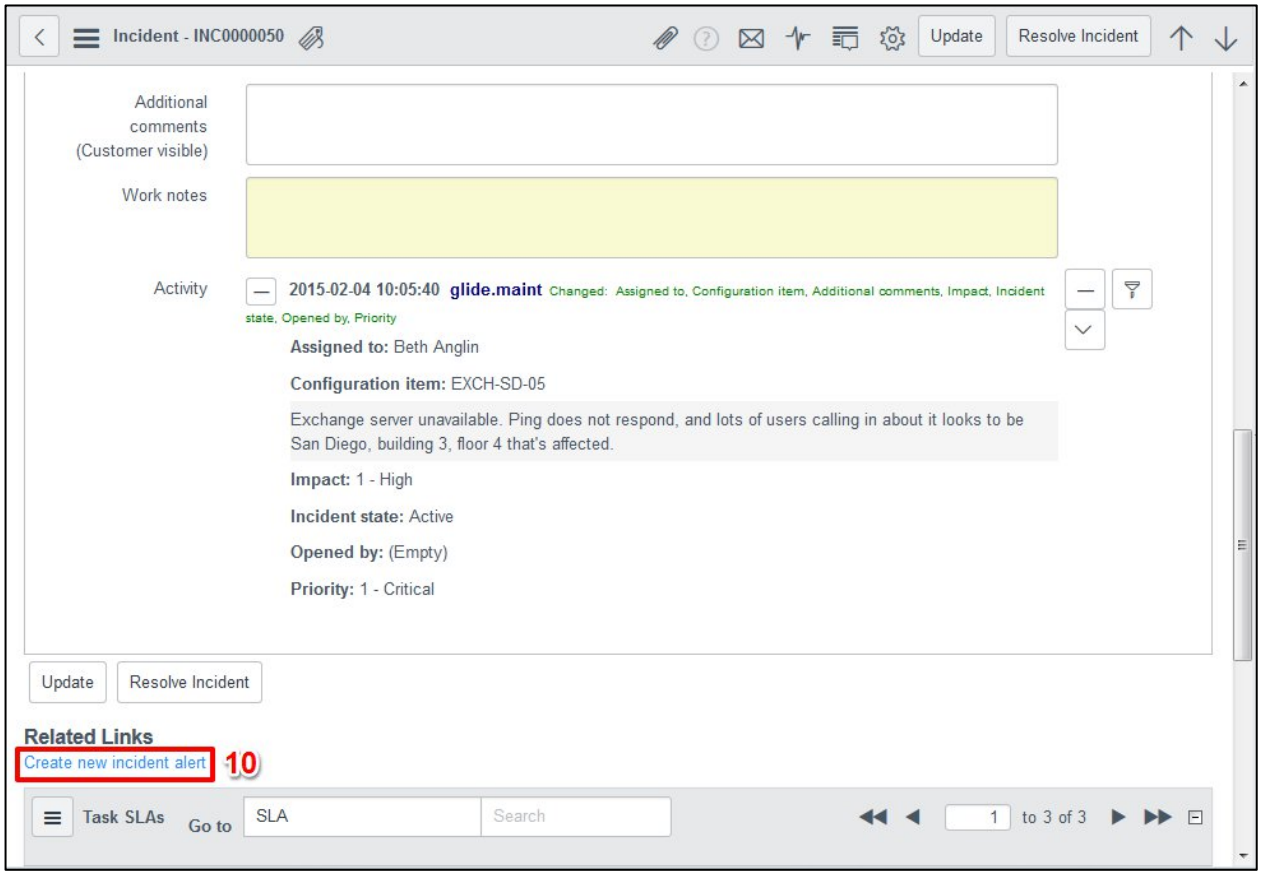


7. Lucy notices that the Exchange server has been reported down in Incident I**NC0000050**.

8. Click **INC0000050** in the list.

®2015 ServiceNow, Inc. All rights reserved.

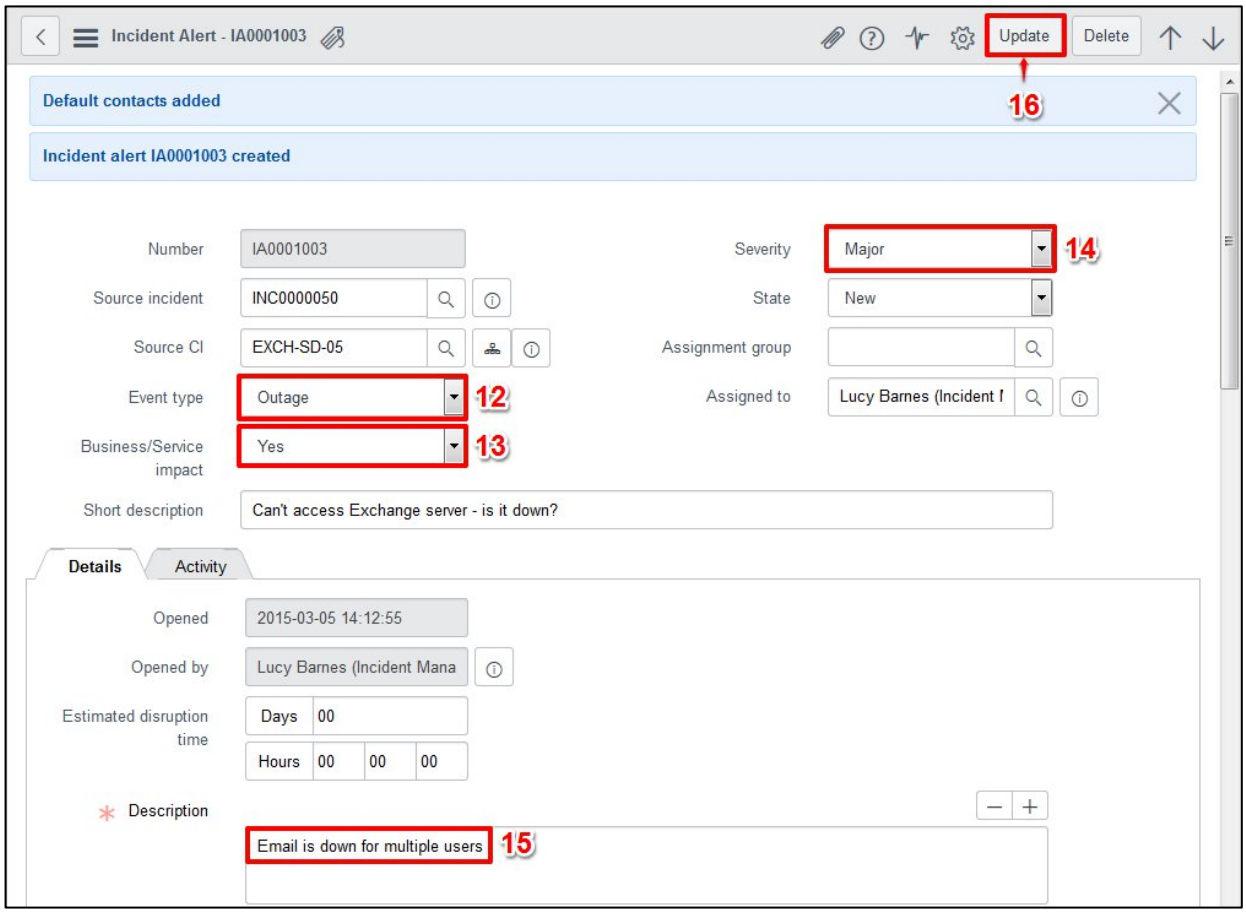
9. Lucy notices that the server is unavailable and multiple users are impacted by this incident. She wants to create an Incident Alert for this incident to communicate this outage. Scroll to the **Related Links** just above the Task SLAs list.



10. Click **Create new incident alert**.

©2015 ServiceNow, Inc. All rights reserved.

11. After the Incident Alert record is created, notice that the Source Incident, Source CI, and Short Description are copied from INC0000050. Now details about the Incident Alert need to be set.



12. Set the Event type to **Outage**.

13. Set the Business/Service Impact to **Yes**.

14. Set the Severity to **Major**.

15. Set the Description to **Email is down for multiple users**.

16. Click **Update**.

©2015 ServiceNow, Inc. All rights reserved.

17. What happened while you were updating the Incident Alert? Did you get an SMS message? You should have received one similar to the example.



18. Scroll to the bottom of the Incident and click **Show related incident alert**. Incidents can only be linked to one Incident Alert, but an Incident Alert can have many incidents linked to it.

19. Go to the Activity section (click the **Activity** tab). Notice all the updates that were done in the previous step were captured as an activity. Also notice that there is an activity stating **Sent SMS to Lucy Barnes (Incident Manager)**.

20. Scroll to the bottom of the **Incident Alert**. Go to the **Impacted CIs** section (or click the **Impacted CIs** tab, if tabs enabled). Notice that all of the business services that are dependent on the **EXCH-SD-05** CI have automatically been listed. This leverages the CMDB relationship hierarchy.

21. Go to the User Contacts section (or click the **User Contacts** tab, if tabs enabled) and notice that Lucy Barnes is listed as the Incident Manager for this Incident Alert. Incident Alert has contact responsibilities where Users and Groups can automatically be associated to an Incident Alert.

22. Go to the Group Contacts section (or click the **Group Contacts** tab, if tabs enabled) and notice that several groups were added to this Incident Alert. Incident Alert has contact responsibilities where Users and Groups can automatically be associated to an Incident Alert.

Note: Lab 6 presents an example of how these groups are used.

©2015 ServiceNow, Inc. All rights reserved.

23. Go to the Related Incidents section (or click the **Related Incidents** tab, if tabs enabled) and notice that INC0000050 is listed automatically. Incident Alerts can have multiple Incidents associated to them. If INC0000050 was a parent of multiple incidents, the child incidents would also be listed here.

24. Go to the SMS Messages section (or click the **SMS Messages** tab, if tabs enabled) and notice that the Incident Alert has a record of all SMS alerts that were sent from the Incident Alert. There is an out-of-the-box business rule that sends an SMS message to all of the contacts for an Incident Alert. The status of the SMS is also **Sent** because we received verification from Twilio that the message was successfully sent.

Lab Success Verification

1. An incident alert was created and you successfully received an SMS message from the Incident Alert.

2. Navigate to **Lab Management > Report Lab Progress. Click I am done!**

®2015 ServiceNow, Inc. All rights reserved.

Lab Goal

This lab explains how to create a new contact definition to add a Communications Manager to Incident Alerts that are of type Outage where the Severity is Major.

Lab 5 Modify Contact Definitions

Create a New Contact Definition for Incident Alert

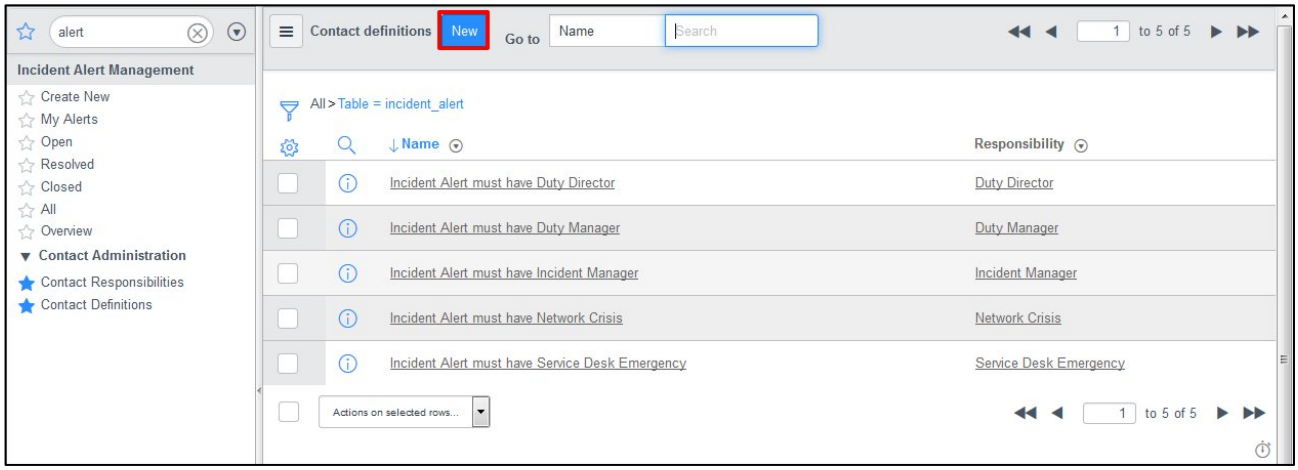
1. In the Filter Text box, type **alert**.

2. Click **Contact Responsibilities** in the Incident Alert Management application. This is a list of the out-of-the-box Responsibilities included with the product. New contact responsibilities can easily be added by Administrators and Incident Alert Administrators.

3. Click **Contact Definitions**. Notice that ServiceNow has several out-of-the-box contact definitions. We saw a demonstration of these when we created the incident alert in the previous exercise in which a Duty Director, Incident Manager, and Duty manager were added along with the Service Desk Emergency and Network Crisis Groups.

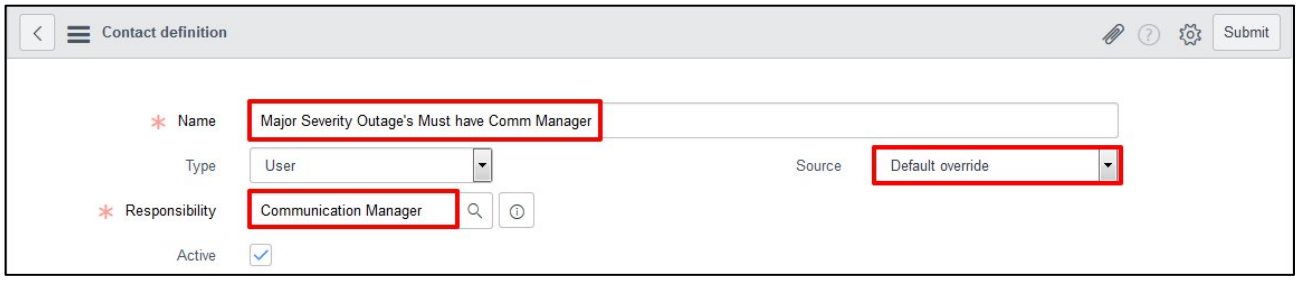
©2015 ServiceNow, Inc. All rights reserved.

4. Click **New**.



5. Enter the following Information:

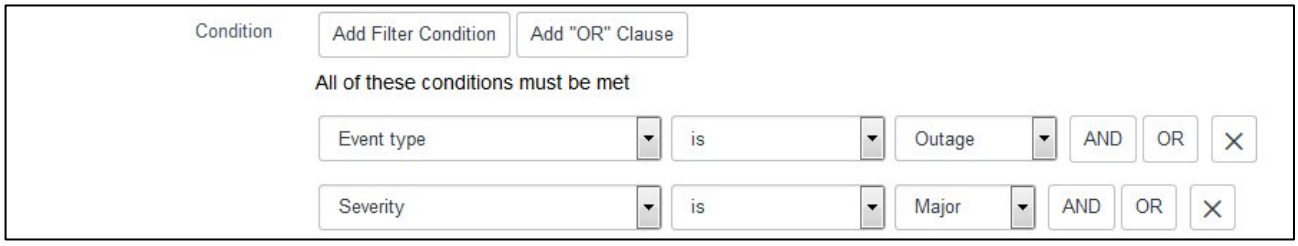
   Name: **Major Severity Outage's Must have Comm Manager**
   Responsibility: **Communication Manager**
   Source: **Default override**



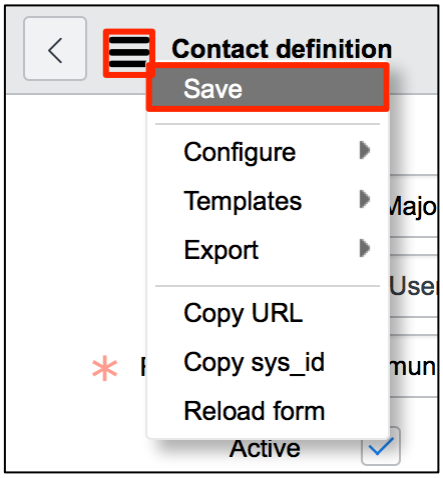
6. Add the condition to add the Communication Manager only if the Incident Alert type is **Outage** and the Severity is **Major**. In the condition, click **choose field** and select **Event Type**. Leave the middle box set to **is**. Click the right **None** select box and choose **Outage**.

©2015 ServiceNow, Inc. All rights reserved.

7. Click the **and** button beside **Outage** to add another condition and repeat for Severity. Click **choose field** and select **Severity**. Leave the middle box set to **is**. Click the right **None** select box and choose **Major**.

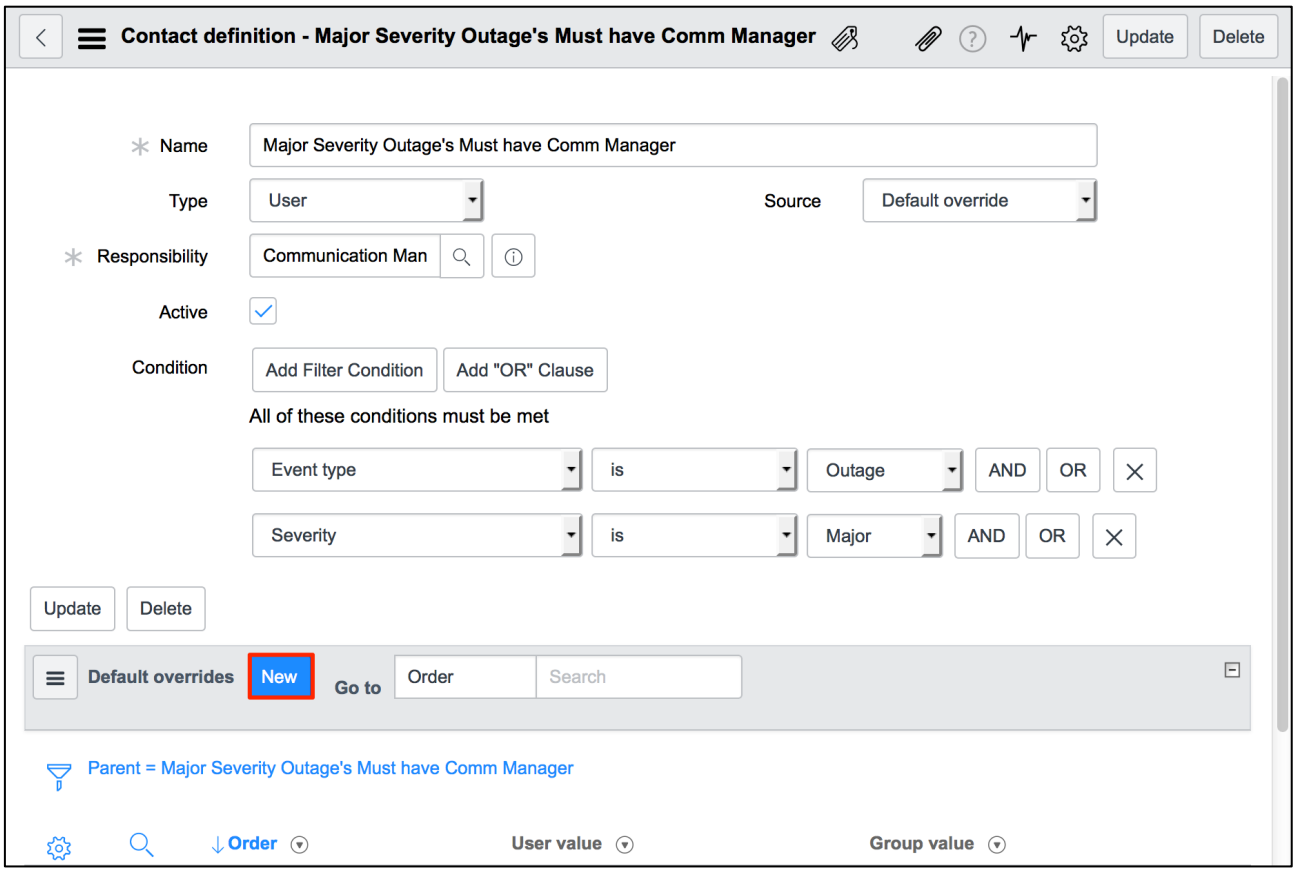


8. Click **Additional actions** and select **Save**, which leaves the screen on this record.

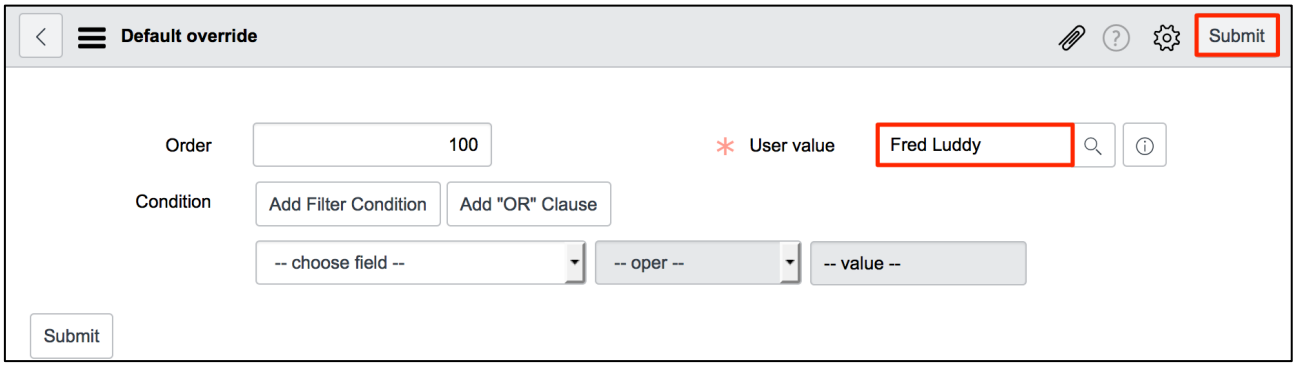


©2015 ServiceNow, Inc. All rights reserved.

