splunk>

Implementing IT Service Intelligence Lab Exercises

Welcome to the Implementing Splunk IT Service Intelligence class. There is one lab exercise for each module of the course. Your instructor will assign you a server and credentials to log on to your server to do your lab work. Your lab server is a cloud instance running on Amazon Web Services. You will need a web browser to connect to the server's Splunk web interface.

Lab Credentials

Your instructor will provide the information needed to fill in this table at the beginning of your first lab exercise. You will need this information to log on to the Splunk server running ITSI. For lab exercises 1 through 4, we will share one server while we work in "user" mode to learn how ITSI works. In lab exercise 5 and onwards, you will work on your own on a private server to configure a new ITSI system.

Get the information from your instructor for the shared "DEMO" server and enter it here:

DEMO Server Address	
Splunk Analyst UID/PWD	**analyst__** /

splunk>

Lab Exercise 1: ITSI User Interface

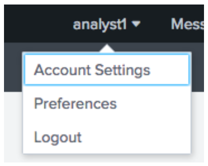
Description

In this exercise, you log in to a demo Service Intelligence server and explore its user interface.

Steps

Task 1: Examine the Service Analyzer and related views.

1. Use your web browser to connect to your ITSI server as an ITSI analyst, using the connection and credential information provided by your instructor. Your Splunk user account is a member of the **itoa_analyst** role.
2. After you log in, you see the Service Analyzer because ITSI is the default app for your user account.
3. First, set up your user account: click your user name (**analyst*N***) and select **Account Settings**. (This is used for grading your lab exercises, so you will receive credit for completing the course.)



4. Set the **Full Name** field to your student ID - first and last name.
   For example: `5 - Mary Roberts`
5. Click **Save**.
6. Click the **splunk>enterprise** logo to return to the **Service Analyzer** page. In the Service Analyzer, note that two services are configured: Online Sales and Web Farm. You see all their KPIs, in descending level of alert severity: critical (if any) first, then high, etc.
7. Click the tree view icon to see the structure of the dependencies between services.
8. Click **Web Farm**. A side panel opens with details about its KPIs and Critical and High Episodes (if any). Notice the Web Farm icon in the tree diagram is now outlined, showing that it is the selected item.

   Notice the Web Farm circle is now outlined and centered in the tree diagram. At the right, notice that its KPIs are displayed. From the panels on the right, you could do one of several things: click **Open all in Deep Dive** to view the listed KPIs in a deep dive, or, at the bottom, click **View All** to view critical and high episodes, if any; episodes list may be empty. Only admins can define services, but you can look at how they were defined. You'll explore deep dives later.
9. In the tree diagram, by default, the selected service is centered. Click to zoom and center the entire tree.
10. In the KPIs list, click the **CPU Overutilization: % System** KPI.

    An additional panel opens displaying the entities associated with the KPI.
11. Click the **Storage Free Space: % System** KPI.

    Notice it contains the same hosts, but the severity levels for each are specific to this KPI.
12. Click **www1**.

    A new browser tab opens. A sortable table displaying the KPIs with which www1 is associated are listed along with their current severity level, service, sparkline, and alert value. Another table displays similar information about services associated with www1, and another with any associated notable events.

13. To view specific information about how www1 is performing, click **OS Host Details**.

    There are several tabs of information you can explore to see the information available.

14. Close the two most recent browser tabs and return to the **Default Service Analyzer** browser tab.

Task 2: Filter Service Analyzer views and save new Service Analyzers.

15. Click **Service Analyzer > Service Analyzers**.

    Note there are no saved service analyzers yet.

16. Return to the default service analyzer.

    Note that it has both a **Save** and a **Save as…** button.

17. Click **Save as…** to create a new service analyzer, name it **All Services** and keep it private. In the Description field, indicate this saved analyzer is your backup and click **Create**.

18. Use the **Filter Services** field to select only the **Web Farm** service.

    Note that now only the Web Farm service and its KPIs are displayed.

19. Click **Save as…** and name your analyzer **# - Web Farm** (where # is your `student ID`), set Permissions to **Shared in App** and click the **Create** button.

20. Make other changes to your **# - Web Farm** service analyzer, such as altering the tile size, time range, maximum number of KPIs (settings ⚙), and selecting Tree view.

21. Click **Save** to make these changes permanent.

22. Use the Standard View ⤢ link to toggle between standard and full screen view. (You may want to save some of your Service Analyzers in full screen because it is useful for operations center displays.)

Task 3: Examine potential issues based on severity by drilling down from KPIs to a filtered deep dive.

23. Display your analyzer in **Tile** view.

24. Hover over a KPI tile that is yellow or worse.

    Notice a checkmark appears in the tile's upper right corner.

25. Click the checkmark.

26. Hover and click the checkmarks for any other KPIs that are yellow or worse.

27. Click the **Drilldown to Deep Dive** link. A deep dive opens comprised only of swim lanes for the KPIs you selected. You'll explore these in detail later.

Task 4: Explore other parts of ITSI.

28. On the main ITSI menu, click **Episode Review**. Review a few episodes in the table.

29. Click **Dashboards > Predictive Analytics**.

30. Click the **Service** dropdown and select **Online Sales**.

31. Under the **Model** dropdown, select **LinearRegression**. Notice the predicted health score in 30 minutes.

32. Click **Cause Analysis**. Notice you can see the top 5 contributing KPIs or open them in a Deep Dive for closer examination.

Optional Task 1: Model changes to Service Health Scores.

33. In the ITSI menu bar, click **Configure > Services**.
34. Click the **Web Farm** service.
35. Click the **Settings** tab.
36. Using the dropdowns under **Web Farm KPIs**, you can model how various combinations of KPI or dependent service severity changes would impact the entire service's health score.
37. Examine the **Simulated Health Score** visualization colors and values changing as you model various changes in KPI severities.

Optional Task 2: Use Search to examine underlying ITSI data.

38. Select **Search > Search**.

    A Splunk search window appears.
39. Click the **Data Summary** button. Examine the list of hosts, sources and sourcetypes to get an idea of the types of data being indexed. The www* web servers will be the main ones you work with, along with the access_combined_wcookie sourcetype, from the web server logs.
40. Run the following search for the **last 60 minutes**:
    `index = itsi_summary`

    This is index stores KPI search results. Each of these events represents one KPI at a point in time, depending on the KPI schedule. There are also service health events for each service. Some of these KPIs are also broken out by entity. Examine some of the more relevant fields: `alert_level, alert_severity, entity_title,` and `kpi.`
41. Run the following search for the **last 60 minutes**:
    `index = itsi_tracked_alerts`

    This index stores notable events. Examine some of the relevant fields including `source` (the correlation search name), `alert_level`, `alert_value`, `entity_title`, `kpi`, `owner`, `status` and `urgency`.