9. Click **New** in the **Default overrides** section at the bottom of the form.

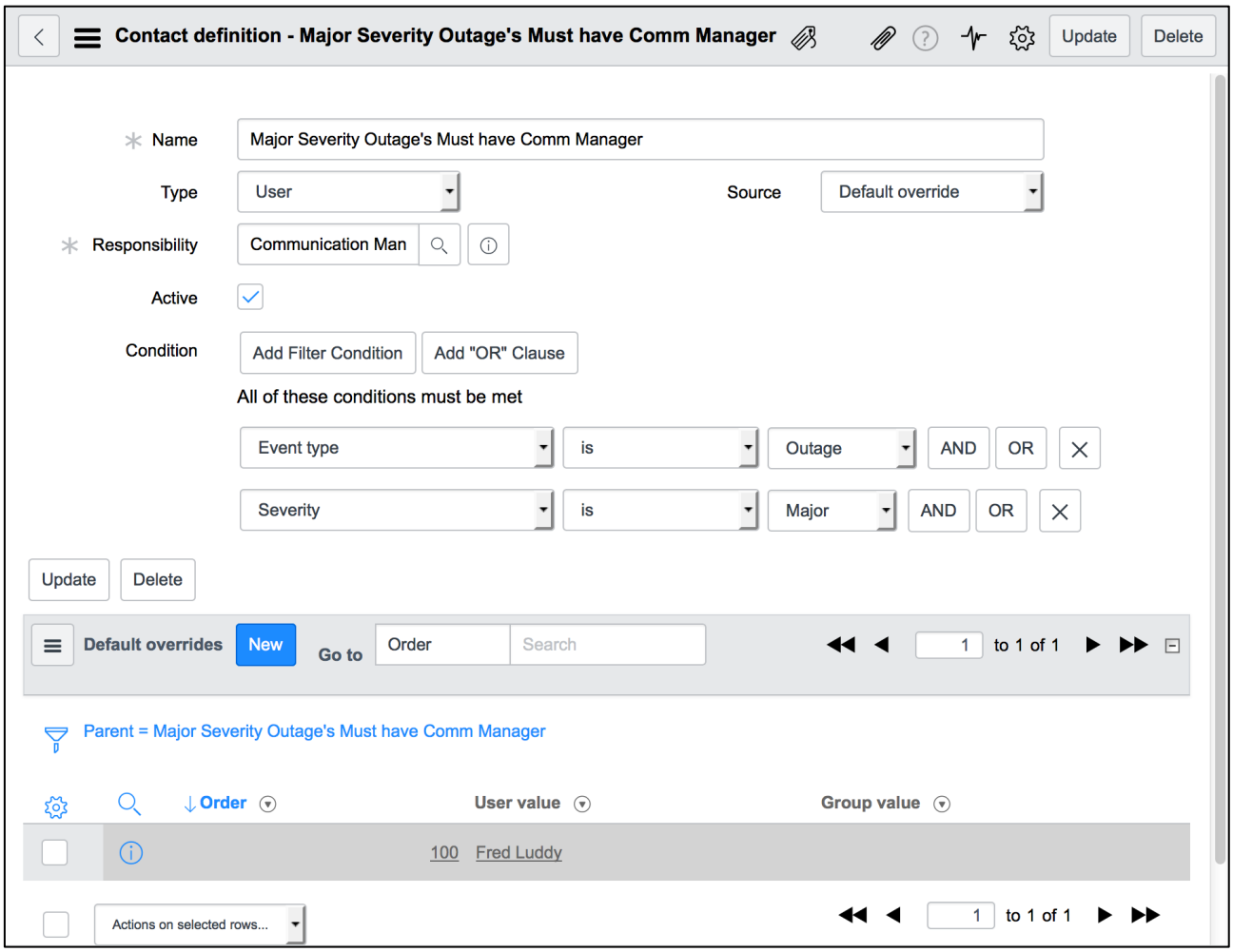


10. In the **User value** field, type **Fred Luddy** and click **Submit**.



®2015 ServiceNow, Inc. All rights reserved.

11. Once this is set up, your Contact Definition should look like the example.



12. Test this new rule. Click **Create New** in the Incident Alert Management application.

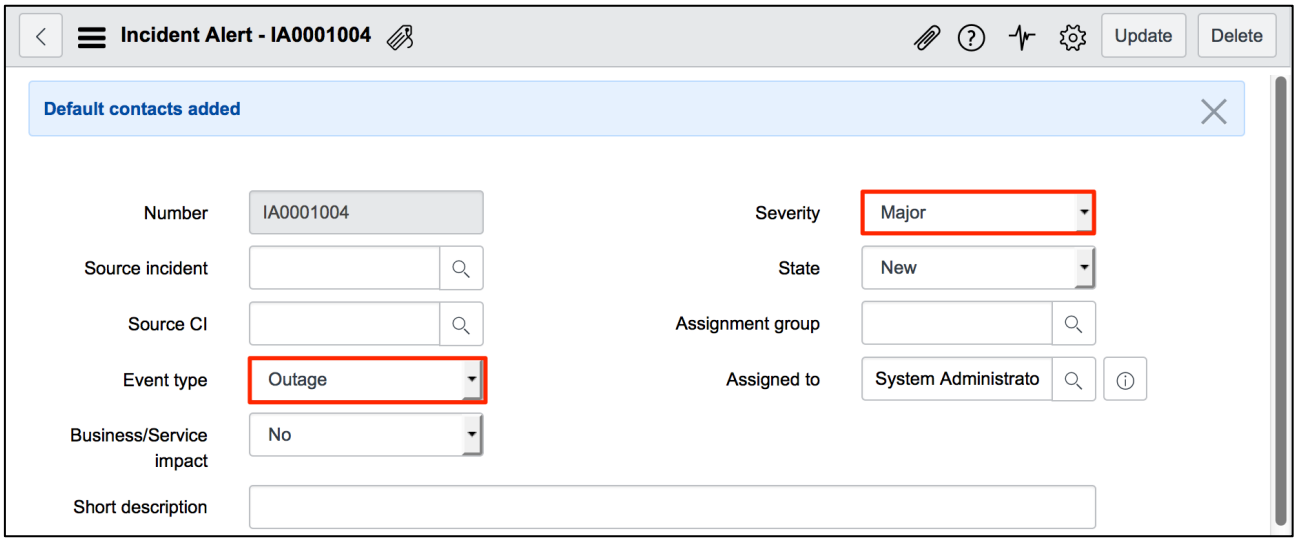
13. Leave all the default selections as they are, but enter some text in the required **Description** field. Click **Additional actions** and select **Save** to remain on the record.

    Note: Once a new Incident Alert is created you should receive an SMS message.

14. Scroll to the User Contacts section (click **User Contacts** tab) and notice that only the Duty Manager, Duty Director, and Incident Manager were added to this new Incident Alert.

©2015 ServiceNow, Inc. All rights reserved.

15. Scroll back to the top and set the Event type to **Outage** and the Severity to **Major**. Click **Additional actions** and select **Save** so to remain on this record.

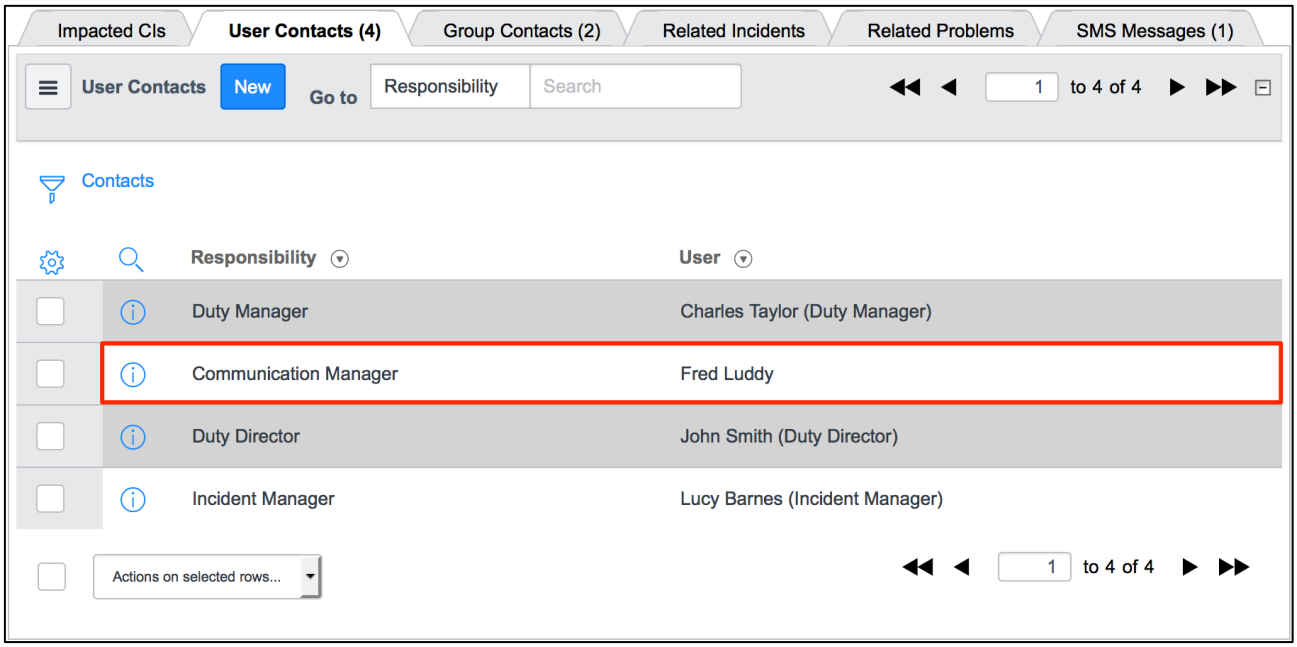


16. Scroll to the User Contacts section (click **User Contacts** tab) and notice that Fred Luddy, the Communications Manager, was added as a User Contact for this Incident Alert.

®2015 ServiceNow, Inc. All rights reserved.

Lab Success Verification

1. A new contact definition was added that successfully adds a Communication Manager to Major Severity Outage Incident Alerts.



2. Navigate to **Lab Management > Report Lab Progress. Click I am done!**

©2015 ServiceNow, Inc. All rights reserved.

Lab Goal

This lab explains how to initiate and manage a conference call.

Incident Alert teamed up with Notify is a powerful combination. So far we have seen automated SMS messages on new Incident Alerts. Conference calls can also be initiated to bring the proper team together to discuss and resolve major issues.

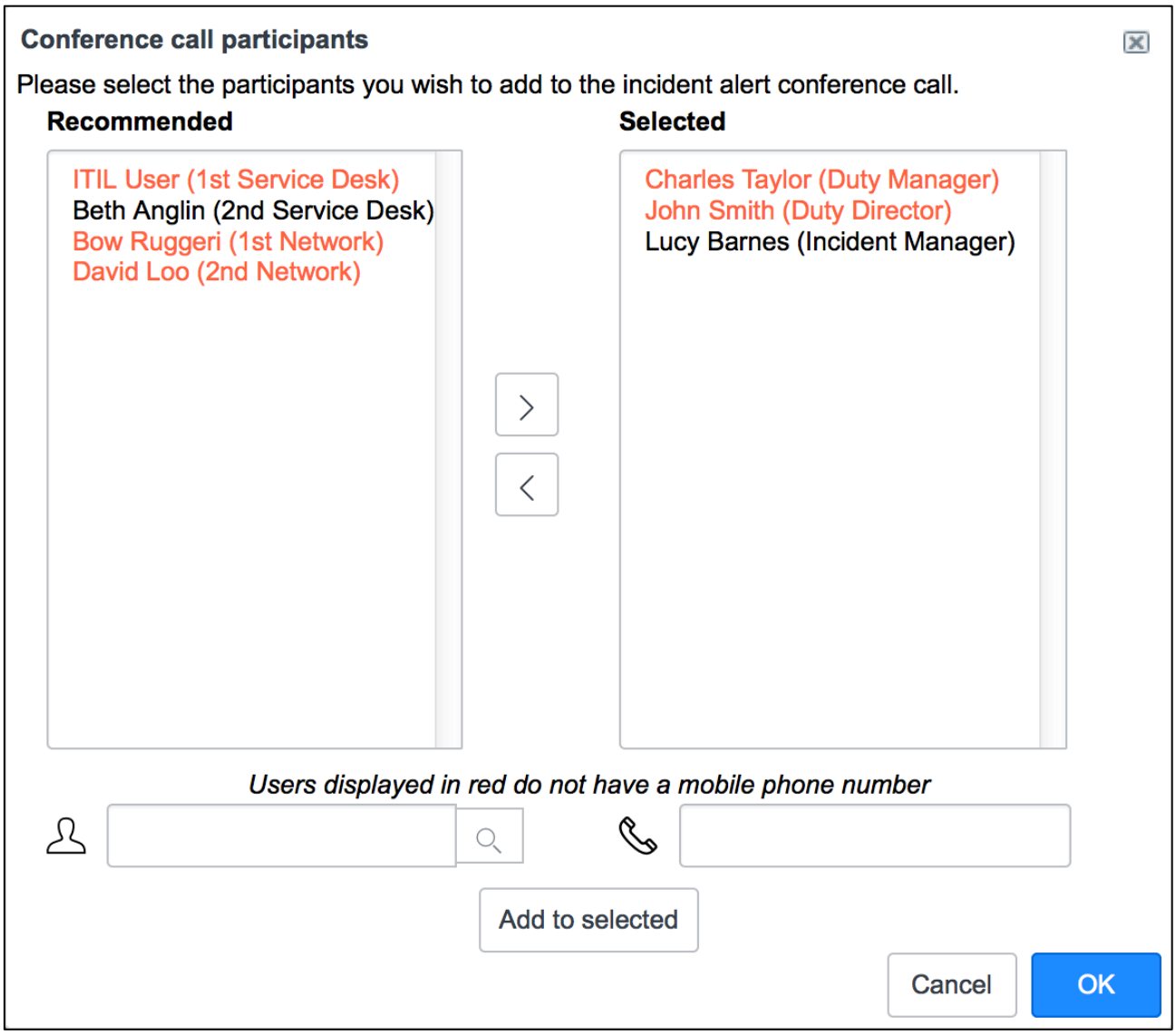
Lab 6
Initiate and Manage Conference Call

Initiate a Conference Call from an Existing Incident Alert

1. In the Filter Text box, type **alert**.
2. Click **Open** under the Incident Alert Management application.
3. In the list on the right, click **IA0001003**, or the lowest numbered Incident Alert created initially for the Exchange server outage.
4. Scroll toward the bottom to the Related Links above the Impacted CI list. Click **Initiate Conference Call**.

®2015 ServiceNow, Inc. All rights reserved.

5. A dialog appears similar to the following.



6. Notice that most of the users show up in **Red**. This means that these users do not have a valid phone number in their user record. Lucy Barnes shows up in **Black** meaning she has a valid phone number (updated in an earlier exercise). Feel free to update another user with another phone number.

7. Notice the list of User Contacts from the Incident Alert that were either automatically added from contact definitions or those manually added automatically show up on the **Selected** list (right side).

©2015 ServiceNow, Inc. All rights reserved.

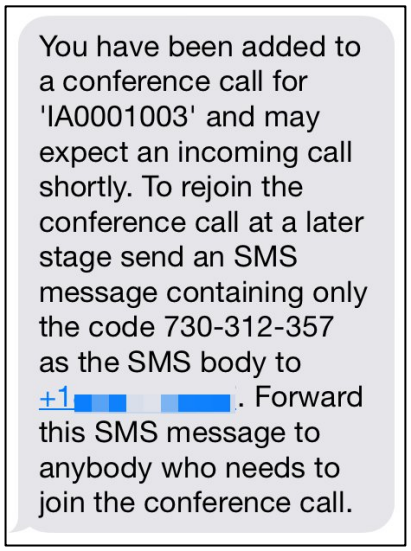
8. Notice on that the left side is a list of **Recommended** contacts. This is populated by the On-call members (1st, 2nd, and 3rd) of the Group Contacts from the Incident Alert. Recall the groups automatically added by contact definitions were the Service Desk and the Network team. To include any of these members in the conference call, double-click them to move them to the **Selected** list (right side).

9. At the bottom, there are options to add other users or specific phone numbers to the conference call.

10. Since most of us only have one phone, proceed with the recommended contacts. Click **OK** at the bottom. At the top of the Incident Alert several messages appear, letting you know that some users do not have a "voice capable phone number configured". You can ignore these messages.

11. Momentarily, a call comes to your mobile phone. Notice that Twilio has text to speech capabilities reading you some details about the Incident Alert. When prompted, click **1** to accept the call. Please remain on the call for the next few steps.
Note: Twilio plays hold music until more than one participant joins the call.

12. You also receive an SMS message to your phone, confirming your invitation to the conference call with details on how to join back or invite other people.