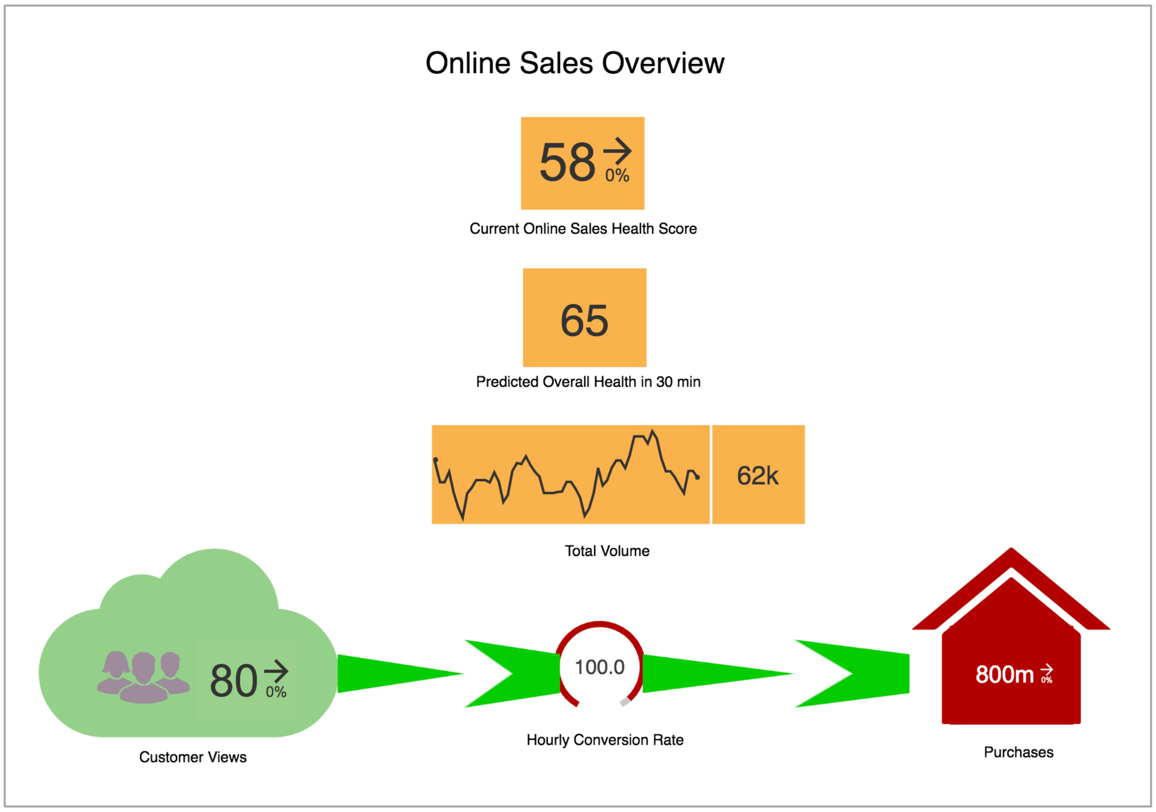
End of Lab Exercise 1

Lab Exercise 2: Implementing Glass Tables

Description

In this lab exercise, you'll create a glass table to visualize the status of the Online Sales service.

The online sales operations team wants to see something like this (challenge lab work included):



All of the graphics are available in the glass table editor icon set.

Glass Table Icon Title	KPI / Service Name(s) in ITSI	Notes
Overall Health	Online Sales ServiceHealthScore	
Customer Views	Views	
Purchases	Purchases	
Total Volume	Volume	
Hourly Conversion Rate	Purchases / Views (expressed as %)	Ad hoc calculation on the glass table

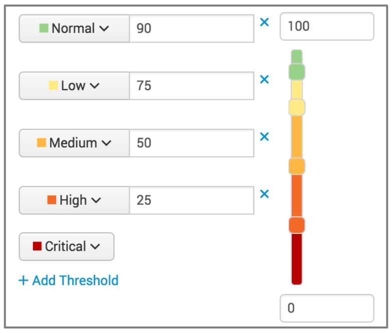
Task 1: Create an Online Sales glass table.

1. In the ITSI menu bar, click **Glass Tables** and click **Create Glass Table**.
2. Name it **# - Online Sales** (where # is your `student ID`), click **Shared in App**, then click **Create**.
3. Click **# - Online Sales** to open it in edit view.
4. First, add the generic icons (the cloud, group, and home).
5. Add text elements T to lay out the overall visualization.
6. Set the colors and sizes similarly to the drawing. (You don't need to match them exactly.)
7. Adjust the layering of the icons as needed. (The cloud widget needs to be behind the people widget).
8. Make sure you click **Save** before navigating away from the glass table editor.
9. Add all the KPI **widgets** from the left side with drag and drop.
10. Turn **Off** the Label Box for each widget and use your text boxes instead.

    You could edit the label field for each widget, but using the text boxes allows you to set layering, fill color, and move them any place that makes sense in the future as your Glass Tables evolve.

    It's important to click **Update** after modifying each KPI widget, and before selecting a different widget.
11. Change the **Volume KPI** widget type to **Sparkline**.
12. Change the Current Online Sales Health Score, Purchases, and Customer Views widget types to **Trending value**.
13. For the **Hourly Conversion %** widget, add an **Ad hoc Search** widget and configure its search as follows:

    ```
    sourcetype=access_combined_wcookie (action=view OR action=purchase)
    | chart count by sourcetype, action
    | eval rate = round(purchase / view * 100)
    | eval rate = if(rate > 100, 100 , rate)
    ```
14. Set the **Earliest time** field to **60 minutes ago** to make this widget evaluate the last hour's conversion rate.
15. Set the **Threshold Field** to **rate**.
16. For **Thresholds**, click **On**.
17. Under the **Thresholds** switch, click **Edit** and configure as follows. (It's easier to add a threshold first and then configure the existing dropdowns.) When you're finished, click the **Done** button.



	100
🟩 Normal	90
🟨 Low	75
🟧 Medium	50
🟧 High	25
🟥 Critical	
	0

18. Change the viz tip to **gauge** and click **Update**.
19. Click **Save**.
20. Click **View** to see the glass table from a user's perspective, and click **Edit** to return to edit view if needed.

splunk>

Optional Task 1: Set the color of an icon to change based on values thresholds.

21. If you closed it, open the glass table you just created. By now some of your glass table widgets may have changed color due to changes in your data. Notice that the icons do not change color.
22. Click the **Edit** button.

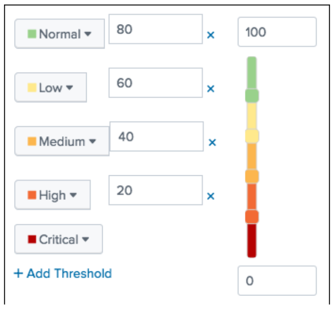
    We would like the cloud to change color along with the Customer Views widget. Let's verify which KPI represents Customer Views.
23. Click the KPI widget inside the Customer Views cloud. Under Configurations, notice the KPI is from the Online Sales service and is called Views.
24. Click the cloud icon.
25. Under Configurations, click the **Color Calculation** dropdown and click **KPI**.
26. In the **Service** dropdown, make sure **Online Sales** is selected.
27. In the KPI dropdown, select **Views**, click **Update** and **Save** your glass table. The cloud icon should now match its widget and will change colors with it as KPI thresholds are crossed.