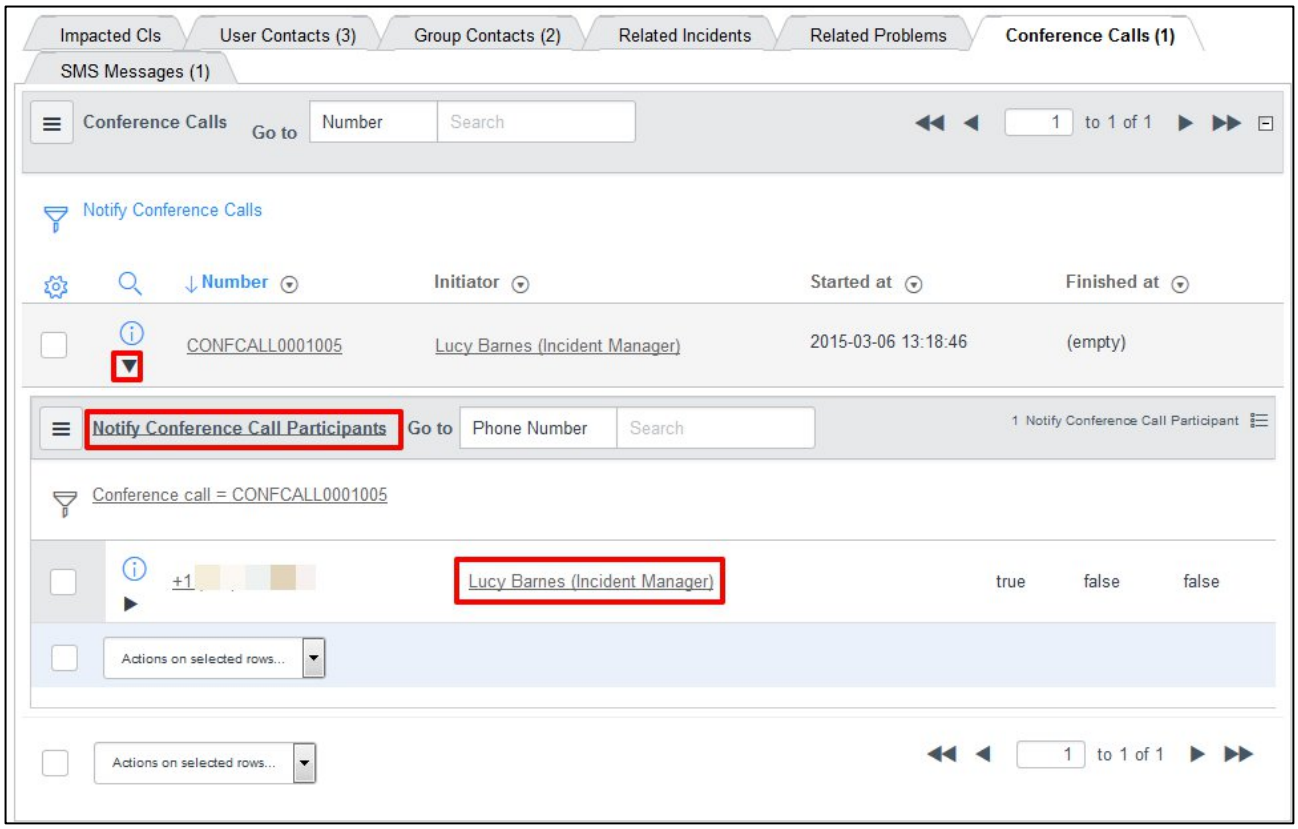


Note: Users can also call the Twilio phone number provided and enter the code to join the call.

13. New with the Fuji release is the ability to manage the conference call participants from ServiceNow. You have the ability to **mute**, **unmute**, and **kick** a participant from a call.

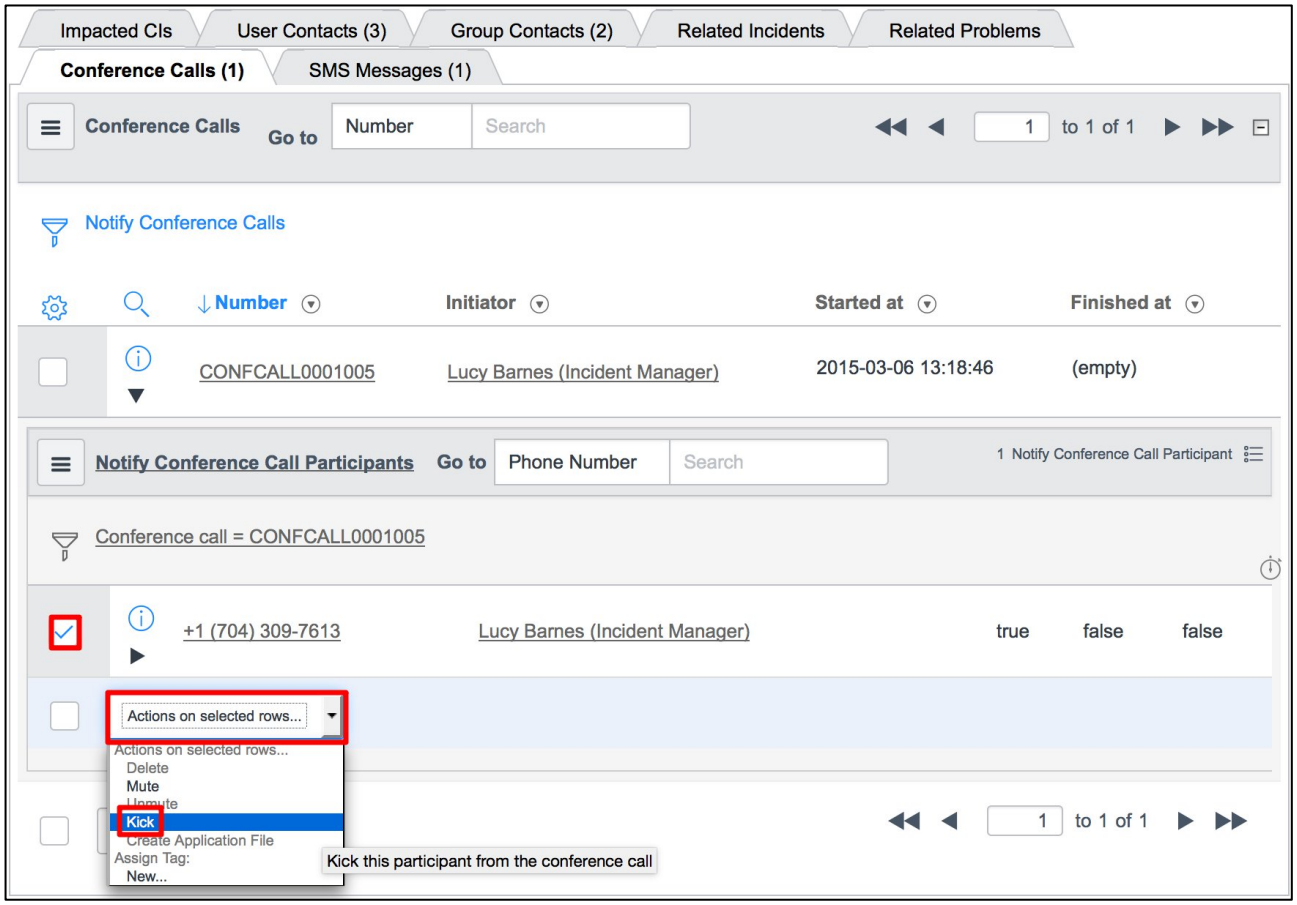
©2015 ServiceNow, Inc. All rights reserved.

14. **You are still on the Conference call**. Scroll to the bottom of the Incident Alert form to the Conference Calls section (click **Conference Calls** tab). Notice a conference call record is created to reflect that a conference call was initiated from this Incident Alert. Expand the hierarchal list to view the participants on the conference bridge.



©2015 ServiceNow, Inc. All rights reserved.

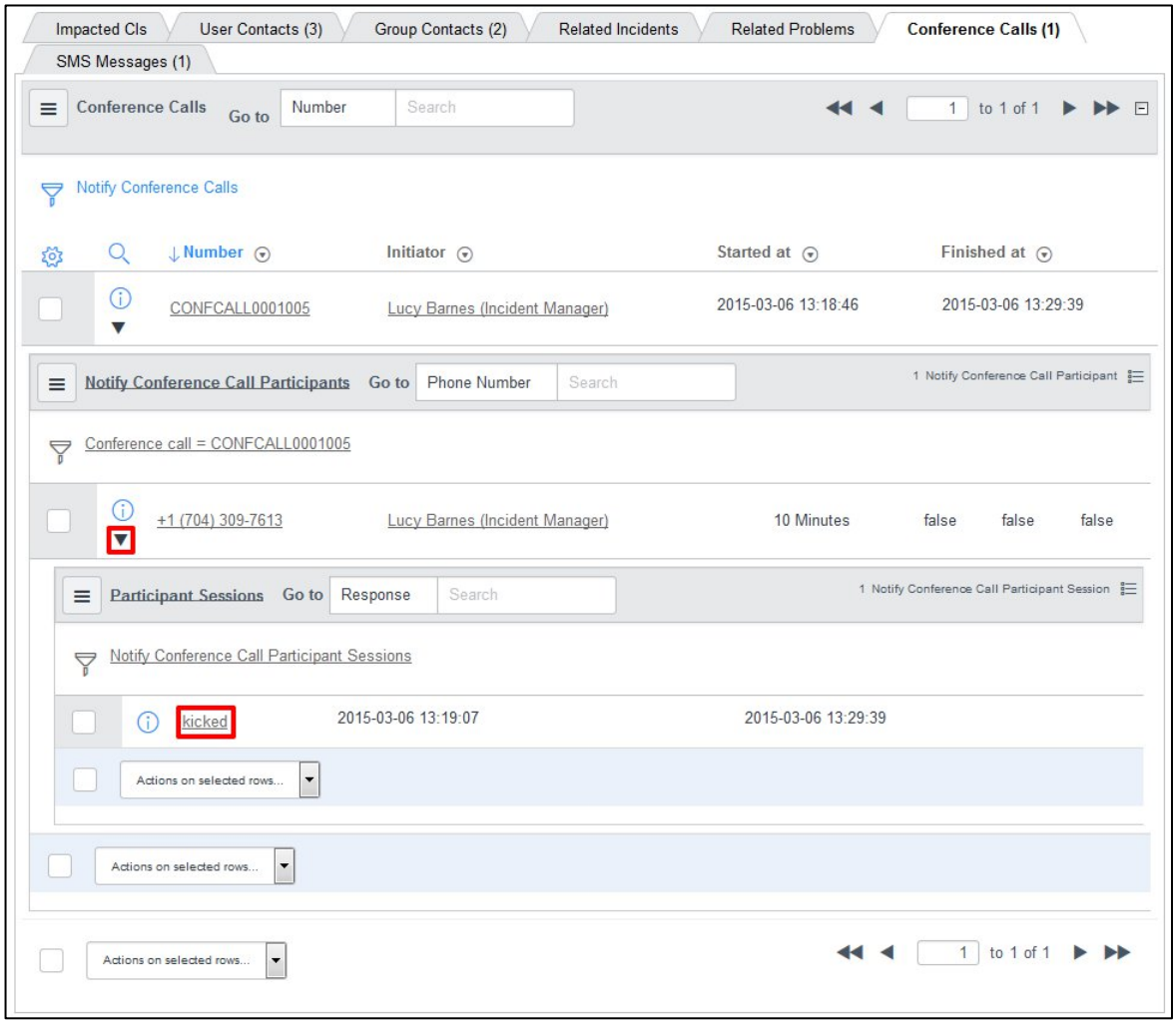
15. This list contains options to **mute/unmute** or **kick** the participant from the call. Click the **check box** on the row with Lucy Barnes and click the **Actions on selected rows** dropdown list. Choose **Kick**.



16. Your phone should now be disconnected from the conference call.

©2015 ServiceNow, Inc. All rights reserved.

17. Expand the **Call Participants** hierarchal list to see that you were kicked from the conference call. Notice the other details such as the duration of the conference call and when participants joined and left.



Lab Success Verification

1. A conference call was successfully initiated from an Incident Alert.
2. You removed a participant from the conference call.
3. Navigate to **Lab Management > Report Lab Progress. Click I am done!**

©2015 ServiceNow, Inc. All rights reserved.

Lab Goal

This lab explains how to resolve an Incident Alert and capture **Post Incident Review** (PIR) details.

Lab 7 Resolve Incident Alert

Resolve an Incident Alert to Show Integration to Problem Management

1. In the Filter Text box, type **alert**.
2. Click **Open** under the Incident Alert Management application.
3. In the list on the right, click **IA0001003**, or the lowest numbered Incident Alert created initially for the Exchange server outage.
4. At the top, change the State to **Resolved**. Click **Additional actions** and select **save** to remain on this record.
5. Notice a new section called Post Incident Review (**Post Implementation Review** tab) appears. Scroll to this section or click the Post Incident Review tab.
6. Notice details about the source incident appear on this tab. Here you can enter details such as resolution notes, summary, and lessons learned.
7. Set the **Resolution Code** to **Complete**, if not set already by default.
8. Enter the following information:

   Resolution notes: **Hardware team restarted the Exchange server and service was restored. Microsoft was engaged and suggested applying a patch.**
   Summary: **Exchange server blue screened.**
   Lessons learned: **Microsoft suggested we keep our servers up to date.**

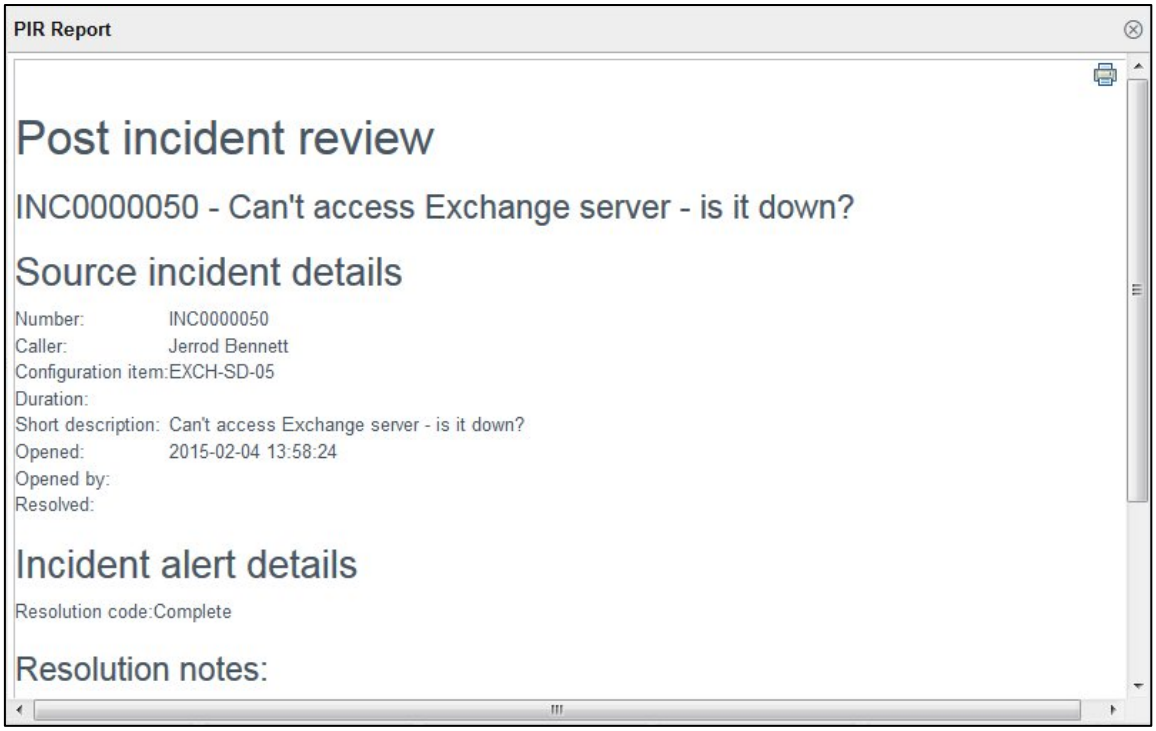
9. Click **Additional actions** and select **Save** to remain on this record.

®2015 ServiceNow, Inc. All rights reserved.

10. Click **View PIR Report** under Related Links. This report can be printed to provide "gun-shot" report of what happened, what we did, what to do next. It is also useful information for a formal Problem or Root Cause Analysis record.

Lab Success Verification

1. The Incident Alert opened previously is now resolved. We are able to view the Post incident review report.



2. Navigate to **Lab Management > Report Lab Progress. Click I am done!**

®2015 ServiceNow, Inc. All rights reserved.

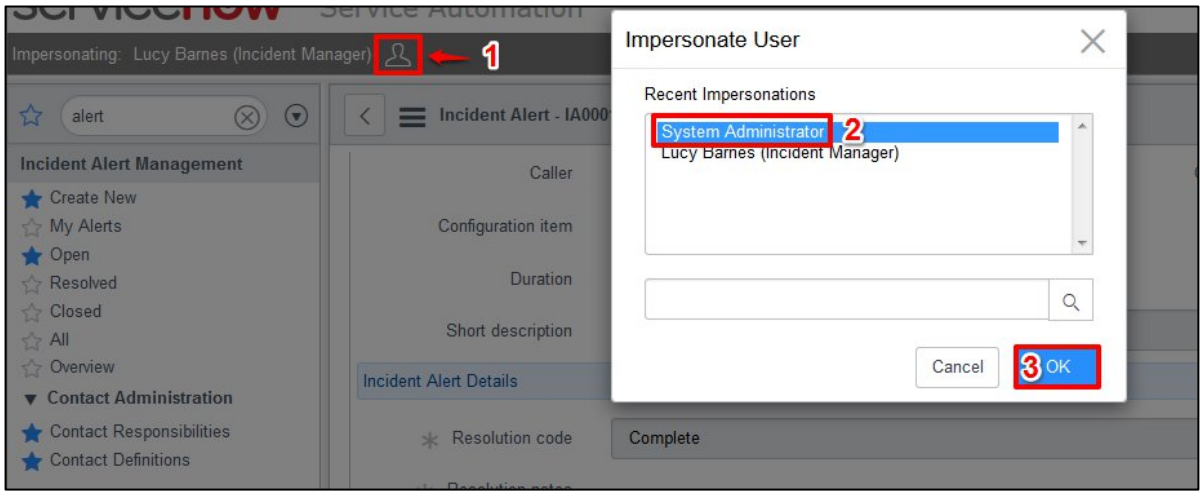
Lab Goal

This lab explains how to create a new On-Call Rotation (Rota) and trigger rule. The Eureka release included an updated Rota On-Call Scheduling (Rota) application. This was integrated with Notify with bidirectional SMS capabilities.

Lab 8
Create On-Call Rota

Create New On-Call Rota

1. Log on as System Administrator by clicking **Impersonate User**.



2. After the Impersonate User dialog appears, select **System Administrator** in the Recent Impersonations list.

3. Click **OK**.

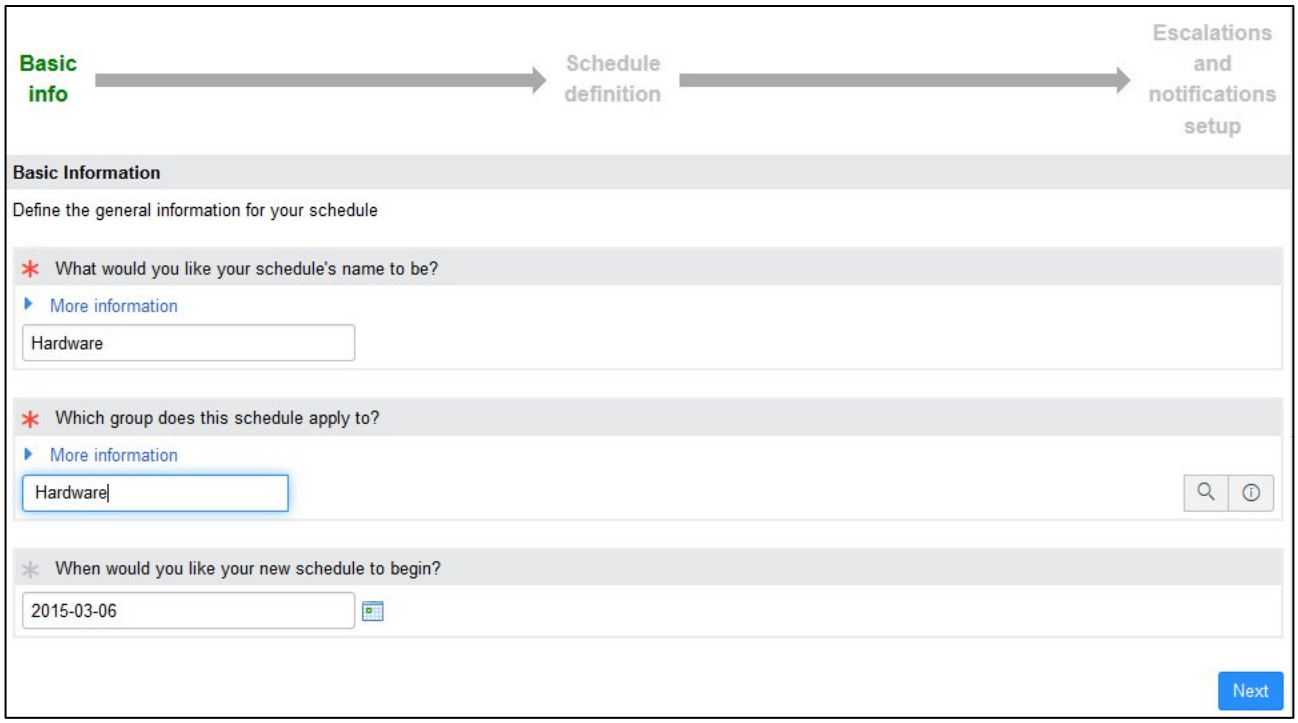
4. In the Filter Text box, type **on-call**.

5. Click **Create New Schedule** under Administration.

©2015 ServiceNow, Inc. All rights reserved.

6. Enter the following information:

   What would you like your schedule's name to be?: **Hardware**
   Which group does this schedule apply to?: **Hardware**



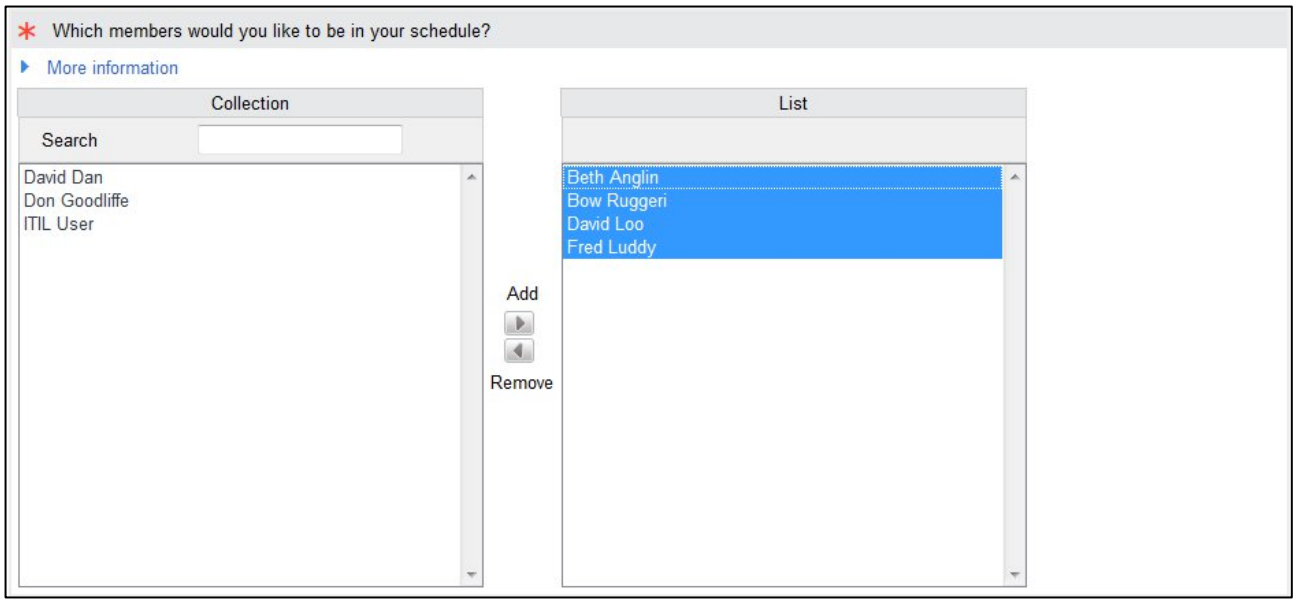
7. Click **Next**.

8. Choose **Yes** to use an existing schedule and choose **24 x 7** for the schedule, if not set by default. Then click **Next**.

9. Choose **1** for the first question on the number of duty rotations.

©2015 ServiceNow, Inc. All rights reserved.

10. Notice you are able to choose which members of the Hardware team are on the on-call rotation, versus the entire group. Select the following members for the schedule: Beth Anglin, Bow Ruggeri, David Loo, and Fred Luddy. Double-click each member to move them from **Collection** (left side) to **List** (right side).



Note: Please confirm that Beth Anglin is the first person in the list.

11. Choose **1** for the number of reminders.

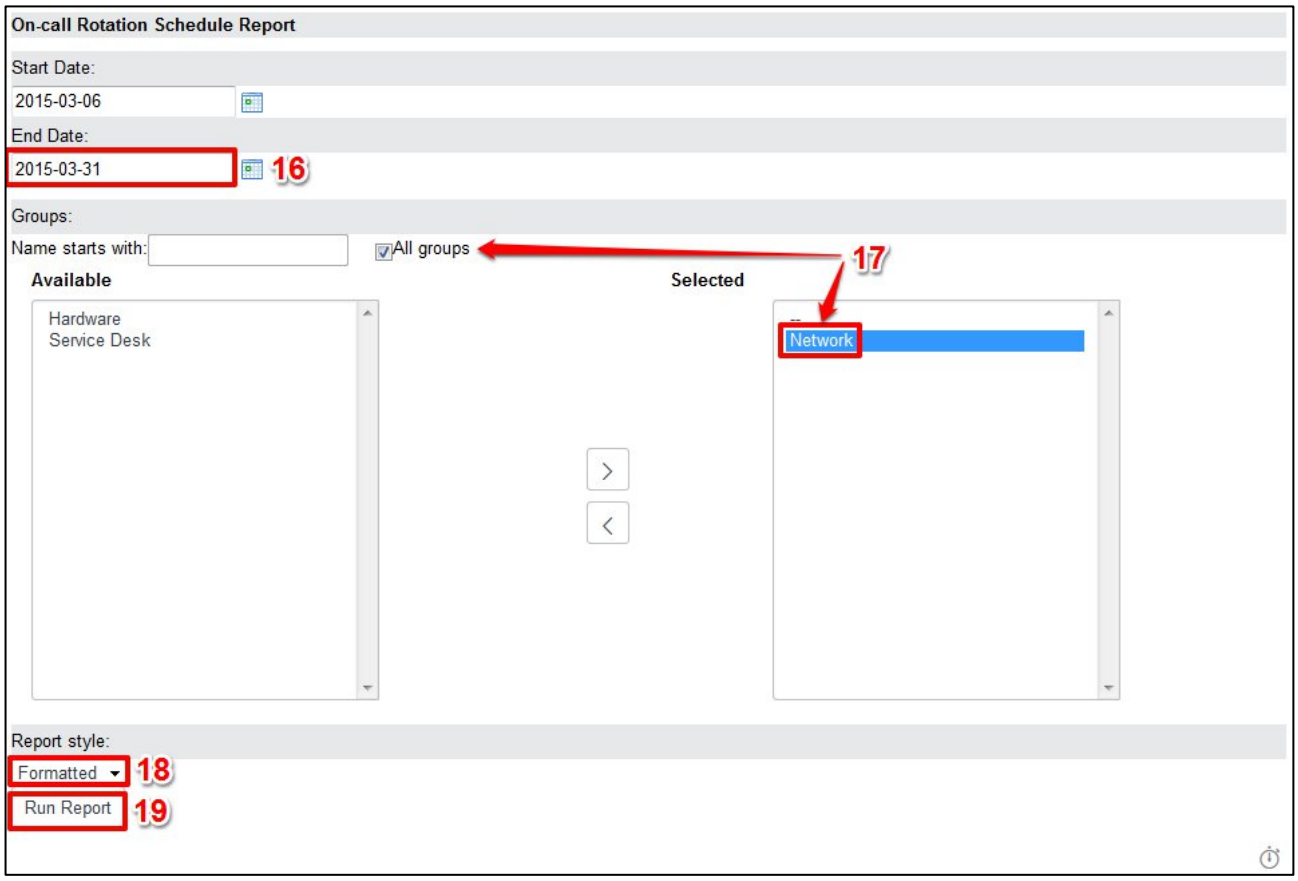
12. Choose **15** minutes for the wait until sending the next reminder.

13. Click **Next**.

14. You should now see a calendar view of the Rota. Notice that Beth is on-call starting today for the next week. Bow Ruggeri is on call after that.

©2015 ServiceNow, Inc. All rights reserved.

15. There is also another view of the schedule. In the On-Call scheduling application, click **Schedule Report**.



16. Leave the **Start Date** the default of **Today's Date**, but change the **End Date** to the **end of the month.**

17. Click the **All groups** check box. Double-click on **Network** to move it to the **Selected** list.

18. There are a few options for report style. Choose **Formatted**.

19. Then click **Run Report** at the bottom.

20. Notice on the formatted report that there are multiple schedules for the Network: EMEA, Americas, and APAC. From this view you can see who is on-call and between what hours. Notice the Network (Americas) has a Primary and Secondary schedule. This view can also be printed.

©2015 ServiceNow, Inc. All rights reserved.

21. Feel free to repeat Steps **15-20** to view the **Table** view of this report or look at another Rota.

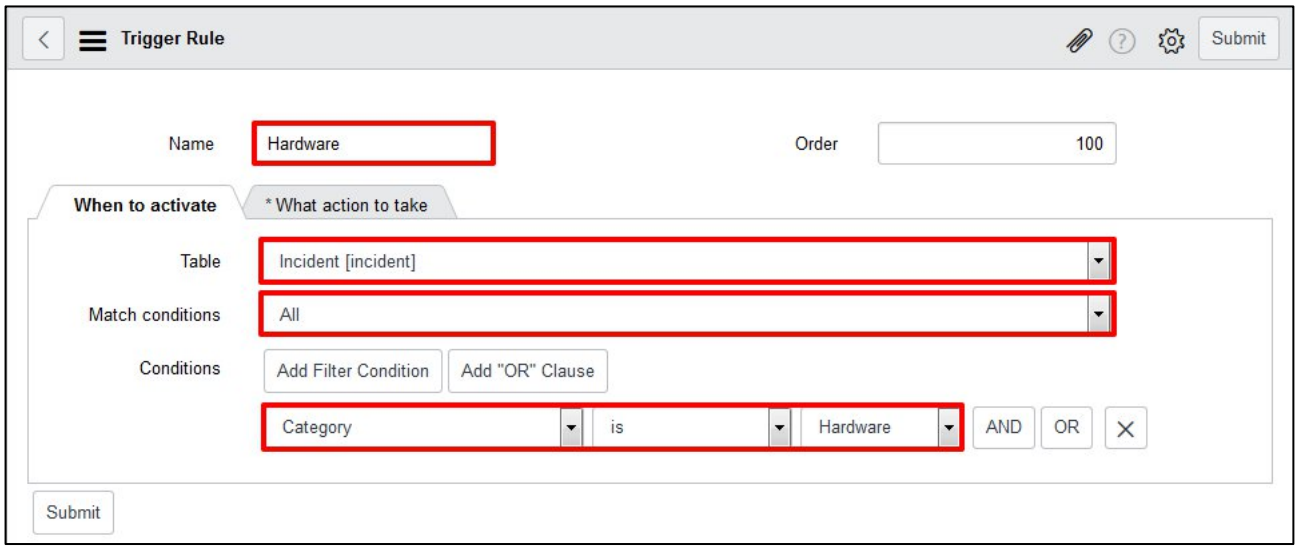
22. Now create a trigger rule to assign the Hardware team to a task and start the On-Call workflow.

23. In the On-Call scheduling application, click **Trigger Rules**.

24. Click the **New** button at the top of the list to create a new rule.

25. Enter the following information:

    Name: **Hardware**
    Table: **Incident**
    Match conditions: **All**
    Conditions: **Category is Hardware**

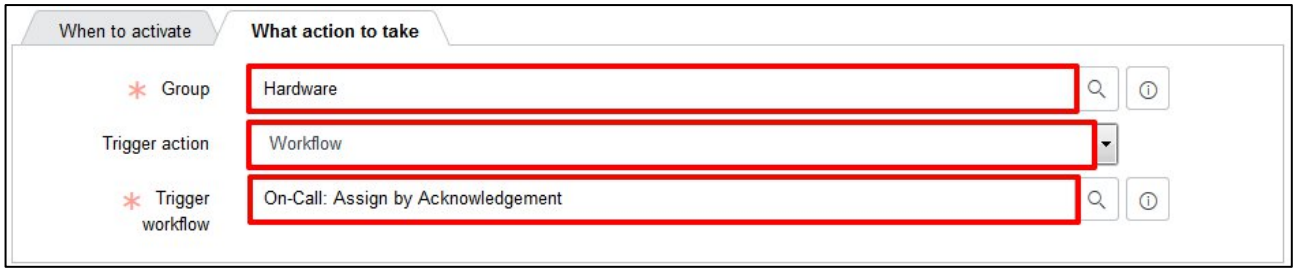


26. Go to the **What action to take** section or click the **What action to take** tab.

©2015 ServiceNow, Inc. All rights reserved.

27. Enter the following information:

    Group: **Hardware**
    Trigger action: **Workflow**
    Trigger workflow: **On-Call: Assign by Acknowledgement**



28. Click **Submit**.

Lab Success Verification

1. A new Rota and Trigger Rule for the Hardware Team has been created.

2. Navigate to **Lab Management > Report Lab Progress. Click I am done!**

©2015 ServiceNow, Inc. All rights reserved.

Lab Goal

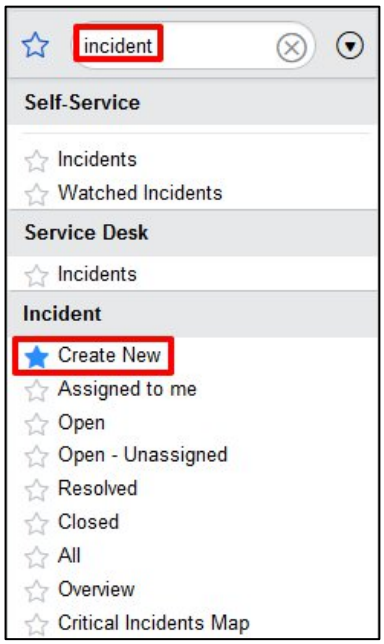
This lab explains how to test a Hardware Rota.

Now that we have created a Hardware Rota and Trigger Rule, any new incident in which the category is Hardware automatically is assigned to the Hardware team and a workflow starts to assign to the On-Call person through a verification process.

Lab 9
Test Hardware Rota

Create Incident to Test Hardware Rota

1. In the Filter Text box, type **incident**.
2. Click **Create New** under the Incident application.



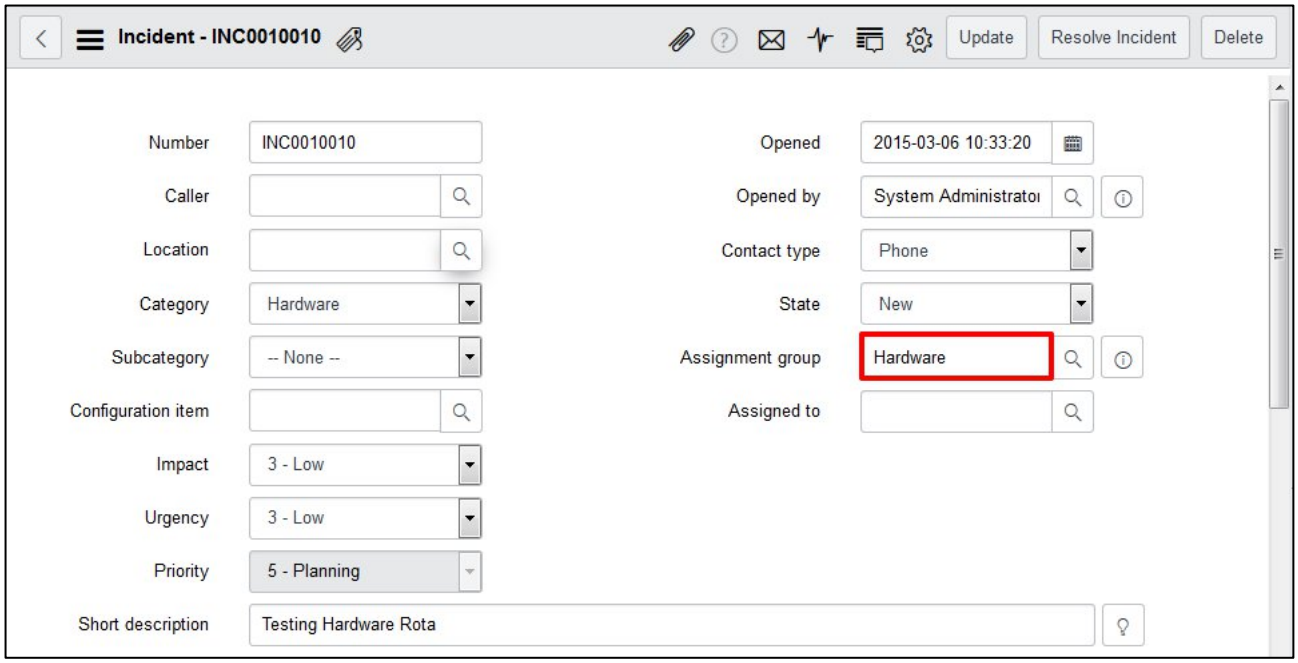
3. Change the **Category** to **Hardware** and enter **Testing Hardware Rota** in the **Short Description** field. Note that the **Assignment** group is currently empty as well as the **Assigned to** person.

©2015 ServiceNow, Inc. All rights reserved.