Optional Task 2: Add an animation to your glass table.

28. If you closed it, open the glass table you just created.
29. Click the **Edit** button.
30. In the toolbar above your glass table, click and double-click the **arrow.gif** file you downloaded at the beginning of class.
31. Position the arrow over the **Customer Views** widget.
32. Resize (stretch) the arrow to extend the right edge over the Purchases widget.
33. Use the Layer control to send the arrow to the back.
34. Click **Save**.

Optional Task 3: Add a predictive analytics widget to your glass table.

35. Click **Dashboards > Predictive Analytics**.
36. Click the **Service** dropdown and select **Online Sales**.
37. Under the **Model** dropdown, select **LinearRegression**.
38. Click the **Open in Search** icon.
39. Copy the search string.
40. Return to editing the glass table.
41. Add an Ad hoc Search widget and use a text widget to label it: **Estimated Overall Health in 30 min**
42. In the **Search** field, paste the entire copied search string.
43. Change **Threshold Field** to **health_score**.
44. Configure the thresholds as follows:



Normal	80
Low	60
Medium	40
High	20
Critical	

Max: 100
Min: 0

45. Click **Update**.
46. Click **Save**.

End of Lab Exercise 2

Lab Exercise 3: Working with Episodes

Description

In this exercise, you'll use the Episode Review dashboard to examine and process issues.

Task 1: Examine episodes generated by your correlation searches and KPI alerts.

1. Navigate to **Episode Review**.

   You should see numerous episodes. One should be for your Web Farm service level, stating that only 3 servers are currently running, and the rest should be warning about a low conversion rate for specific product codes. The conversion rate is the rate at which visitors to the online store are purchasing products. It is based on a comparison of the Purchases and Views KPIs. If there are significantly more views than purchases in a 15-minute period, this type of episode is generated.

   Notice the time selector menu, which defaults to Last 24 hours. You can use this to control the time period for the list of episodes. For instance, if you want to look at the episodes from yesterday, you could click the dropdown, click Presets and click Yesterday. For now, leave it set to Last 24 hours.

   By default, episodes are sorted in reverse chronological order and are color coded to indicate the severity of each episode. You can sort the episode display by Title, Owner, or other fields with the sort control on the left of the view above the list of episodes.

2. Next to the Add Filter dropdown, in the search field, type **Low Conversion** and you should now see only the Low Conversion Rate episodes. Expand the time range if necessary.

   These episodes are being generated by a multi-KPI alert called **Low Conversion Rate**. It looks for instances where the **Views** KPI severity is normal or low, but **Purchase** KPI severity levels are high or critical.

3. Click to display the **View Settings**. You can alter the display options, such as which columns to display, their order, and how to display the notable severity. The Viewing Option Prominent, for example, fills the entire row with the severity color rather than just the edge. Try out some of these settings, but make sure you **leave Episode View set to On**.

4. Click the **Show Timeline** control.

   This control displays a timeline of all episodes in the current time period (defaults to 24 hours). You can use the zoom controls to locate specific episodes by time. Clicking on a timeline column filters the episodes to that unit of time and zooms to that unit. This view can be very useful to determine patterns of episode creation, such as every hour, or at a similar time each day, etc.

Task 2: Take ownership and create a custom view.

All class participants are working with the same set of episodes, so you'll each work with a different episode. Notice each episode description starts with **Product Code: N**. This is an arbitrary value added to the description in your lab environment to give you your own episodes to work. Note that new events are being added to the episodes every minute. This is arbitrary in our lab environment, to make sure we have plenty of episodes to work with and so we can see the effects of changes quickly.

5. From the **Episode Review** page, filter episodes to: **Low Conversion**
6. Locate an episode with a product code value matching your own `student ID`.

   For example, if you are assigned the user **analyst5**, in the Description column at the right, locate an episode associated with **Product Code: 5**.
7. Click the episode.

   A details panel appears on the right.
8. At the top of the details panel that just opened, click **Acknowledge**.

   This assigns the episode to you and changes its status to **In Progress**. You have now taken ownership of your episode, which is the first step in working the issue.

Task 3: Narrow your view using a filter to see only your episodes and save it.

Let's narrow your view to only your episode.

9. Select **Episode Review** to clear previously applied filters.
10. Select **Add Filter > Owner** and select your name.

    The list of episodes should now be filtered to the episode you acknowledged in the previous task. This could be a useful view—just the episodes you're working.
11. To save this filtered view, click **Save as**.
12. Name your view: {student ID} - Episodes for {your_name}
13. Click **Create**.

    Your view name is now displayed at the top of the page.
14. Navigate to the default service analyzer and then back to **Episode Review**.

    Note that your saved view is still in effect. This view will now open for you each time you navigate to the Episode Review page, until you change to a different view. (Your most recently accessed view will be the default to appear when you click Episode Review from the ITSI menu.)
15. At the top left, to the left of the title, click **>** to view a dropdown from which you can select any other saved view. (The last view you selected or viewed will become default).

    You won't do this, but to delete a view quickly, click **>**, mouse over the view name, click ···, then click Delete View.

Task 4: Work the issue.

16. Click your acknowledged episode to display the episode details.

    Note the severity (Medium) and status (In Progress).

    You may see a warning that some events in the episode are outside the selected time range or you may not have permissions to view some events.
17. Click the **Activity** tab.

    Notice your actions are being automatically logged for later use.
18. Return to the **Impact** tab.

    The description at the top is generated by the correlation search. You also see the service tile (with a link to its deep dive) and service topology (with a link to tree view service analyzer details) and impacted entities and related tickets. There may be other drilldown options (which may exist as customizations at your site).
19. Click **Analyze in Deep Dive** to see the Online Sales' service health score.

    You'll continue investigating this in the next module on deep dives.
20. Close the deep dive browser tab and return to the **Episode Review** page.
21. To add a comment, select the **Comments** tab, and type: **Investigating low purchase rate**
22. Click **Comment**.
23. Select the **Events Timeline** tab.
24. In the **Sort for:** dropdown, click **Root cause analysis** and hover over the exclamation point icon.

    It looks like the first concern was that purchases were critical while views were normal.
25. Below **Root cause analysis** and under **Event type** we see the "Indicator …" search. Click the link to that search to view the individual notables. From here you can edit columns or search for more detail. You can also access this view by clicking the **All Events** tab.
26. Click **Common Fields** > **all_info**. Here you can examine the frequency of the events you were just viewing.

Optional Task 1: Open your glass table to see snapshots of how the issue affects the business.