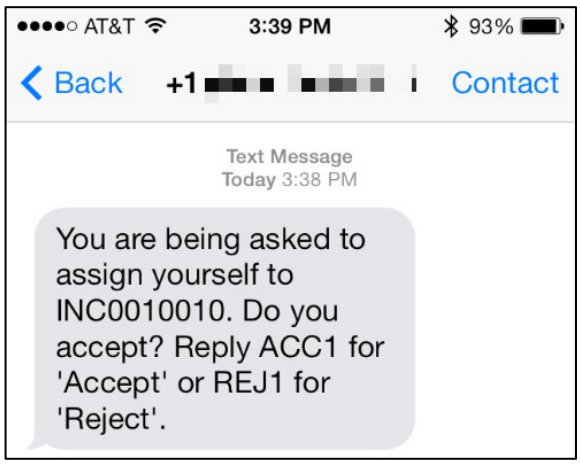
4. Click **Additional actions** and select **Save** to remain on this record.

5. Note the Incident number, which is used in later steps.

6. After the record has been saved, the Assignment group should automatically be populated with **Hardware**. This is because of the trigger rule we created.



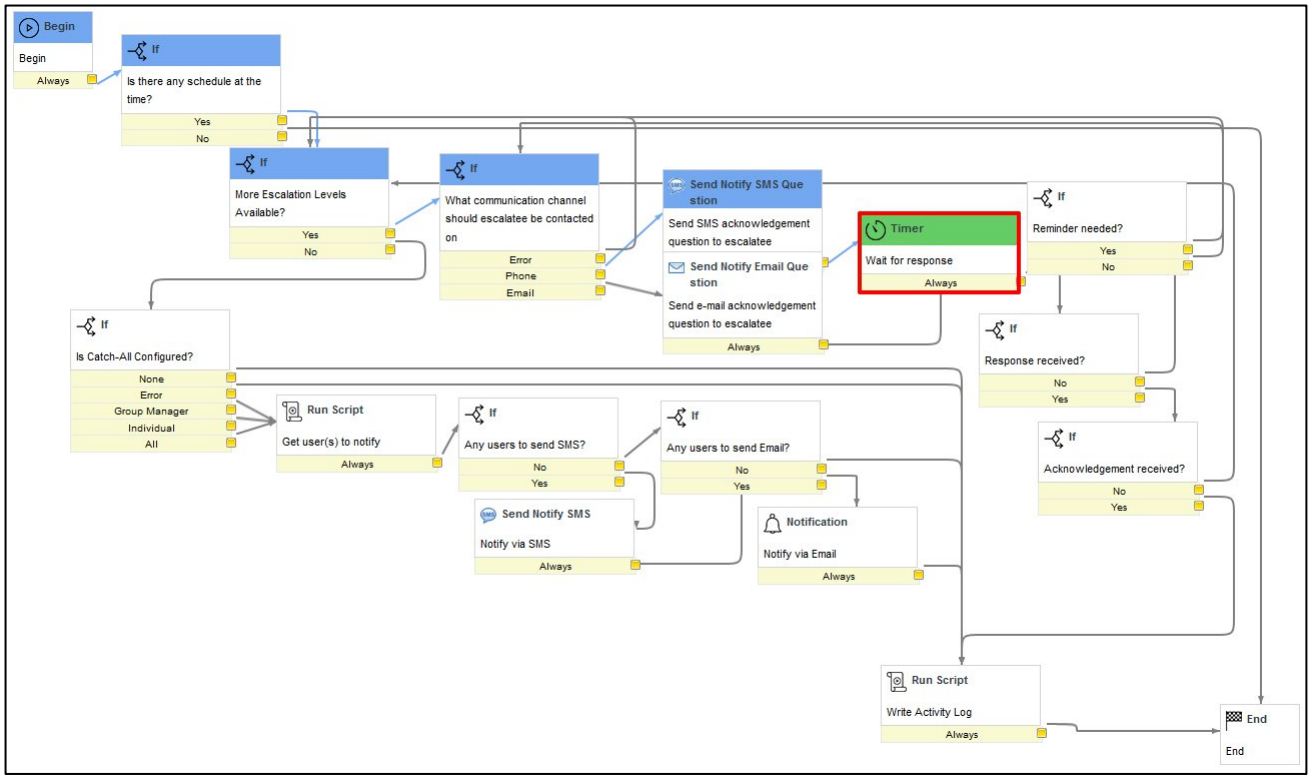
7. Shortly you should receive a Text on your Mobile Phone asking you to accept or reject the assignment of this new incident.



©2015 ServiceNow, Inc. All rights reserved.

8. Before you respond to this text, view the workflow. Scroll down to Related Links. Click **Show Workflow**.

9. This is an out-of-the-box workflow that comes with the new version of On-Call Rotation. Notice that the workflow is waiting for a response from the SMS that was generated.



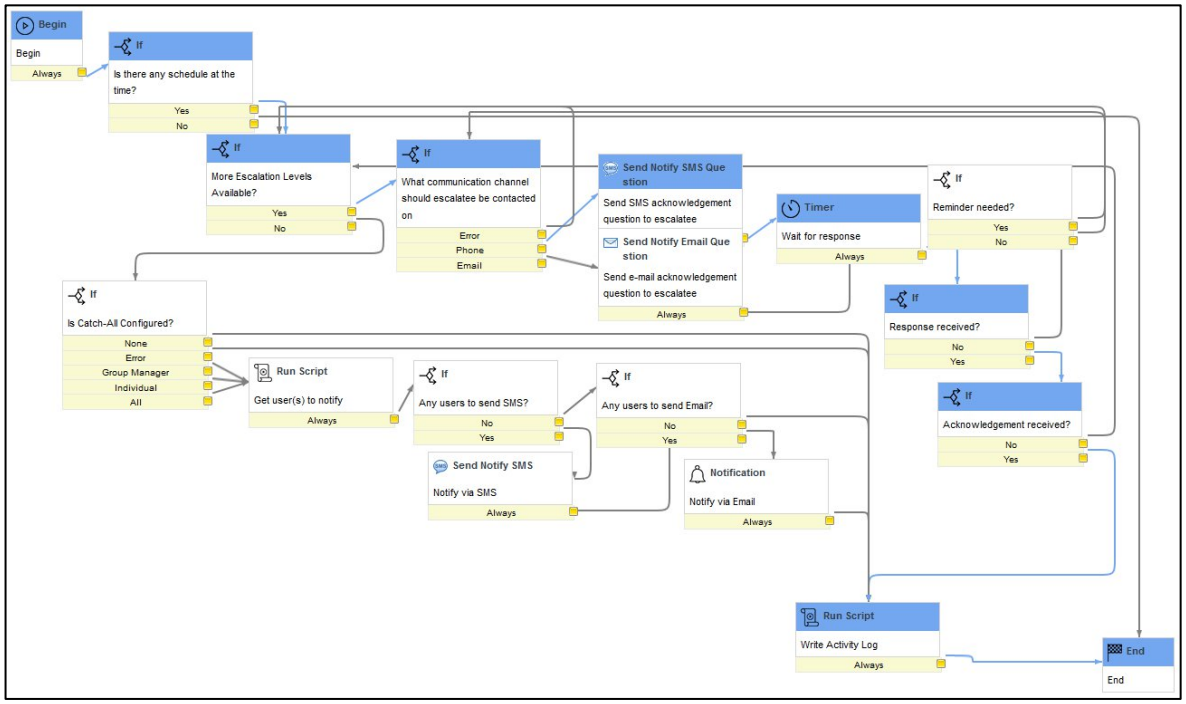
10. Now respond back to the SMS that you received earlier. Reply back with **ACC1** to accept the assignment.

11. Return to ServiceNow and click the **Refresh** button in the upper-right corner.



®2015 ServiceNow, Inc. All rights reserved.

12. Notice that after you accept the assignment of the Incident, the workflow is complete.



13. Go back to the Incident. In the Filter Text box, type **incident**.

14. Click **Open** under the Incident application. In the list to the right, open the incident previously created. The number will be in the SMS you received.

15. Notice that Beth Anglin is now assigned to the incident. If you recall, you added your phone number to her user record earlier.

Lab Success Verification

1. Incidents of category Hardware are automatically assigned to the Hardware Group.

2. The On-Call person receives an SMS message to accept or reject the assignment.

3. Replying to the SMS updates the Incident.

4. Navigate to **Lab Management > Report Lab Progress. Click I am done!**

©2015 ServiceNow, Inc. All rights reserved.

Lab Goal

This advanced lab presents an alternative to using On-Call trigger rules.

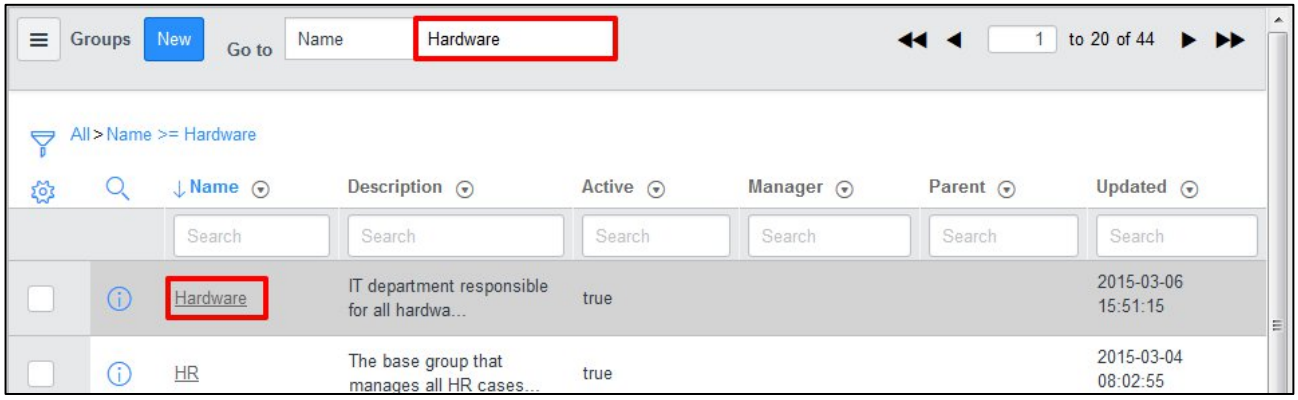
Although trigger rules are useful, they do not work with all use cases. Some customers may prefer to set the Assignment Group based on the Support Group attribute on the Configuration Item. Once the incident is assigned, then the on-call workflow should trigger.

Lab 10
Advanced Trigger Rule Alternative

Log in as System Administrator

A feature of the trigger rules is allowing each group to choose their own escalation workflow. To mimic this same feature, we add a **Workflow** field on the **Group** table.

1. In the Filter Text box, type **system security**.
2. Under the Users and Groups submenu, click **Groups**.
3. Search for the Hardware Group. Click **Hardware** in the list to pull up the group form.

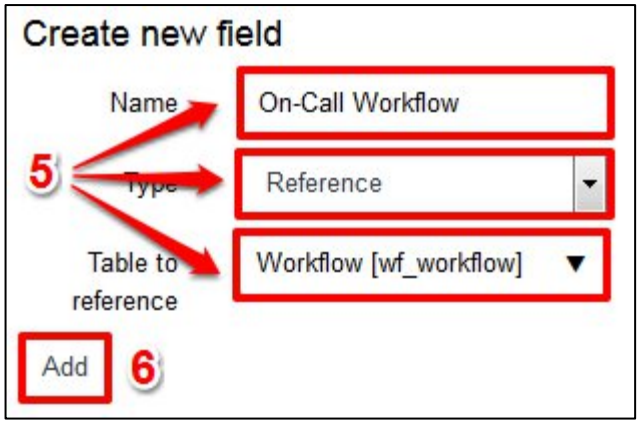


4. Click **Additional** actions and select **Configure.** Select **Form Layout**.

©2015 ServiceNow, Inc. All rights reserved.

5. In the **Create new field** section in the bottom-right corner, enter the following information:

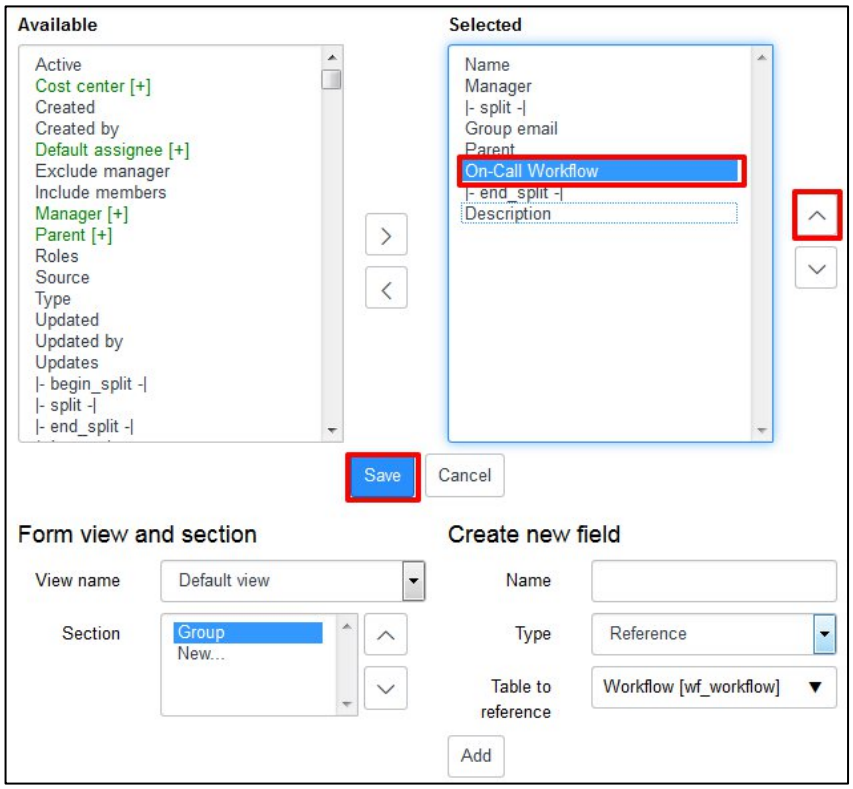
   Name: **On-Call Workflow**
   Type: **Choose Reference**
   Table to reference: **Workflow [wf_workflow]**



6. Click **Add**.

®2015 ServiceNow, Inc. All rights reserved.

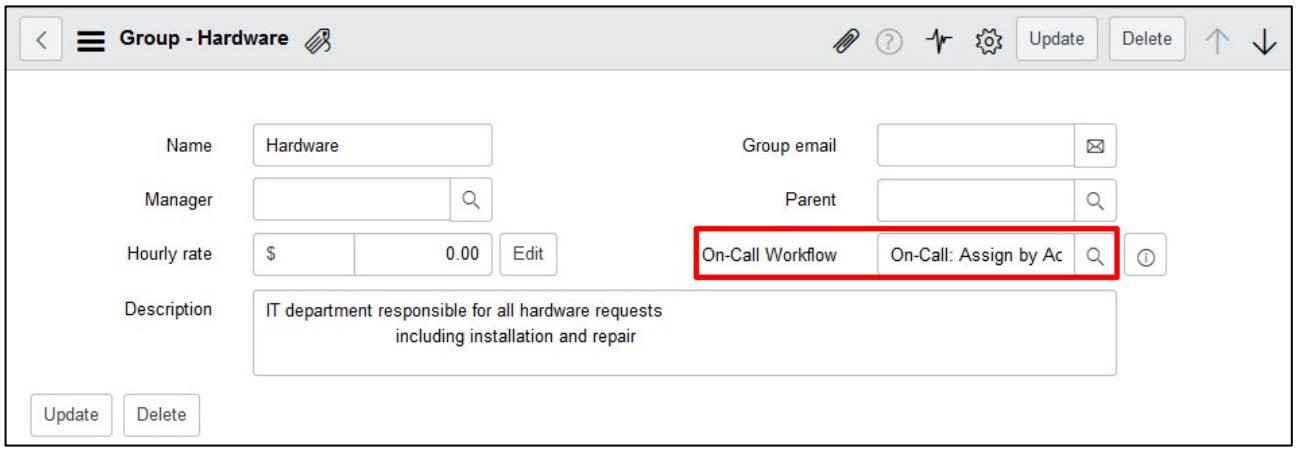
7. The new On-Call Workflow field appears at the bottom of the Selected Column. Click it and then click the **up** arrow to move it under the Parent Field.



8. Click **Save** to add the new field to the database. Click **Close** on the next dialog to show the updated Group form.

®2015 ServiceNow, Inc. All rights reserved.

9. In the new On-Call Workflow field, type or select **On-Call: Assign by Acknowledgement**. Your form should look like the example.



10. Click **Update** to save the On-Call Workflow setting.

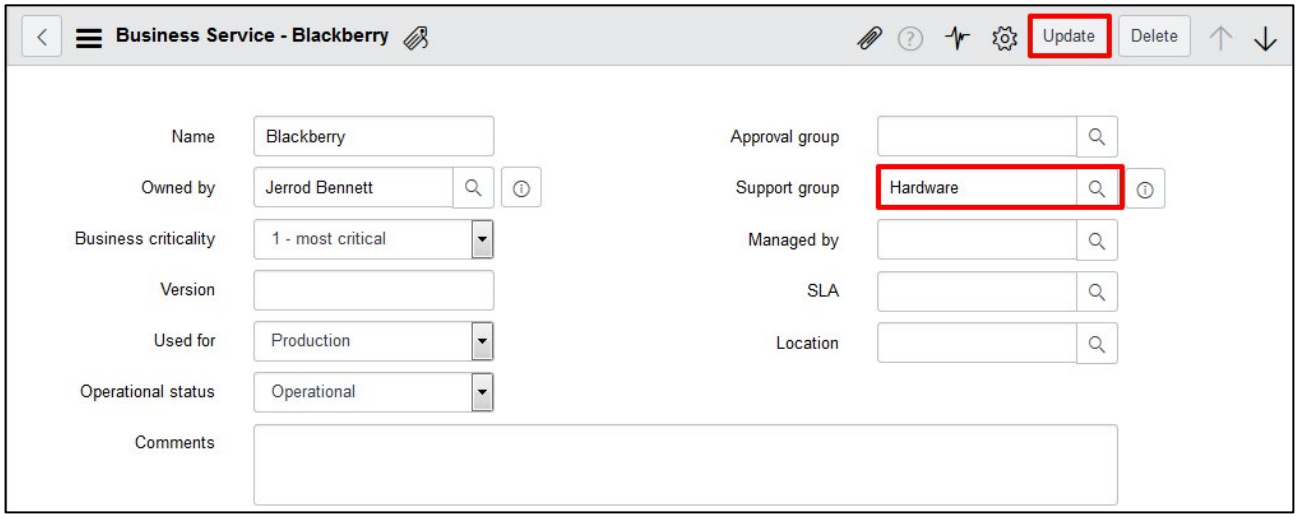
11. Now set the support group of a CI to **Hardware** to utilize the on-call schedule.

12. In the Filter Text box, type **config**.

13. At the top of the Configuration application, **click Business Services**.

14. In the list click **Blackberry**.

15. Type **Hardware** into the **Support group** field. Click **Update**.



©2015 ServiceNow, Inc. All rights reserved.

16. Now delete the trigger rule that we created previously; we utilize another method to set the Incident Assignment Group.