27. Make a note of the timestamp of your episode, including time zone.
28. Navigate to **Glass Tables**.
29. Open your **Online Sales** glass table.
30. Make a note of how it looks now.
31. In the upper right, click the **Now** button and substitute the current time for a the approximate time of your episode. (Don't forget to convert to your local time zone.) Adjust the **Now** dropdown to a little earlier and a little later. You may be able to see how the outage affected customer-facing business processes.

End of Lab Exercise 3

Lab Exercise 4: Using and Customizing Deep Dives

Description

In this lab exercise, you'll use a deep dive to diagnose a problem, and you'll also create a custom deep dive.

Scenario: For our case study, you'll walk through the stages of investigating a web server outage. Our Web Farm service monitors the health of our web servers—www1, www2, and www3. In the sample data, one of these servers has begun to exhibit an issue, which you'll diagnose with ITSI.

Task 1: Examine a potential problem by navigating from Service Analyzer to Deep Dive.

1. Navigate to the default service analyzer and examine the **Web Farm** KPIs. The **Storage Free Space** KPI should display an exclamation point (!).

   By default, you're seeing the **aggregate** for this KPI, which is measuring the storage space available on your web farm server's disk arrays.
2. In the service analyzer, select **KPI Value: Aggregate** and change it to **Max Severity**.

   Now the KPIs that are measuring individual entities will show the alert level for whichever entity has the worst alert level. Now you should see that storage free space is critical.
3. Click the **Storage Free Space** KPI to open the service's detail page, with the storage free space KPI selected. If prompted, leave the analyzer without saving.
4. In the service detail page, note that the **Storage Free Space** KPI is showing an elevated alert level, and on the right side, that one entity, **www2**, is in critical condition. Something's going on with storage availability on this server.
5. Mouse over the **Web Farm** service tile and click the checkmark.
6. Click the **Drilldown to Deep Dive** link to open the service's default deep dive.

Task 2: Facilitate your investigation by creating a custom Deep Dive.

First, you will save your own personal deep dive so you can alter it without affecting other people.

7. Click **Save as**.
8. Name the deep dive **{student ID} - Server Info**, select **Shared in App**, and click **Create**.
9. Select the boxes to the left of the lanes for **CPU Overutilization** and **Memory Free**, click the **Bulk Actions** dropdown and select **Delete**.
10. Change the time range to the **Last 4 hours** (under **Presets**) so you can examine the behavior in more detail.

    The alert level for the web farm became worse within the last hour or so. It's unclear whether www2 is involved, though. Let's break out the information by server.
11. Click the ⚙ icon for the **Storage Free Space** lane and select **Lane Overlay Options**.
12. For **Enable Overlays**, click **Yes**. Under **Selected Entities**, make sure **www1, www2,** and **www3** are selected and that "static" is selected. Click **Save**.

    Now the lane shows 3 trend lines for the 3 servers, but it's not easy to see which line maps to each server.

13. Click anywhere within the graph of the **Storage Free Space** lane and select **Add Overlay as Lane**.

    Now each server has its own lane and you can see the point in time that **www2**'s storage space KPI dropped to 0.
14. Click **Save**.

Task 3: Gather more information for an effective IT ticket by examining the OS Host Details dashboard for the web servers.

15. In the deep dive you just created, click anywhere on the graph of the www2 lane. A dropdown appears.
16. Click **OS Host Details**.

    Another browser tab opens. This is the detailed diagnostic for the **www2** server.
17. Since this is a storage-related KPI, click the **Storage** tab.

    You can see all the important details for the server, including filesystems, device IO statistics, storage used, read/write operations, and latency. This information comes from OS logs for performance and system monitoring.
    Note that some panels at the bottom of the display are blank, indicating there are no events to display. This tells you that the normal logging events are not being generated.
18. To confirm what Storage Free Space should look like, leave the www2 OS Host Details dashboard open and return to the deep dive and click anywhere on the graph for www1 and click OS Host Details. You should see all storage file systems listed with their size and free space.
19. Switch back to your browser tab displaying the **www2** OS Host Details dashboard, and select the **Splunk Events** tab.

    This shows you all events related to this server. If you navigate through a few pages of the events, you should see some events from the **df** and **iostat** source types that show errors for devices and the RAID bus. You will investigate these in more detail.

Task 4: Find more details by examining the underlying events in your deep dive so you can close the ticket. Save the deep dive for use in the future.

20. If you navigated away, return to your **# - Server Info** deep dive.

    You'll add a lane to examine the df and iostat source type errors.
21. Click **+ Add Lane** and select **Add Event Lane** from the dropdown and complete these fields:

    Title: Storage Error Events
    Event Search: index=* host=www* sourcetype=df OR sourcetype=iostat error
22. Click **Create Lane**.

    A new event lane is added. You should note darker colored event bars starting near where the **www2** storage failure occurred.
23. **Save** your deep dive.
24. In your new event lane, click one of the darker colored bars to open the events.

    You should see some error messages like "RAID BUS ERROR" and "ERROR device not found". You know now that the RAID bus failed at this time on this server.

    You could now submit an IT ticket using the server ID, failed device, time of failure. You could also keep this deep dive for the future for examining web server status, or ask you admin to create a Multi-KPI Alert.

Scenario: In this scenario, you build a new Deep Dive and a view-to-purchase conversion rate metric lane. Then you'll optimize access to your new Deep Dive by adding a Glass Table widget. Finally, you'll configure a custom drilldown action for the widget that takes you to the new deep dive.

Task 5: Build a view-to-purchase rate over time metric lane.

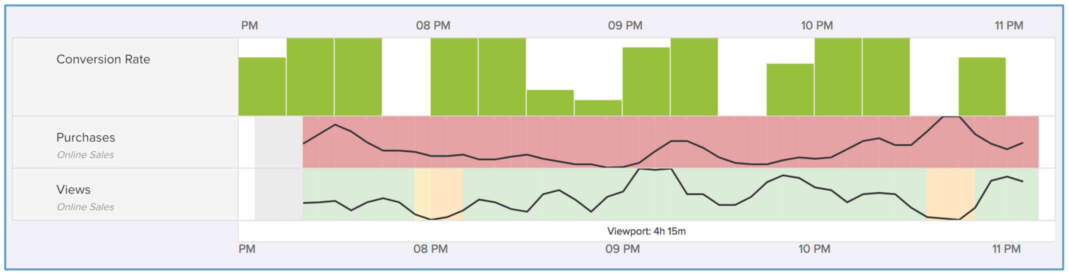
25. From the default service analyzer, click the **checkmark** for the **Online Sales** service tile and click **Drilldown to Deep Dive** to open the **Online Sales** default deep dive.
26. Click **Save** as to create custom deep dive named `{student ID}` **- Conversion Rate Deep Dive** and **Shared in App**.
27. Click **Create**.
28. Delete all lanes **except** the Purchases and Views lanes.
29. Add and configure a **Metric** lane as follows:

    Title: Conversion Rate
    Graph Type: Column
    Graph Color: Green
    Lane Size: Medium
    Search Type: Ad hoc
    Search:

```
sourcetype=access_combined_wcookie (action=view OR action=purchase)
| timechart span=15m count by action
| eval rate = round(purchase / view * 100)
| eval rate = if(rate > 100, 100 , rate) | fields - purchase,view
```

30. Click and drag the **Conversion Rate** lane to the top. Make sure **State Thresholds** are enabled for the **Purchases** and **Views** lanes.

    Remember that thresholding is not supported for metric lanes.
31. For Time Range, select **Last 4 hours**. Your deep dive should look something like this:



32. Click **Save**.

Task 6: Optimize access to the metric lane you built by adding a custom drilldown to your Hourly Conversion Rate widget.

33. Open your **Online Sales** glass table in **Edit** mode.
34. Click the **Hourly Conversion Rate** widget to display its configuration settings.
35. For Drilldown, click **On**.
36. Click the **Default** dropdown.
37. From the menu that appears, select **Saved Deep Dive**.
38. Click the blank dropdown menu that appears and select your **# - Conversion Rate** deep dive.
39. Click **Update** and **Save** the glass table.
40. Change to **View** mode and click the **Hourly Conversion Rate** widget. Your Conversion Rate deep dive should display.

End of Lab Exercise 4

Lab Exercise 5: Post-installation Check and Initial Configuration

Description

In this exercise, you'll examine an ITSI server that has just been installed and verify that it is ready to configure. You will also begin the process of identifying Splunk events and data models that you'll need to build the customer's services.

From this lab exercise onward, references to your "lab server" will be a new server for your own private use. Get the address for the server from your instructor and record it. The shared server you used for the preceding lab exercises will remain available for reference, but all new work will be on your private server.

LAB Server Address	34.244.1.172
Splunk Analyst UID/PWD	**admin** / splunk3du

Steps

Task 1: Log into Splunk as admin.

1. In your web browser, navigate to your assigned private lab server log in as **admin**.
2. Navigate to the IT Service Intelligence app.

   Note the Welcome dialog—nothing has been done here yet. As soon as you create a service, the Welcome dialog is disabled.
3. Close the **Getting Started** modal. Note that the service analyzer is empty.
4. Select **Service Analyzer > Service Analyzers**—there are no saved service analyzers yet.
5. Check the **Glass Tables** and **Deep Dives** menus—there are none yet.
6. Select **Search > Search** to open the search view in IT Service Intelligence.
7. Click **Data Summary**. Verify that data is being indexed. Examine some of the most common sourcetypes and hosts.
8. Run the following search over **All Time**:

   ```
   source=/opt/log/*
   ```

   These sources are generated from business lines in Buttercup Games. Source types like `access_combined_wcookie` (web server logs) will be very important. You should find that these sources go back 60 days or more. This will be important for ITSI implementation, as several features require a long baseline of events to analyze.
9. From the **App** menu, select **Manage Apps**.
10. Filter the list for "**ITSI**."

    You should see all the apps installed by ITSI (plus one for class lab data generation.) In particular, note the apps starting with "ITSI Module…". These are the ITSI modules that help us model services. You'll begin using them in the next lab exercise.
11. Navigate to **Settings** > **Indexes** and filter for **SA-IndexCreation.** These are the indexes installed with ITSI.

12. Navigate to **Settings > Data Inputs**.

    Note the extensive set of modular inputs associated with ITSI or IT Service Intelligence. None of these require admin interaction, except for the **IT Service Intelligence CSV Import**, which is used to discover and import entities—you'll learn about this later.

Task 2: Configure the Operating System module

13. Navigate to **Settings > Searches, reports and alerts**.
14. For **App** select **ITSI Module for Operating Systems (DA-ITSI-OS)**. These are the saved searches for the OS module.
15. Note the **DA-ITSI-OS-OS_Hosts_Search** saved search. This is the saved search the OS Module uses to do entity discovery for servers in your environment.
16. Click **Run** next to the above search. The search executes, but returns no results.
17. Note the search begins with the macro `` `itsi_os_module_indexes` ``. If you remove this macro from the search string and re-run the search, it should return results.

    The problem is the macro that ships with the module is hard-coded to look in specific indexes for events from the **nix** and **windows** add-ons, but these indexes are not guaranteed to exist at any given ITSI installation.
18. Navigate to **Settings > Advanced search > Search macros**.
19. Filter for the macro name **itsi_os_module_indexes**.
20. Click **itsi_os_module_indexes** to open the macro editor.
21. In the **Definition** field, add **"`OR index=main`"** to the existing search string. It should now look like this:

    ```
    (index=windows OR index=perfmon OR index=os OR index=main)
    ```
22. Click **Save**.
23. Repeat the steps above to run the **DA-ITSI-OS-OS_Hosts_Search** again, and confirm that results are now returned by the saved search. The module will now be able to discover entities in your environment.

End of Lab Exercise 5

Lab Exercise 6: Designing Services

Description

You'll work together to define one of the services for Buttercup Games. You will initially focus on defining the service to support the online sales operations team. Then, you'll build some base searches for the data you need in your services.

Task 1: Define the service name and the KPI requirements for the business service.

1. Based on statements of the customer sales ops team, define a service and general KPI requirements.

   From the customer:

   We want a status board that updates every minute and shows the last 15 minutes' overall efficiency of our online sales, broken down by the number of times products has been viewed or bought, and the total volume of web content our customers viewed. We are really focused on the number of items bought—this is a key factor. For online sales, the purchases are the important thing. The purchases KPI should be twice as important as the others. Volume is significant, but less critical than other items.
2. What shall we call the new service? **Service Title:** Online Sales
3. Now, identify the KPIs based on the customer statement. Remember, that service health scores are created automatically so you don't need to define them here. For each KPI, fill in the following grid, using information from the customer statement:
   - Fill in a quote from the customer statement that establishes the KPI
   - Give the KPI a name
   - Fill in frequency and time span to match their needs
   - On a scale of 1 to 10, identify the importance. Remember that 5 is the default, and 11 means the service health score cannot be better than the KPI's alert severity.
   - For threshold type, enter "low" if lower values are desirable for this KPI; "high" if the KPI is better with higher values, or "mid" if either extreme low or high values could be bad.

KPI Name	Requirement	Freq.	Time Span	Imp.	Threshold Type
Service Health Score	Overall efficiency	1 minute	15 min	--	--
Views	Number of times a product has been viewed	1 minute	15 min	5	high
Purchases	Number of time product has been purchased	1 minute	15 minute	10	critical

Total volume	total volume of web content customers viewed	1 minute	15 min	5	high

4. Based on the customer requirements, which if any KPIs need to be split by entity?

Task 2: Design the web farm service.

Use the same approach you used when designing the Online Sales service to plan a new service for the web farm. Do not implement it in ITSI yet—this is just planning.