17. In the Filter Text box, type **trigger**.

18. In the On-Call Scheduling application click **Trigger Rules**.

19. In the Trigger Rules list, click **Hardware** to pull up the record created earlier.

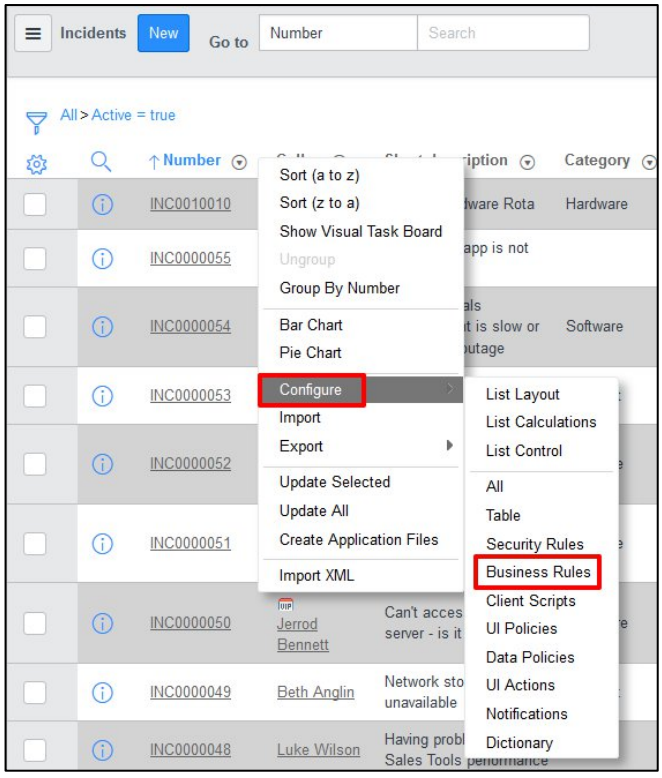
20. After the form loads, click **Delete**. Click **Delete** again on the confirmation popup.

21. Next create a business rule on the Incident table to set the incident's assignment group to the Support Group value from the CI and then trigger an event that will execute the On-Call Workflow from the group record.

22. In the Filter Text box, type **incident**.

23. Click **Open** under the Incident Application.

24. On the incident list, **right-click** the column header and choose **Configure > Business Rules**.



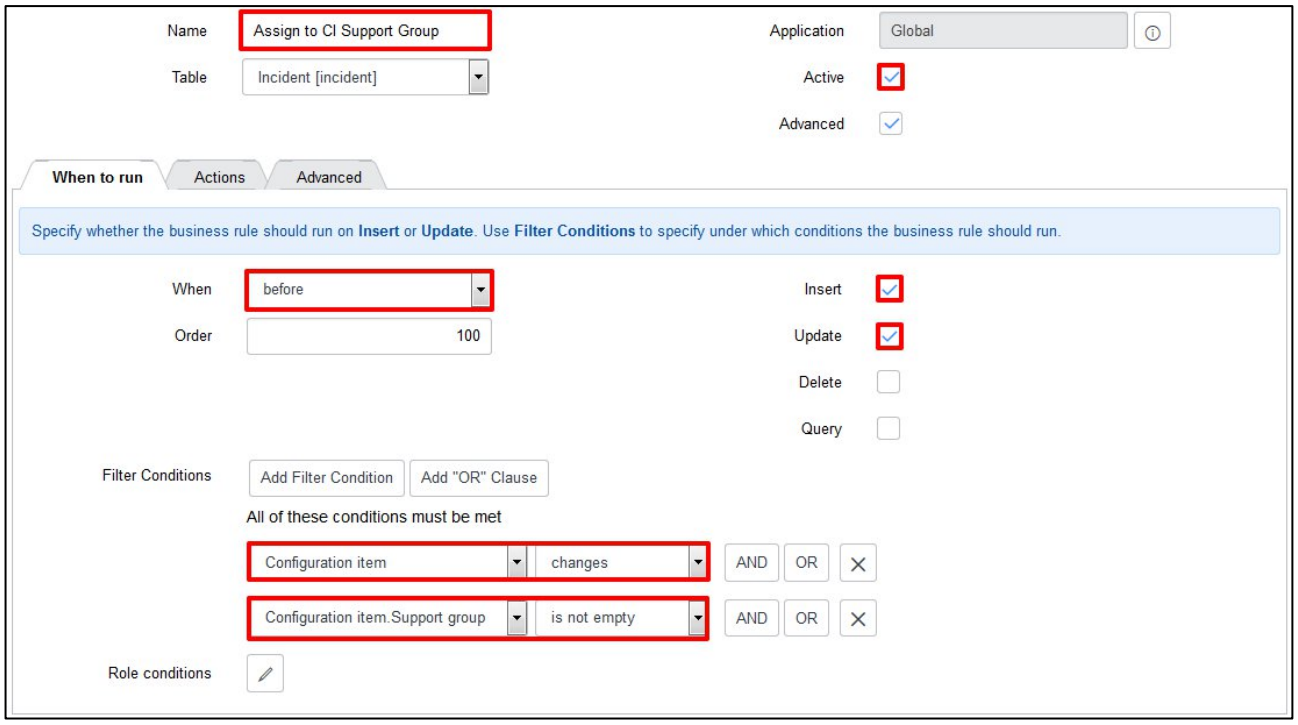
©2015 ServiceNow, Inc. All rights reserved.

25. On the Business Rules List, click **New**.

26. Enter the following Information on the initial screen and **When to run** section:

    Name: **Assign to CI Support Group**
    Advanced: **Checked**
    Insert: **Checked**
    Update: **Checked**
    When: **before**
    Filter Conditions: **Configuration item changes and Configuration item.Support group is not empty**

    Note: In order to filter on the Configuration Item's Support Group, you may need to scroll down to the bottom of the list and select **Show Related Fields**. Once that is done you will select **Configuration item ➔ Configuration Item fields** and then **Support group**.



®2015 ServiceNow, Inc. All rights reserved.

27. Go to the Advanced section or click the **Advanced** tab and enter the following script.

    Note: The steps below walk you through creating this script with explanation. The entire script can also be copied from the Lab Management application under the Lab Exercises section by clicking **Lab 10 – Business Rule Script**.

    Script:

```
function onBefore(current, previous) {
    //Set the assignment group to the CI's Support Group Value
    current.assignment_group = current.cmdb_ci.support_group;

    //Check to see if the group has an on-call workflow set and if so trigger
an event to execute the workflow
    if (JSUtil.notNil(current.assignment_group.u_on_call_workflow)) {
        gs.eventQueue("on_call.process", current, "incident",
current.sys_id);
    }
}
```

28. Click **Submit**. If you receive any errors saving the record please check your script and compare it to the Lab 10 Business Rule Script that you can access in the Lab Management application.

29. Next, this new event needs to be registered by creating an Event Registry entry for “on_call.process”. In the Filter Text box, type **events**.

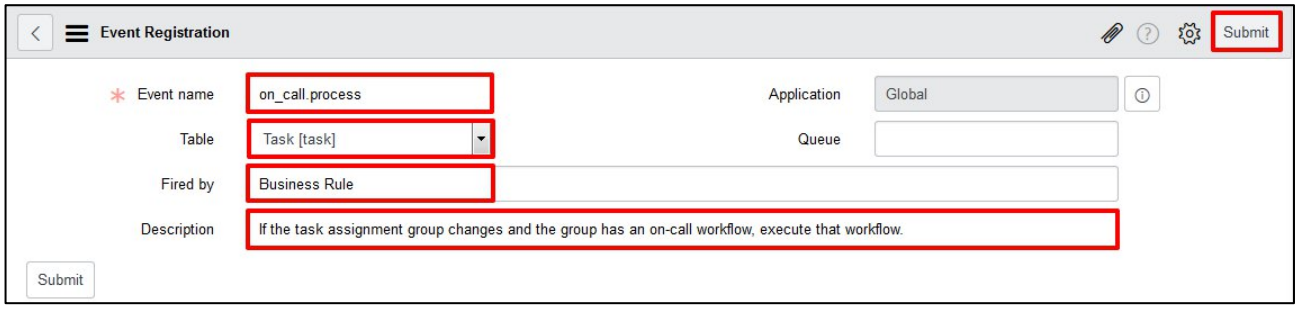
30. In the System Policy application click **Registry**.

31. Click the **New** button at the top of the list to create an event registry.

©2015 ServiceNow, Inc. All rights reserved.

32. Enter the following information:

    Event name: **on_call.process**
    Table: **Task [task]**
    Fired by: **Business Rule**
    Description: **If the task assignment group changes and the group has an on-call workflow, execute that workflow.**



33. Click **Submit**.

34. Next, a Script Action needs to be created that will execute the workflow once the on_call.process event is fired. Click **Script Actions** in the System Policy Application.

35. Click the **New** button at the top of the list to create a script action.

    Enter the following information:

    Name: **Process On-Call Assignment**
    Event name: **on_call.process**
    Active: **checked**

    Note: The steps below walk you through creating this script with explanation. The entire script can also be copied from the Lab Management application under the Lab Exercises section by clicking **Lab 10 – Script Action Script**.

©2015 ServiceNow, Inc. All rights reserved.

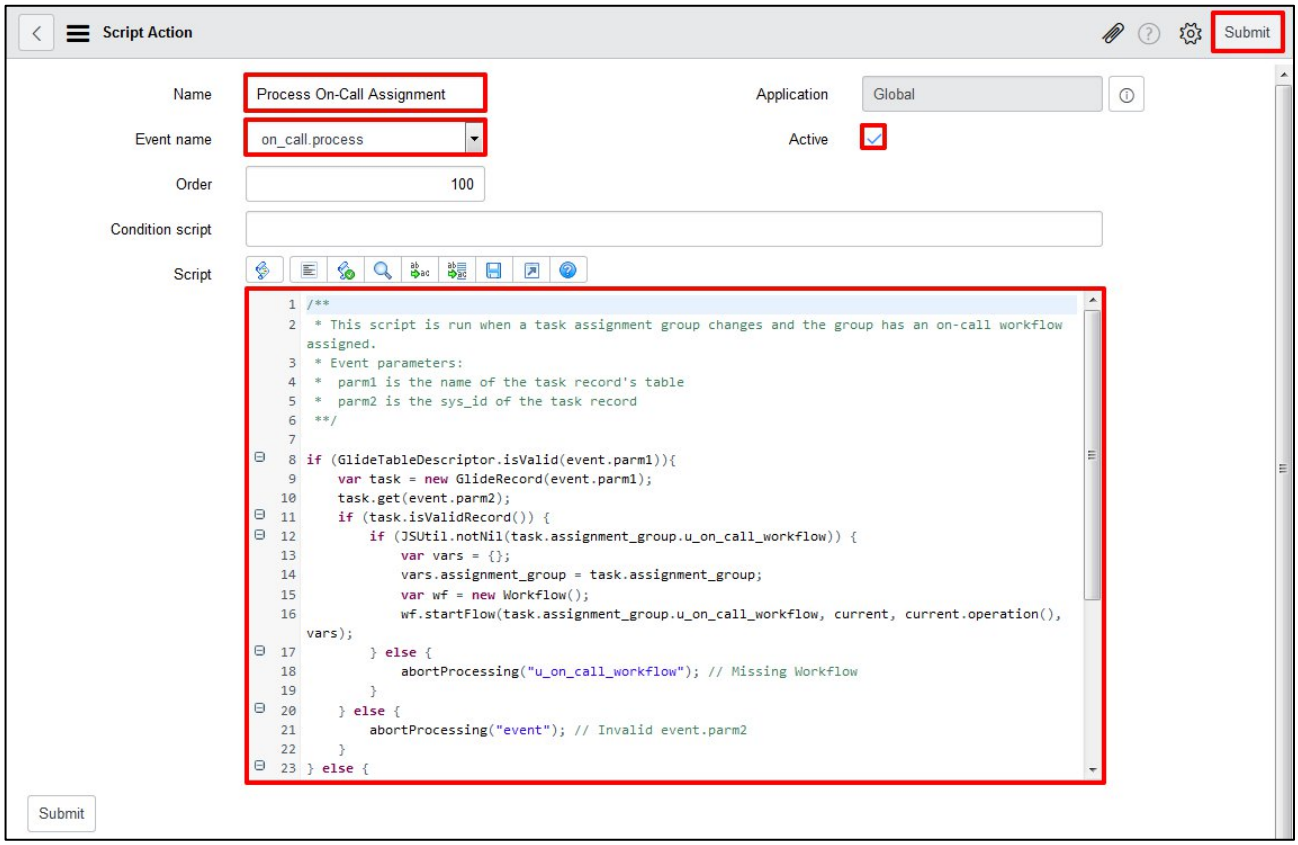
Script:

```
/**
 * This script is run when a task assignment group changes and the group has
an on-call workflow assigned.
 * Event parameters:
 *  parm1 is the name of the task record's table
 *  parm2 is the sys_id of the task record
 **/

if (GlideTableDescriptor.isValid(event.parm1)){
    var task = new GlideRecord(event.parm1);
    task.get(event.parm2);
    if (task.isValidRecord()) {
        if (JSUtil.notNil(task.assignment_group.u_on_call_workflow)) {
            var vars = {};
            vars.assignment_group = task.assignment_group;
            var wf = new Workflow();
            wf.startFlow(task.assignment_group.u_on_call_workflow, current,
current.operation(), vars);
        } else {
            abortProcessing("u_on_call_workflow"); // Missing Workflow
        }
    } else {
        abortProcessing("event"); // Invalid event.parm2
    }
} else {
    abortProcessing("event"); // Invalid event.parm1
}

function abortProcessing(failedField){
    if (failedField=="event") {
        gs.logWarning("Task On-Call processing aborted: Event on-call.process
does not contain valid parameters in parm1, parm2");
    } else {
        gs.logWarning("Task On-Call processing aborted: Misconfigured field "
+ failedField + " on assignment group " + task.assignment_group.sys_id);
    }
}
```

®2015 ServiceNow, Inc. All rights reserved.



36. Click **Submit**. If you receive any errors saving the record please check your script and compare it to the Lab 10 Script Action Script that you can access in the Lab Management application.

37. Now test it! In the Filter Text box, type **incident**.

38. Click **Create New** in the Incident Application.

39. Set the Configuration Item to **Blackberry** and the Short Description to **Testing Business Rule**.

40. Click **Additional actions** and select **Save** to remain on this record.

41. Notice the incident has been assigned to the Hardware Group.

42. Momentarily you should receive an SMS message asking you to accept the newly created incident.

©2015 ServiceNow, Inc. All rights reserved.

Lab Success Verification

1. Incidents should automatically be assigned to the CI Support Group.
2. The On-Call person will receive an SMS message to accept or reject the assignment.
3. Navigate to **Lab Management > Report Lab Progress. Click I am done!**

®2015 ServiceNow, Inc. All rights reserved.

Lab Goal

Lab 11 SMS Reactor Rules

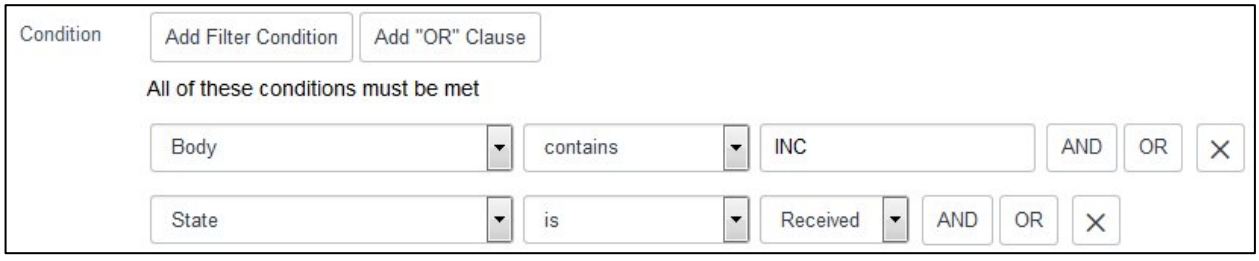
In the On-Call Scheduling Lab, we reviewed **Notify** and **Twilio's** bidirectional SMS capability. A ServiceNow consultant took that one step further and created an SMS Reactor Rule capability that allows you to react to an incoming SMS message. This feature is available on ServiceNow Share:

https://share.servicenow.com/app.do#/detailV2/041d5d6e2b367100bddaff70f8da155c/overview

The update set has been preloaded into your instance.

Send an SMS Message to Your Twilio Account to Create a New Incident

1. In the Filter Text box, type **notify**.
2. In the **Content from Share** section, click **SMS Reactor Rules**.
3. The update set on Share does not provide an example SMS Reactor Rule by default, but one was added to your lab instance.
4. In the **SMS Reactor Rules List**, click the one example **Generate Incident from SMS**.
5. When Twilio receives an SMS message, it forwards that onto your ServiceNow instance. Those messages show up in the Notify\Messages table. These reactor rules execute when a new Message is added to the table.
6. Notice in the Generate Incident from the SMS example that we have access to filter any of the data coming from Twilio. In this example look for an SMS message to contain the characters **INC** and when the State is Received. If those conditions are met, the script below executes.



©2015 ServiceNow, Inc. All rights reserved.

7. The script allows you to do anything that ServiceNow scripting allows. In this example, since the SMS message contains INC, generate an incident. In doing so it queries the **User** table for a mobile phone that matches the SMS From and sets that person as the caller of the incident.

8. Test this SMS Reactor Rule: By now you should have received several SMS messages from your lab instance, and your Twilio phone number should be in your phone already. If not, you click **Properties** in the **Notify** application to see your Twilio number.

9. From your phone, compose a new SMS message to your Twilio phone number with the following message: **INC – testing SMS reactor rules**. Note that the conditions are **CaSe** sensitive; make sure you capitalize the **INC**.

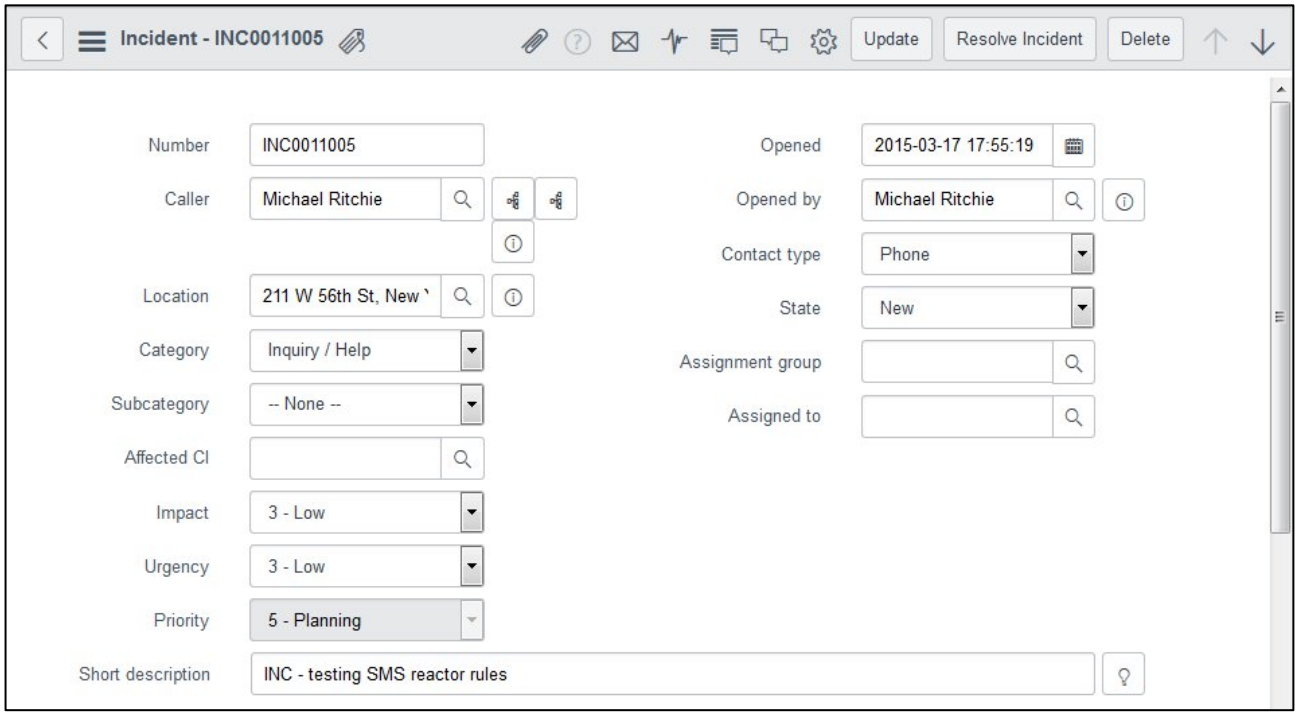


10. Within seconds, a new incident is created. In the Filter Text box, type **incident**.

11. Click **Open** in the Incident application.

©2015 ServiceNow, Inc. All rights reserved.