5. Use this input from the IT team for requirements:

   The most important thing for us is our website health. How many errors are being generated? Errors in the storefront web app cause huge problems for us. Also, what is the average usage of CPUs, memory and disk space per machine in the web farm? We'd like to see this update every minute and show the last 15 minutes' data. And we want to be alerted if the number of servers in the web farm falls below our service level.

6. First, define the web farm service name: **Service Title:** Web Service Health
7. Next, document the KPIs for this service, using the information you gathered during the initial data audit (in lab exercise 1):

KPI Name	Requirement	Freq.	Time Span	Imp.	Threshold Type
Service Health Score	Website health	1 minute	15 min	--	--
Errors	How many errors are being generated	1 minute	15 min	5	low
Cpu Utilization	Average CPU utilization	1 minute	15 minute	4	low
Memory Utilization	Average MEM utilization	1 minute	15 minute	4	low
Disk Space Utilization	Average DISK utilization	1 minute	15 minute	4	low

8. Based on the customer requirements, which if any KPIs need to be split by entity?

Yes, all the kpis for web service can be split into entities as servers

End of Lab Exercise 6

splunk>

Lab Exercise 7: Data Audit and Base Searches

Description

From the previous lab exercise, we know the services we're building and the KPIs required, along with some of their settings, like scheduling, importance and threshold type. Now we want to complete the service implementation plan by filling in the source of the data each KPI will display: What events are we looking for, which fields in the events are relevant, and how will we summarize the data (count, sum, average, etc.).

Task 1: Identify useful kinds of events.

Based on discussions with the customer (from the slides), you know they use an Apache-based web storefront that logs NCSA web events to the **access_combined_wcookie** source type.

System	Related Source Types or Modules
NCSA logs showing customer interaction with the website, as well as error events	**access_combined_wcookie**
Server performance metrics, including CPU, memory and disk space data	**Operating Systems** module, **CPU**, **Memory** and **Storage** KPI groups

And their statement of requirements:

"We want a status board that updates every minute and shows the last 15 minutes' overall efficiency of our online sales, broken down by the number of times a product has been viewed or bought, and the total volume of web content our customers viewed."

Use the following breakdown of customer expectations to map out the source types and fields you should focus on.

Note that you do not need to fill in your answers in the tables below—you can use your own scratch paper or text editor file to do that.

Online sales operations requirements:

Statement	Source Type & Field / Module & KPI
"Overall efficiency of our online sales…"	Service health score
"Number of times a product has been viewed…"	access_combined_wcookie field = action, value = view
"Number of times a product has been bought…"	
"Total volume of web content…"	

IT team requirements:

Statement	Source Type & Field / Module & KPI
"How healthy is the website…"	Service health score
"…are errors being generated…"	
"…what is the usage of CPU…"	Operating System module CPU Utilization: % template
"…what is the usage of memory…"	
"…what is the usage of disk space…"	

1. Fill in the empty cells in the tables above.

- Use Splunk search commands to examine the available fields in the source types and identify the field(s) that contain(s) the information needed to fulfill the stated requirement.
- Use **Configure > Modules** to examine the ITSI Module for Operating Systems module and identify the KPI templates that can be used to meet your requirements.
- For module-based KPIs, use the module's **Run Search** feature to verify the module is discovering events (hint: Services tab, base searches sub-tab, link under search string.)

Task 2: Build base searches.

Now that you know what kind of KPIs you'll be creating, you should consider using base searches to improve performance. In the real world, base searches improve performance by producing multiple values for KPIs with one search. In this case, you won't be realizing true performance improvement, because you're only creating three KPIs, and each will need to be in its own use case because each uses a different set of source events. Each base search needs to be defined with a single source (discrete set of events). But, you can then define multiple metrics, or results, to gather from the set of events.

In this project, the source events all come from the same data source (sourcetype=access_combined_wcookie), but require three different filters:

- Volume of all events (no filter)
- View events (action=view)
- Purchase events (action=purchase)

In this task, you only create one metric in each base search—so you won't realize any performance improvement. However, you are preparing for the creation of additional KPIs in the future. For each new KPI, based on the same selection of events, you can add more metrics, which will not increase the total number of searches for the Splunk scheduler to manage. In addition, you can revise the base searches as needed, and the changes are automatically propagated to all dependent KPIs immediately.

You'll only be building base searches for the online sales service, since the web farm service KPIs will depend mainly on module KPIs.

2. Make sure you are logged in to your private lab server as **admin**.
3. Select **Configure > KPI Base Searches**.

   Note there are numerous pre-built base searches—these are defined in the default modules that ship with ITSI. Note the naming conventions—the module name is the prefix. You should use a naming convention so you can find your base searches easily.
4. Click **Create KPI Base Search** and enter the title: **BCG: All Web Traffic**
5. Click **Create**.
6. In the search field, replace the asterisk with the following search:

   `sourcetype=access_combined_wcookie`
7. Click **Run Search** to make sure you get results.
8. Configure the following parameters:

   KPI Search Schedule: Every Minute
   Calculation Window: Last 15 Minutes
   Monitoring Lag (seconds): 30
   Split by Entity: No
   Filter to Entities in Service: No
9. Click **Add Metric** and configure the following parameters:

   Title: total_bytes
   Threshold Field: bytes
   Service/Aggregate Calculation: Sum
   Fill Data Gaps with: Last available value
10. Click **Add**.
11. Click **Save**.

    At this point, you're done with this base search for lab purposes. You don't have any other metrics you need for this set of events. In the real world, you might also add metrics for the total count of events, distinct count of sessions, average of duration, etc. All of these would be generated each time the base search runs, and would be available for use in future KPIs, without increasing scheduler load as much as individual searches for each KPI would.
12. Create the base search for the Views KPI, using the following parameters:

    Title: BCG: Web Views
    Search: `sourcetype=access_combined_wcookie action=view`
    KPI Search Schedule: Every Minute
    Calculation Window: Last 15 Minutes
    Monitoring Lag (seconds): 30
    Split by Entity: No
    Filter to Entities in Service: No

    Metric parameters:

    Title: view_count
    Threshold Field: _time
    Service/Aggregate Calculation: Count
    Fill Data Gaps with: Null values
    Threshold level for Null values: Unknown

13. Create the base search for the Purchases KPI using the following parameters:

    Title: BCG: Web Purchases
    Search: `sourcetype=access_combined_wcookie action=purchase`
    KPI Search Schedule: Every Minute
    Calculation Window: Last 15 Minutes
    Monitoring Lag (seconds): 30
    Split by Entity: No
    Filter to Entities in Service: No

    Metric parameters:

    Title: purchase_count
    Threshold Field: _time
    Service/Aggregate Calculation: Count
    Fill Data Gaps with: Null values
    Threshold level for Null values: Unknown

Consolidating base searches

You might be asking, “Why build three base searches, each with only one metric, when the search strings are so similar?” And you’d be right. Given the small scale of our lab environment, these are not very realistic. You might be trying to think of a way to consolidate the three base searches into a single base search that uses three different metrics.

You could create one base search with the search string:

```
sourcetype = access_combined_wcookie | eval is_{action} = 1
```

This generates additional fields in each event based on the value of the action field. If **action=view**, then **is_view=1** is created. Now, you can create three metrics: **sum(bytes)**, **sum(is_view)**, and **sum(is_purchase)**. This will scale better, and you can produce individual KPIs for the Purchase and View actions, using a single base search!

End of Lab Exercise 7

Lab Exercise 8: Implementing the Business Service

Description

Using the information from the previous lab exercise, implement the online sales service for Buttercup Games.

Task 1: Create the Online Sales service.

1. Navigate to the IT Service Intelligence app.
2. Select **Configure > Services** and click **New Service > New Service**.
3. Complete the dialog as follows:

   Title: Online Sales
   Description: Monitor online sales
   Team: Global
   Select **Manually add service content**
4. Click **Create**.
5. You won't be adding entities to this service at this time. Click the **KPIs** tab.

   You'll be adding the three KPIs from your plan. You'll start by creating the **Views** KPI.
6. Click **New** and select **Generic KPI**.

   The 6-step KPI modal appears.
7. In step 1, set the title of the KPI: **Views**, with the description: **Online Sales views by customers**
8. In step 2, click **Base Search**, and select the **BCG: Web Views** base search.
9. Select the **view_count** metric.
10. Click **Next** until you get to step 5: Optional Setup.
11. Backfill the KPI for the **Last 14 days**.
12. Click **Finish** to skip the remaining steps (we will configure the KPI's thresholds in a subsequent lab exercise).
13. Click **Save**, then **Save and Enable** to save the current service configuration and enable the new service.

    Note that in some cases you might want to leave a new or partially completed service disabled, but you will enable this one now to save time and so you can begin to immediately generate KPI events for testing.

    Congratulations, you have created your first KPI! The 2 week backfill started running when you saved and enabled the service; you'll see a system message in a minute or two when it completes.

    You can now clone the **Views** KPI to create the **Purchases** KPI because it is almost identical.
14. On the KPIs tab, click **Clone**, select the **Online Sales** service, and the **Views** KPI.
15. Click **Clone**.

    The **Views (copy)** KPI is created.
16. Click this KPI and make the following changes:
    - Change the title to **Purchases** and click **Done**, then change the description to "**Number of purchases**" and click **Done.**
    - Under **Search and Calculate**, click **Edit (for Source)** and change the selected base search to **BCG: Web Purchases** and the metric to **purchase_count**.

17. Click **Finish**, and click **Save** to save the new KPI. This will start the backfill for this KPI also.
18. Create the **Volume** KPI now, using the same process you used for the **Views** KPI, except as follows:

    Title: Volume
    KPI Source: Base Search
    Base search: BCG: All Web Traffic
    Metric: total_bytes
    Enable backfill: checked
    Backfill period: Last 14 days

    Do not set any thresholds now.
19. Click **Finish**.
20. **Save** the service. The final backfill starts.

    Note that while we have configured backfill for all KPIs, this is not always required. Because we will be configuring adaptive time thresholds and also using some predictive analytic tools soon, we want a baseline of KPI data to work with. At production sites, there may be enough time between KPI creation and later analytical work that backfill might not be needed.

    Also, if you ever need to "re-backfill" a KPI, you can do so by first cloning it, then deleting the old KPI and re-naming the clone to replace the original. The "new" version of the KPI can now be backfilled again.

Task 2: Configure service health importance.

Now you'll configure the service health score importance for each of our KPIs.

21. Click the **Settings** tab.
22. Use the sliders to set the importance for each KPI as previously identified (lab exercise 6).
23. Click **Save**.
24. Try using the Simulated Severity controls to test different alert level combinations and see what the effect is on the service health. This is a simulation only—it has no effect on the service.

Task 3: Inspect the itsi_summary index.

First, we can check to make sure our KPIs are being generated in itsi_summary. We can also create a new service analyzer.

25. Open a search window and search for all events in the **itsi_summary** index over the **Last 30 days**. Because you backfilled your KPIs for 14 days, you should see thousands of events.
26. Examine the values for the **kpi** field. These are the titles for your KPIs, plus the service health score.
27. Examine the values for the **alert_value** field. These are the actual values the KPI searches detected.
28. Examine the values for the **alert_severity** field. These are the labels for the state of the KPI. The only threshold set is the default, **normal**.
29. Note that the service name is not shown—the service is identified by an ID instead of a name.

Task 4: Create a Service Analyzer.

By default, there is one global service analyzer, which is displayed as the default page for the ITSI app. However, if you click the **Service Analyzers** menu, none are listed.

30. Select **Service Analyzer > Default Service Analyzer**. The default ITSI service analyzer showing your new Online Sales service is displayed. (it sometimes takes a few minutes for a new service to show.)
31. Click **Save as…** and type **Buttercup Games Services** for the service analyzer title.
32. Select **Shared in App** for permissions.
33. Click **Create**.
34. Select **Service Analyzer > Service Analyzers**.

    Note your new service analyzer is displayed here. In the future, you can make changes to this service analyzer, such as filtering which services are displayed or how many KPIs are displayed.
35. Select **Edit > Edit Permissions** and note that you can configure which roles have access to this analyzer. Along with other Service Intelligence UI views like glass tables and deep dives, you can use roles-based permissions to configure tools based on user needs.
36. Modify the permissions for your new service analyzer. Configure it so only **admin** users can modify it.
37. Open the **Buttercup Games Services** service analyzer.

    Notice that all the tiles are showing a status of green. This is because you have not configured thresholds yet—you'll do this in the next lab.

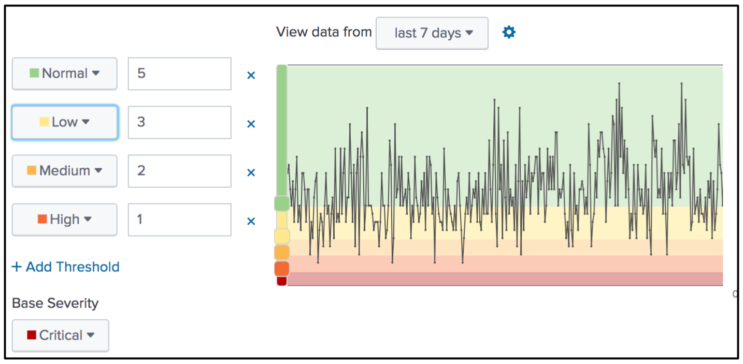
Lab Exercise 9: Thresholds and Time Policies

Description

In this lab exercise, you'll configure thresholds for the Online Sales service.

Task 1: Configure a basic threshold.