12. In the Incident list you should have a new incident created from your SMS Message:



Lab Success Verification

1. An Incident was created by you sending an SMS message to your Twilio phone number.
2. Navigate to **Lab Management > Report Lab Progress. Click I am done!**

Note: Your lab instance also includes the SMS Approval Rules also found on ServiceNow Share. We do not cover this in this lab, but this feature allows users to approve their approval records via SMS. https://share.servicenow.com/app.do#/detailV2/51a976202bd3a1004a1e976be8da154f/overview

©2015 ServiceNow, Inc. All rights reserved.

Lab Goal

Lab 12 Incident Click to Call

Twilio offers a **Click to Call** feature that can bridge two people together with the click of a button. This feature is helpful in Incident Management because it can connect the technician working a ticket with the user having the issue.

This feature is not out-of-the-box, but the content of this feature is shared via an update set on ServiceNow Share.

https://share.servicenow.com/app.do#/detailV2/be2709a02b87b1004a1e976be8da15f5/overview

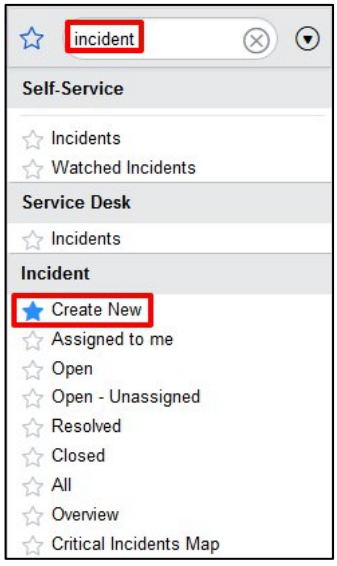
The update set has been preloaded into your instance.

Create a New Incident to Test Click to Call

1. Please partner with someone and decide who will be the caller and who will be the technician. **If you are the Caller please utilize your instance** for this lab because your mobile number is already stored in Beth Anglin's user record.

2. In the Filter Text box, type **incident**.

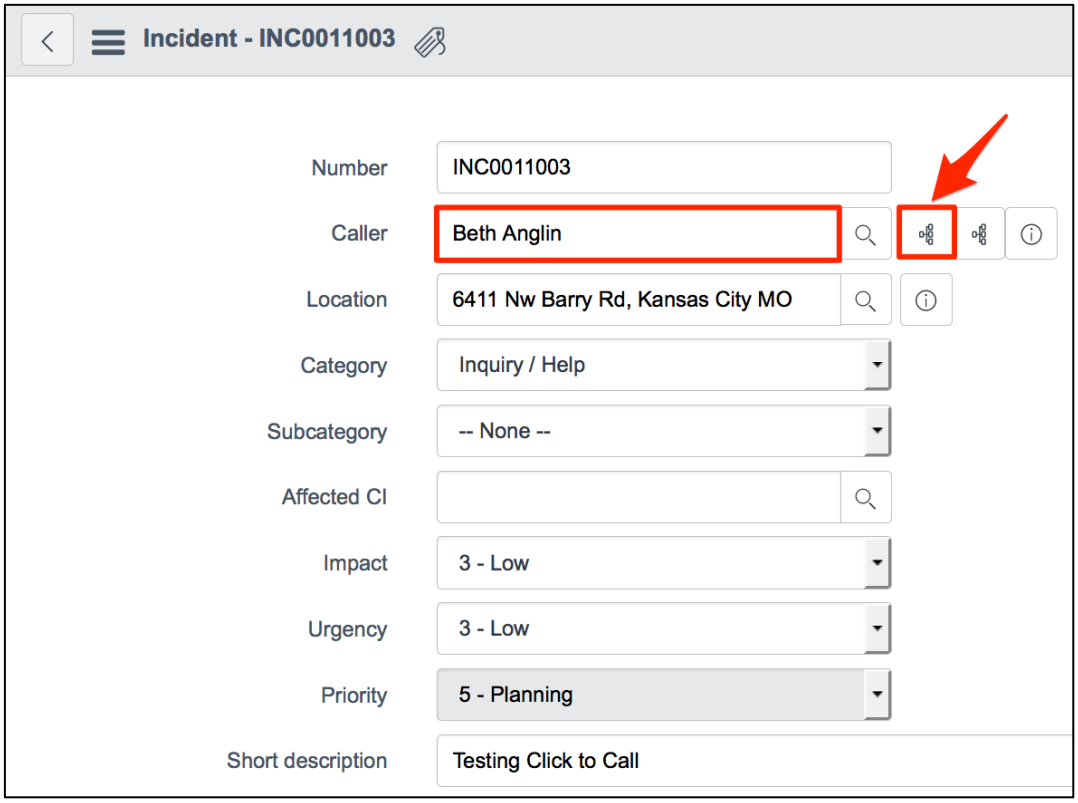
3. Click **Create New** under the Incident application.



©2015 ServiceNow, Inc. All rights reserved.

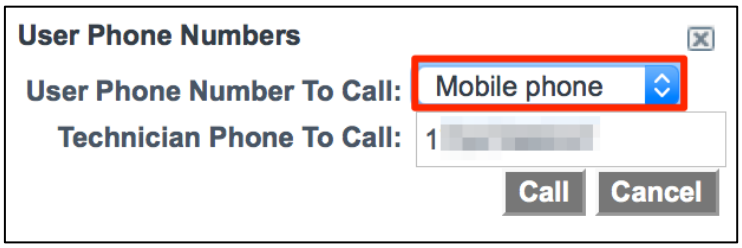
4. In the **Caller** field type/select **Beth Anglin** and in the **Short Description** field type **Testing Click to Call**.

5. Click **Additional actions** and select **Save**, which leaves the screen on this record.



6. Two UI macros appear beside the caller. Hover the cursor over the first one. It should say **Call User**. Click that button.

7. A popup dialog appears. The dropdown contains a list of phone numbers currently entered in the Caller's user record. In Beth Anglin's case, only her mobile number populates her user record; therefore, you only have one choice.



©2015 ServiceNow, Inc. All rights reserved.

8. Select Mobile phone and **enter your partner's mobile number** in the Technician Phone field. Confirm that the phone number is entered in **E.164** format such as **+17045551212**.

   Note: The Technician phone number is saved as a user preference so the Technician does not have to reenter the phone each time. This was designed deliberately, in case the Technician was working from home or in another location.

9. **Click** the **Call** button.

10. Shortly the Technician number is called first. This person is prompted to "**Please press 1 to connect your call**". **Press 1**. The idea here is to get the Technician on the line first to mimic as if the Technician called the Caller directly.

11. Shortly after pressing 1, the Caller's phone should ring. Upon answering this call you and your partner will be on a call together.

Lab Success Verification

1. You and your partner bridged together on a call from an Incident record.

2. Navigate to **Lab Management > Report Lab Progress. Click I am done!**

®2015 ServiceNow, Inc. All rights reserved.

Lab Goal

Twilio offers text-to-voice capabilities, which we saw when we initiated a conference call. This lab explains how to leverage this feature to call Incident Alert users and read details from the Incident Alert.

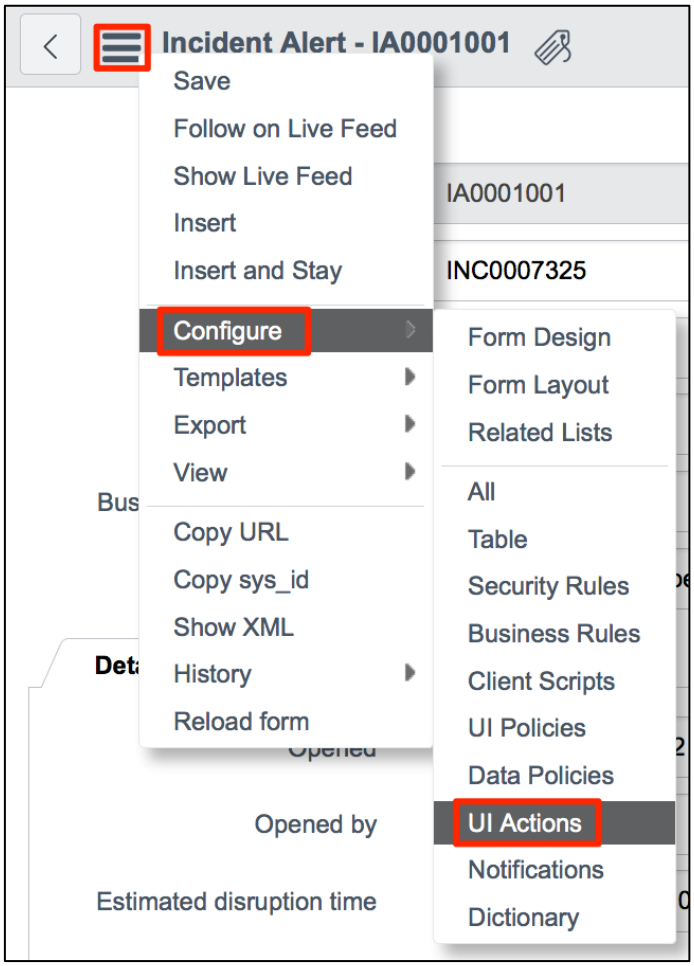
Lab 13
Initiate
Voice Alert

Login as System Administrator to Configure Your Instance for Voice Alerts

1. First create a UI action in the **Incident Alert** application that a user can click to initiate the voice call to the users.
2. In the Filter Text box, type **alert**. Click **Open** under the Incident Alert Management application.
3. Click on any Incident Alert in the list to pull up the Incident Alert form.

®2015 ServiceNow, Inc. All rights reserved.

4. Once the incident alert record is on your screen, click **Additional Actions**, then **Configure**, then click **UI Actions**.



5. Click **New** to create a new UI Action.

   Enter the following information:

   Name: **Initiate Voice Alert**
   Show insert: **unchecked**
   Show update: **checked**
   Form Link: **checked**
   Condition: **leave blank** – for this lab, it available to anyone, but you can lock this down as needed in certain situations or users.

6. Click **Additional actions** and select **Save**, which leaves the screen on this record.

©2015 ServiceNow, Inc. All rights reserved.

7. Next define a script to loop through the User Contacts and call each one of them if they have a mobile phone number defined.

   Note: The steps below walk you through creating this script with explanation. You can also copy the entire script from the Lab Management application under the **Lab Exercises** section by clicking **Lab 13 – UI Action Script**.

8. The following script selects the User Contact records associated to the incident alert and loops through them. It calls a **placeCall()** function to make the phone call. In this simple example, query for all the Incident Alert User Contact records and loop through them calling each one. The complexity can be easily expanded to meet your needs.

```
var incAlertUsers = new GlideRecord("contact");
incAlertUsers.addQuery("document", current.sys_id);
incAlertUsers.addQuery("table", "incident_alert");
incAlertUsers.addQuery("type", "sys_user");
incAlertUsers.query();
while (incAlertUsers.next()) {
  if (!JSUtil.nil(incAlertUsers.user.mobile_phone)) {
        placeCall(incAlertUsers.user.mobile_phone);
  }
}
//Remain on the incident alert
action.setRedirectURL(current);
```

9. The above code calls a **placeCall()**. Now create that function in the UI Action that makes a REST call to Twilio. The Twilio API is documented here for your reference:

   https://www.twilio.com/docs/api

©2015 ServiceNow, Inc. All rights reserved.

10. Below your code append the following to declare a function that we can insert code into:

```
function placeCall(userPhone) {

}
```

11. Instead of hard coding the Twilio API credentials, get them from the **Notify System Properties**. Append the following within your **placeCall function** – between the **{}** brackets:

```
var twilioNumber = gs.getProperty("glide.notifynow.twilio.phonenumber");
var twilioAccSID = gs.getProperty("glide.notifynow.twilio.accountsid");
var instanceURL = gs.getProperty("glide.servlet.uri");
var twilioToken = gs.getProperty("glide.notifynow.twilio.token");
```

Note: Be careful with cutting and pasting from this document: the double quotes may need to be retyped.

12. So far your script should look similar to the example.

```
1  //In this simple example we will query for all the Incident Alert User Contact records and loop through them calling each one.
2  var incAlertUsers = new GlideRecord("contact");
3  incAlertUsers.addQuery("document", current.sys_id);
4  incAlertUsers.addQuery("table", "incident_alert");
5  incAlertUsers.addQuery("type", "sys_user");
6  incAlertUsers.query();
7  while (incAlertUsers.next()) {
8      if (!JSUtil.nil(incAlertUsers.user.mobile_phone)) {
9          placeCall(incAlertUsers.user.mobile_phone);
10     }
11 }
12 //Remain on the incident alert
13 action.setRedirectURL(current);
14
15 //This function will make a REST call to Twilio to place the call.
16 function placeCall(userPhone) {
17     var twilioNumber = gs.getProperty("glide.notifynow.twilio.phonenumber");
18     var twilioAccSID = gs.getProperty("glide.notifynow.twilio.accountsid");
19     var instanceURL = gs.getProperty("glide.servlet.uri");
20     var twilioToken = gs.getProperty("glide.notifynow.twilio.token");
21
22 }
```

13. New in Fuji is the ability to make a REST API call via script without creating the Rest Message record in the System Web Services application. We utilize that feature here. The following Wiki article describes this new feature:

    http://wiki.servicenow.com/index.php?title=Scripting_Outbound_REST

©2015 ServiceNow, Inc. All rights reserved.

14. Twilio is expecting a post to their Calls API. Along with the post, pass the Twilio Account **SID** and **Token** from your Twilio Account. Append the following into the function:

```
var restMessage = new sn_ws.RESTMessageV2();
restMessage.setHttpMethod("post");
restMessage.setBasicAuth(twilioAccSID, twilioToken);
restMessage.setEndpoint("https://api.twilio.com/2010-04-01/Accounts/" +
twilioAccSID + "/Calls");
restMessage.setRequestHeader("Content-Type", "application/x-www-form-
urlencoded");
```

15. So far the **placeCall()** script should look similar to the example.

```
16 function placeCall(userPhone) {
17     var twilioNumber = gs.getProperty("glide.notifynow.twilio.phonenumber");
18     var twilioAccSID = gs.getProperty("glide.notifynow.twilio.accountsid");
19     var instanceURL = gs.getProperty("glide.servlet.uri");
20     var twilioToken = gs.getProperty("glide.notifynow.twilio.token");
21
22     var restMessage = new sn_ws.RESTMessageV2();
23     restMessage.setHttpMethod("post");
24     restMessage.setBasicAuth(twilioAccSID, twilioToken);
25     restMessage.setEndpoint("https://api.twilio.com/2010-04-01/Accounts/" + twilioAccSID + "/Calls");
26     restMessage.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
27
28 }
```

16. Next tell Twilio what to do through the Rest Message Request Body. According to Twilio's API documentation for making a call, there are several required parameters that need to be passed: https://www.twilio.com/docs/api/rest/making-calls

    - **From phone number**: Twilio accounts support multiple phone numbers so this will be the number that shows up on the caller's phone. Note that ServiceNow **Notify** currently only supports one number per instance. This will be changed in a future release.
    - **To phone number**: This is the number you expect Twilio to call.
    - **URL to consult**: This is a reference back to your ServiceNow instance that will instruct Twilio what to do next such as read text or prompt for phone input once the call is answered. In this example the URL will include a custom URL suffix called "IncidentAlertCall.do", which will be created later, and the **SysID** of the Incident Alert that will be referred back to for information.

©2015 ServiceNow, Inc. All rights reserved.

17. Twilio expects the request body to be encoded. Encode each value with a JavaScript function. Append the following into your function:

```
var encFrom = GlideStringUtil.urlEncode(twilioNumber);
var encTo = GlideStringUtil.urlEncode(userPhone);
var encURL = GlideStringUtil.urlEncode(instanceURL +
"IncidentAlertCall.do?sysparm_messageid=" + current.sys_id);
var requestBody = "From=" + encFrom + "&To=" + encTo + "&Url=" + encURL;
restMessage.setRequestBody(requestBody);
```

18. Finally, invoke the REST API to Twilio. Append the following into your function:

```
var response = restMessage.execute();
```

19. Your **placeCall()** function script should look similar to the example.

```
16 function placeCall(userPhone) {
17     //Get the Twilio account information from the Notify system properties so we don't have to hard code it.
18     var twilioNumber = gs.getProperty("glide.notifynow.twilio.phonenumber");
19     var twilioAccSID = gs.getProperty("glide.notifynow.twilio.accountsid");
20     var instanceURL = gs.getProperty("glide.servlet.uri");
21     var twilioToken = gs.getProperty("glide.notifynow.twilio.token");
22
23     var restMessage = new sn_ws.RESTMessageV2();
24     restMessage.setHttpMethod("post");
25     restMessage.setBasicAuth(twilioAccSID, twilioToken);
26     restMessage.setEndpoint("https://api.twilio.com/2010-04-01/Accounts/" + twilioAccSID + "/Calls");
27     restMessage.setRequestHeader("Content-Type", "application/x-www-form-urlencoded");
28     //Twilio expects the request body to be encoded so we will encode each value.
29     var encFrom = GlideStringUtil.urlEncode(twilioNumber);
30     var encTo = GlideStringUtil.urlEncode(userPhone);
31     var encURL = GlideStringUtil.urlEncode(instanceURL + "IncidentAlertCall.do?sysparm_messageid=" + current.sys_id);
32     var requestBody = "From=" + encFrom + "&To=" + encTo + "&Url=" + encURL;
33     restMessage.setRequestBody(requestBody);
34     var response = restMessage.execute();
35 }
```

20. Click **Update** to save the new UI Action. If you receive any errors while saving the record, check your script and compare it to the **Lab 13 UI Action Script**, accessible in the **Lab Management** application.

©2015 ServiceNow, Inc. All rights reserved.

21. As noted above, Twilio expects a URL to be passed to it so it knows what to do once the call is imitated and what to say to the caller. To accomplish this, leverage a platform feature called **Processors**. You have likely leveraged Processors before without realizing. They are used in CTI Processing, Exporting Lists, Downloading Files, and even extracting an update set to an XML. Once you create the new processor, you need to make it public (no login required) so that Twilio can access it. The following post on the ServiceNow Community provides further explanation:

    https://community.servicenow.com/people/Christopher.Maloy/blog/2012/05/24/2579

22. In the Filter Text box, type **processors**. Click **Processors** under the **System Definition** application.

23. Click **New** to create a new processor.

24. Enter the information below.

    Name: **IncidentAlertCall**
    Type: **Script**
    Path: **IncidentAlertCall**

25. Click **Additional actions** and select **Save** to remain on this record.

26. Next, define a script that tells Twilio what to say to the caller. **Note**: The steps below walk you through creating this script with explanation. The entire script can also be copied from the Lab Management application under the **Lab Exercises** section by clicking **Lab 13 – Processor Script.**

27. In the UI Action you passed Twilio the **SysID** of the Incident Alert record, which you now can leverage. Capture the **SysID** from the Twilio API call back to your instance and set several other variables. Paste the following into the script field:

```
//Instead of hardcoding the instance in this script, lets get it
dynamically
var instanceURI = gs.getProperty("glide.servlet.uri");
//Get the Incident Alert SysID from the URL so we can query for it.
var messageIDParameter = g_request.getParameter('sysparm_messageid');
```

©2015 ServiceNow, Inc. All rights reserved.

28. Unless you set it up otherwise, Twilio calls the user, speaks the text, and hangs up. There also is an option to repeat the text. The caller can able to press **1** to repeat the message or **2** to end the call. In order to simplify the code required for this feature without requiring another processor to manage, additional variables are set. Twilio appears to allow only one parameter to be passed to it that is then passed back to ServiceNow. Therefore, note that multiple parameter values are sent to the same parameter but are split. If an additional parameter is NOT passed in, then you know it is a new cal; if one is passed such as **callerinput** or **hangup**, then you know it is an in-flight call. Append the following to your script:

```
//Twilio appears to only accept one URL parameter (unconfirmed).  This
script needs to pass two so utilizing the same parameter but separating
them by an underscore.
messageIDParameter = messageIDParameter.toString().split("_");
var messageID = messageIDParameter[0];
var requestParameter = messageIDParameter[1];
//If the request parameter is NULL, then we know this is a new call and
will tell Twilio to read the severity and short description.
var isNewCall = JSUtil.nil(requestParameter);
```

29. Now that you have the parameters passed from Twilio in local variables, query for the Incident Alert record based on the **SysID** passed. Before doing that query, check to see if this is a new call or not based on the variable above. Append the following to your script:

```
if (isNewCall) {
  var incAlert = new GlideRecord("incident_alert");
  incAlert.addQuery("sys_id", messageID);
  incAlert.query();
  if (incAlert.next()) {

  }
}
```

30. Twilio is expecting the response to be in a certain format utilizing certain “verbs” that are documented in their API documentation: https://www.twilio.com/docs/api/twiml

```
1  <?xml version="1.0" encoding="UTF-8" ?>
2  <Response>
3       <Say>Hello World</Say>
4  </Response>
```

©2015 ServiceNow, Inc. All rights reserved.

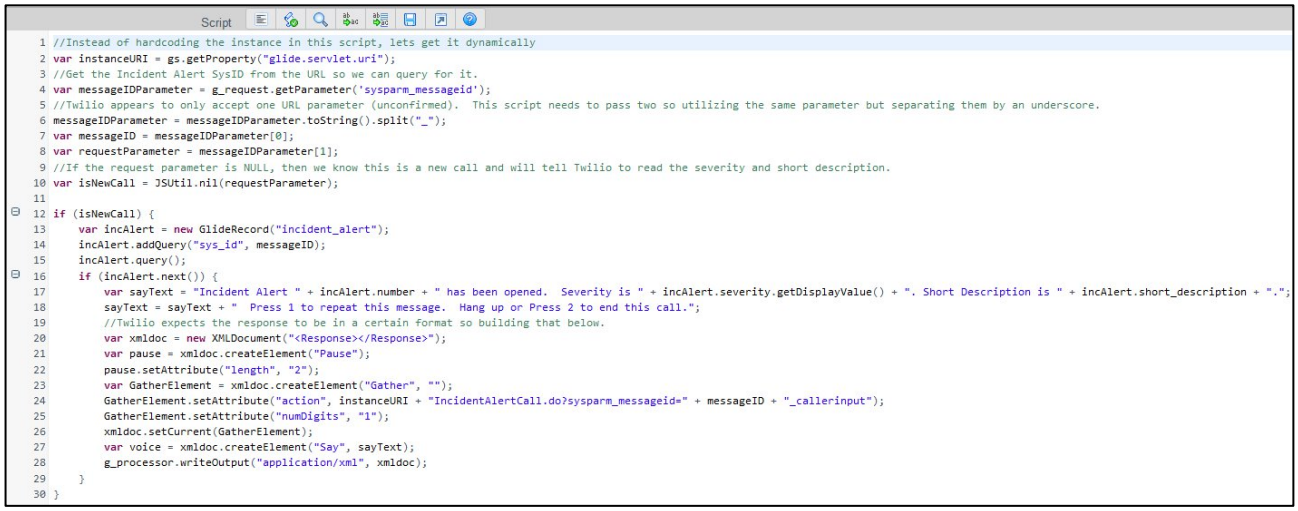
31. ServiceNow includes several functions that allow easy creation of the response in the expected format. The following code does the following:
    - Builds the XML document with the expected tags
    - Creates a variable with a string of text that you want Twilio to speak to the caller. This includes telling the caller to Press 1 to repeat the message or hang up or press 2 to end the call.
    - Tells Twilio to pause 2 seconds before it starts speaking
    - Tells Twilio to wait for input from the caller in case they press 1 to repeat or 2 to end the call.

    Append the following code **within your second If statement**:

```
var sayText = "Incident Alert " + incAlert.number + " has been opened.
Severity is " + incAlert.severity.getDisplayValue() + ". Short Description
is " + incAlert.short_description + ".";
sayText = sayText + " Press 1 to repeat this message.  Hang up or Press 2
to end this call.";
//Twilio expects the response to be in a certain format so building that
below.
var xmldoc = new XMLDocument("<Response></Response>");
var pause = xmldoc.createElement("Pause");
pause.setAttribute("length", "2");
var GatherElement = xmldoc.createElement("Gather", "");
GatherElement.setAttribute("action", instanceURI +
"IncidentAlertCall.do?sysparm_messageid=" + messageID + "_callerinput");
GatherElement.setAttribute("numDigits", "1");
xmldoc.setCurrent(GatherElement);
var voice = xmldoc.createElement("Say", sayText);
g_processor.writeOutput("application/xml", xmldoc);
```

©2015 ServiceNow, Inc. All rights reserved.

32. At this point, your script should look similar to the example.



```
Script
1  //Instead of hardcoding the instance in this script, lets get it dynamically
2  var instanceURI = gs.getProperty("glide.servlet.uri");
3  //Get the Incident Alert SysID from the URL so we can query for it.
4  var messageIDParameter = g_request.getParameter('sysparm_messageid');
5  //Twilio appears to only accept one URL parameter (unconfirmed).  This script needs to pass two so utilizing the same parameter but separating them by an underscore.
6  messageIDParameter = messageIDParameter.toString().split("_");
7  var messageID = messageIDParameter[0];
8  var requestParameter = messageIDParameter[1];
9  //If the request parameter is NULL, then we know this is a new call and will tell Twilio to read the severity and short description.
10 var isNewCall = JSUtil.nil(requestParameter);
11
12 if (isNewCall) {
13     var incAlert = new GlideRecord("incident_alert");
14     incAlert.addQuery("sys_id", messageID);
15     incAlert.query();
16     if (incAlert.next()) {
17         var sayText = "Incident Alert " + incAlert.number + " has been opened.  Severity is " + incAlert.severity.getDisplayValue() + ". Short Description is " + incAlert.short_description + ".";
18         sayText = sayText + "  Press 1 to repeat this message.  Hang up or Press 2 to end this call.";
19         //Twilio expects the response to be in a certain format so building that below.
20         var xmldoc = new XMLDocument("<Response></Response>");
21         var pause = xmldoc.createElement("Pause");
22         pause.setAttribute("length", "2");
23         var GatherElement = xmldoc.createElement("Gather", "");
24         GatherElement.setAttribute("action", instanceURI + "IncidentAlertCall.do?sysparm_messageid=" + messageID + "_callerinput");
25         GatherElement.setAttribute("numDigits", "1");
26         xmldoc.setCurrent(GatherElement);
27         var voice = xmldoc.createElement("Say", sayText);
28         g_processor.writeOutput("application/xml", xmldoc);
29     }
30 }
```

33. Next, you need to handle what happens if the caller does press 1 to repeat message or 2 to end the call. In the above code you sent an **action** to tell Twilio to consult this processor again if the user does press a digit on the phone, along with passing an additional parameter value back to this processor **callerinput**. Based on the first **If** statement, the script skips that code. You need to add an additional **Else If** statement to analyze the input from the user. Append the following code to the end of your script **before the last }**:

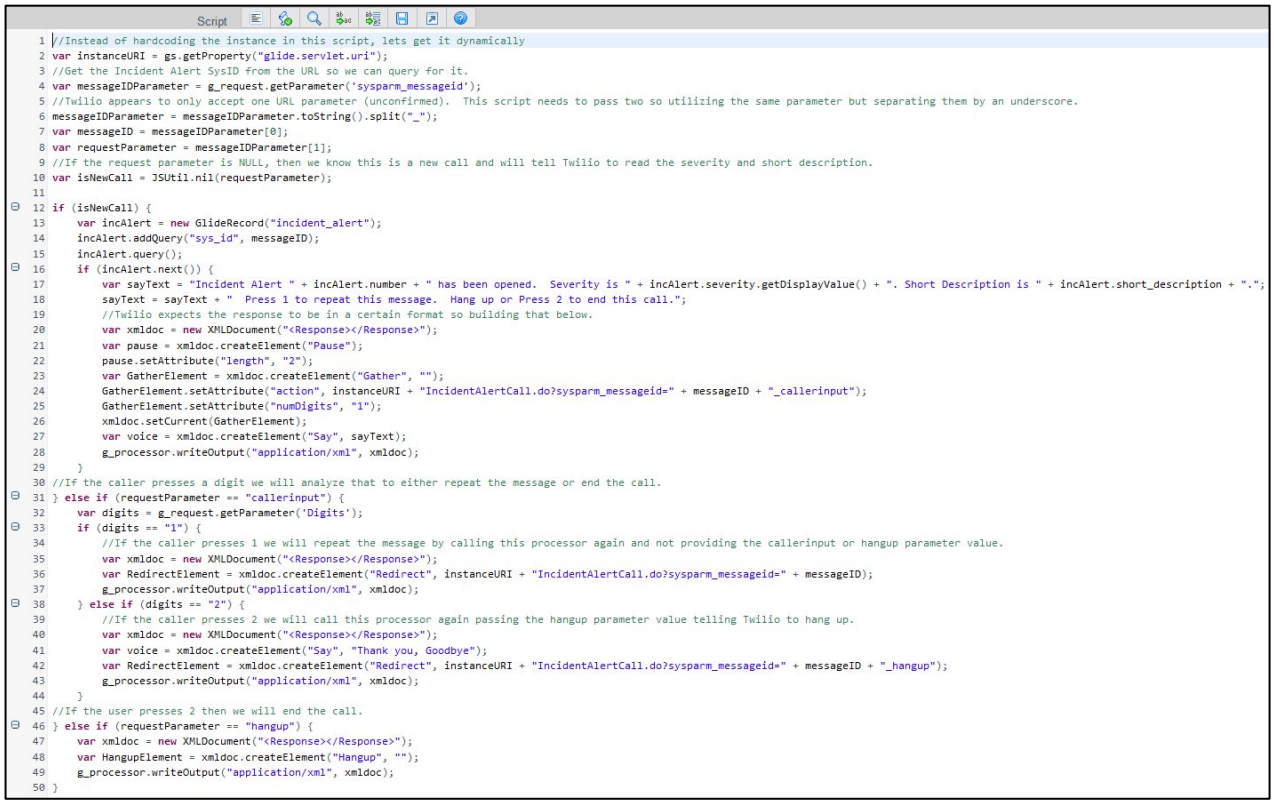
```
//If the caller presses a digit we will analyze that to either repeat the
message or end the call.
} else if (requestParameter == "callerinput") {
 var digits = g_request.getParameter('Digits');
 if (digits == "1") {
       //If the caller presses 1 we will repeat the message by calling this
processor again and not providing the callerinput or hangup parameter
value.
       var xmldoc = new XMLDocument("<Response></Response>");
       var RedirectElement = xmldoc.createElement("Redirect", instanceURI +
"IncidentAlertCall.do?sysparm_messageid=" + messageID);
       g_processor.writeOutput("application/xml", xmldoc);
 } else if (digits == "2") {
       //If the caller presses 2 we will call this processor again passing
the hangup parameter value telling Twilio to hang up.
       var xmldoc = new XMLDocument("<Response></Response>");
       var voice = xmldoc.createElement("Say", "Thank you, Goodbye");
       var RedirectElement = xmldoc.createElement("Redirect", instanceURI +
"IncidentAlertCall.do?sysparm_messageid=" + messageID + "_hangup");
       g_processor.writeOutput("application/xml", xmldoc);
 }
```

©2015 ServiceNow, Inc. All rights reserved.

34. Finally, you need to handle what happens is the user does press 2 to end the call. You need to tell Twilio to hang up. Append the following code to the end of your script **before the last }**:

```
//If the user presses 2 then we will end the call.
} else if (requestParameter == "hangup") {
   var xmldoc = new XMLDocument("<Response></Response>");
   var HangupElement = xmldoc.createElement("Hangup", "");
   g_processor.writeOutput("application/xml", xmldoc);
```

35. Your script should look like the example.



```
Script
1  //Instead of hardcoding the instance in this script, lets get it dynamically
2  var instanceURI = gs.getProperty("glide.servlet.uri");
3  //Get the Incident Alert SysID from the URL so we can query for it.
4  var messageIDParameter = g_request.getParameter('sysparm_messageid');
5  //Twilio appears to only accept one URL parameter (unconfirmed).  This script needs to pass two so utilizing the same parameter but separating them by an underscore.
6  messageIDParameter = messageIDParameter.toString().split("_");
7  var messageID = messageIDParameter[0];
8  var requestParameter = messageIDParameter[1];
9  //If the request parameter is NULL, then we know this is a new call and will tell Twilio to read the severity and short description.
10 var isNewCall = JSUtil.nil(requestParameter);
11
12 if (isNewCall) {
13     var incAlert = new GlideRecord("incident_alert");
14     incAlert.addQuery("sys_id", messageID);
15     incAlert.query();
16     if (incAlert.next()) {
17         var sayText = "Incident Alert " + incAlert.number + " has been opened.  Severity is " + incAlert.severity.getDisplayValue() + ". Short Description is " + incAlert.short_description + ".";
18         sayText = sayText + "  Press 1 to repeat this message.  Hang up or Press 2 to end this call.";
19         //Twilio expects the response to be in a certain format so building that below.
20         var xmldoc = new XMLDocument("<Response></Response>");
21         var pause = xmldoc.createElement("Pause");
22         pause.setAttribute("length", "2");
23         var GatherElement = xmldoc.createElement("Gather", "");
24         GatherElement.setAttribute("action", instanceURI + "IncidentAlertCall.do?sysparm_messageid=" + messageID + "_callerinput");
25         GatherElement.setAttribute("numDigits", "1");
26         xmldoc.setCurrent(GatherElement);
27         var voice = xmldoc.createElement("Say", sayText);
28         g_processor.writeOutput("application/xml", xmldoc);
29     }
30 //If the caller presses a digit we will analyze that to either repeat the message or end the call.
31 } else if (requestParameter == "callerinput") {
32     var digits = g_request.getParameter('Digits');
33     if (digits == "1") {
34         //If the caller presses 1 we will repeat the message by calling this processor again and not providing the callerinput or hangup parameter value.
35         var xmldoc = new XMLDocument("<Response></Response>");
36         var RedirectElement = xmldoc.createElement("Redirect", instanceURI + "IncidentAlertCall.do?sysparm_messageid=" + messageID);
37         g_processor.writeOutput("application/xml", xmldoc);
38     } else if (digits == "2") {
39         //If the caller presses 2 we will call this processor again passing the hangup parameter value telling Twilio to hang up.
40         var xmldoc = new XMLDocument("<Response></Response>");
41         var voice = xmldoc.createElement("Say", "Thank you, Goodbye");
42         var RedirectElement = xmldoc.createElement("Redirect", instanceURI + "IncidentAlertCall.do?sysparm_messageid=" + messageID + "_hangup");
43         g_processor.writeOutput("application/xml", xmldoc);
44     }
45 //If the user presses 2 then we will end the call.
46 } else if (requestParameter == "hangup") {
47     var xmldoc = new XMLDocument("<Response></Response>");
48     var HangupElement = xmldoc.createElement("Hangup", "");
49     g_processor.writeOutput("application/xml", xmldoc);
50 }
```

36. Click **Update**. If you receive any errors while saving the record, \ check your script and compare it to the **Lab 13 Processor Script** that you can access in the Lab Management application.

37. By default ServiceNow requires an authenticated user to call a Processor record or any URL for that matter. Since Twilio cannot pass credentials you have to make this processor publicly accessible.

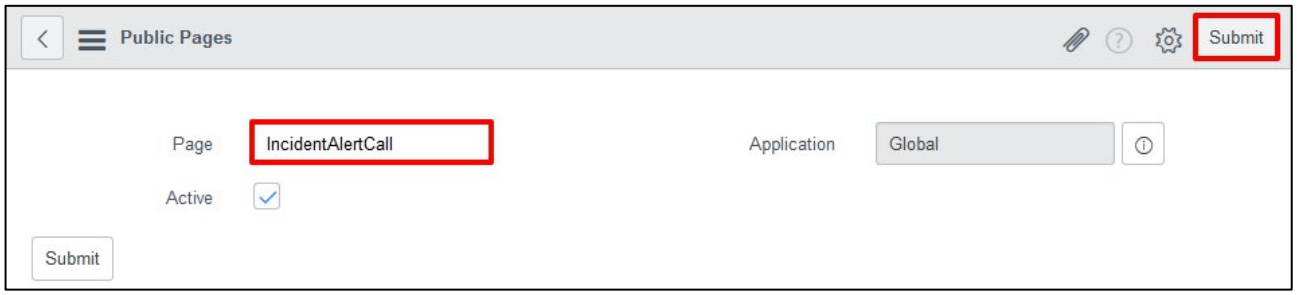
©2015 ServiceNow, Inc. All rights reserved.

38. In the Filter Text box, type **sys_public.list** – similar to below. The last "**t**" was not entered on purpose in the screenshot below because as soon as you do, it will take you to the list of sys_public records.



39. After the Public Pages list appears, click **New**.

40. In the **Page** field, enter the path value from our processor: **IncidentAlertCall**.



41. Click **Submit.**

42. Now you are ready to test.

43. In the Filter Text box, type **alert**. Click **Open** under the Incident Alert Management application.

44. In the list on the right, click **IA0001003**, or the lowest numbered Incident Alert that you created initially for the Exchange server outage.

45. Scroll down to the Related Links and click **Initiate Voice Alert**.

46. Shortly you should receive a call speaking details about this incident alert.

Lab Success Verification

1. You should receive a call to your mobile device with Twilio speaking details about the incident alert record.

2. Navigate to **Lab Management > Report Lab Progress. Click I am done!**

©2015 ServiceNow, Inc. All rights reserved.