1. Select **Configure > Services** and open the **Online Sales** service.
2. From the **KPIs** tab, select the **Views** KPI, and expand the **Thresholding** section.
3. Use the **View data from** … link to examine the pattern of the source events over the last 7 days. Use the **Add Threshold** link to define five thresholds as per the image below. Adjust the sliders for the thresholds until most of the data for the last 7 days is normal, but approximately the lower 10% make up the highest alert levels. The exact values of the threshold alert levels do not matter for this exercise, but set high to 1, so that critical only occurs if there are zero views during the 15 minute period. The following screenshot is an example—your actual thresholds will likely be different.



Normal	5
Low	3
Medium	2
High	1
Base Severity	Critical

Example thresholds for Views

4. Click **Save**.
5. Repeat the steps above to configure the thresholds for the **Purchases** KPI. Remember to save the service after making changes.

Task 2: Configure adaptive thresholds.

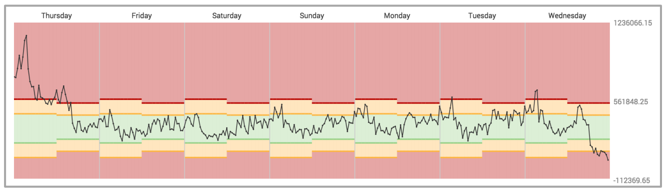
6. Make sure the backfill job for Volume is complete—you should see a system message at the top of the Splunk Web page.
7. Select the **Volume** KPI and expand the **Thresholding** section.
8. Enable both **Time Policies** and **Adaptive Thresholding**.
9. Set the **Training window** to **14 days**.

   The **Preview Aggregate Thresholds** display appears and shows the alert values for this KPI over the last 7 days. This graph shows the KPI values that were backfilled.
10. Hover your mouse over the preview panel to assess the data ranges. Check the highest and lowest values.
11. Expand the **Configure Thresholds for Time Polices** section. Note that everything is "normal" right now—you have not added any time policies yet.
12. Scroll up to the top of the **Thresholding** section, click the **Select a thresholding template** dropdown and select the **AM, PM (adaptive/stdev)** policy. You'll be prompted to apply the template and discard existing threshold settings—click **Apply**.

13. Expand the **Configure Thresholds for Time Policies** section and you'll see there are now two additional time policies in the left panel—*Everyday, 12AM-12PM* and *Everyday, 12PM-12AM*. But the adaptive thresholds are not set—it's using a generic standard deviation pattern at the moment.
14. Click **Apply Adaptive Thresholding**.

    Note: You might need to scroll down inside the **Aggregate Threshold Values** pane to see the button.

    The adaptive threshold dataset is now applied. You should see something like this in the **Preview Aggregate Thresholds** panel:



15. Click **Save** to make sure your changes are saved.

Task 3: Create a custom threshold template.

To extend our understanding of thresholding, let's see what we need to do to create our own special threshold template. You'll create a template that applies the adaptive standard deviation policy Monday through Friday, but exempts the weekends.

Before you do this task, you should make sure you have not altered your timezone settings in the administrator's account preferences. It should be set to default(server). This setting keeps your view synchronized with the indexer time (UTC) to avoid confusion while setting time periods.

16. In the thresholding section of the **Volume** KPI page, select **Set Custom Thresholds**.
17. Expand the **Configure Thresholds for Time Policies** section.
18. Select the policy for **Everyday, 12AM-12PM**.
19. Click the ⚙ icon next to **Aggregate Threshold Values** to edit the time policy settings.
20. Uncheck **Su** and **Sa**, then click **Done**.
21. Repeat the above steps for the **Everyday, 12PM-12AM** policy.

    At this point, you've removed the weekend from the policy list. Now the **Default** policy will apply to KPIs during that time interval. You could go on to customize the default policy, but instead you'll add a policy for the weekend.
22. Click **+ Add Time Policy**.
23. Enter the title **Weekend**, and check only **Su** and **Sa**.
24. Set **Duration** to **24 hours**.
25. Click **Add**.
26. For the new **Weekend** policy, set **Base Severity** to **Info**.

    This KPI will no longer affect the service health score during the weekend.
27. Scroll up to the top of the Thresholding section, click the **Save as template** link, and entitle the template **BCG: adaptive/standard deviation, no weekends**, then click **Save** to save the new template.
28. **Save** the service definition.

29. Navigate to **Configure > KPI Threshold Templates** and type "**BCG**" into the filter.

    Note your template is now saved for future use.

30. Open your threshold template.

    Note that you can edit it here. If you make changes to the threshold template, it would affect all KPIs using that template.

End of Lab Exercise 9

Lab Exercise 10: Entities and Modules

Description

In this lab exercise, you'll add entities and a new technical service based on module KPIs.

Task 1: Break online sales purchases down by product category.

If you look at the `access_combined_wcookie` source type fields, you'll note a `categoryId` field that identifies the type of product involved in a transaction. The sales team would like to be able to monitor sales per category, so you will need to alter the base search for purchases to split by entity, where the `categoryId` field identifies the entity. You'll use the categories as an abstract entity. For this use case, you don't need to formally import entities—a simple entity split does the job.

1. Navigate to **Configure > KPI Base Searches** and select the **BCG: Web Purchases** base search.
2. Change **Split by Entity** to **Yes**.
3. Change the **Entity Split Field** to **category`Id`**.
4. Edit the **purchase_count** metric.
5. Set **Entity Calculation** to **Count** and **Service/Aggregate Calculation** to **Sum**.

   This counts the purchase events for each category, then sums the category counts as the aggregate value of the KPI.
6. Click **Done**, then **Save**.
7. A warning message displays; click **Save** to complete the save operation.
8. Navigate to **Configure > Services** and edit the **Online Sales** service.
9. Open the **Purchases** KPI and expand the **Thresholding** section.
10. Click **Per Entity Thresholds**. This displays a new threshold map.
11. The base severity should be set to Normal. Change it to **Info**. You won't be changing the alert severity per category.
12. **Save** the service.
13. Navigate to the default service analyzer and click the **Purchases** KPI tile.

    You should now see a list of entities for the KPI.
14. Create a new deep dive using the following parameters:

    Title: Purchases by Category
    Permissions: Shared in App
15. Edit the deep dive.
16. Change the time range to **Last 24 hours**.
17. Add the **Purchases** KPI from the **Online Sales** service.
18. Edit the **Purchases** lane settings and select **Lane Overlay Options**.
19. Enable lane overlays, and select all the categories (using Static mode), other than NULL.
20. Click **Save**.

    The lane now shows multiple trend lines for the selected categories.
21. Click in the lane, and select **Add Overlay as Lane**. The categories are now added as individual lanes.
22. **Save** the deep dive.

Task 2: Create entities for the Web Farm service.

Now you'll create the service for the Web Farm. In preparation, you'll create the entities for the web farm servers.

23. Navigate to **Configure > Entities**.
24. Select **Create Entity > Import from Search**.
25. Select **Modules > ITSI Module for Operating Systems**.

    This search is loaded:

    ```
    | savedsearch DA-ITSI-OS-OS_Hosts_Search
    ```

    The **DA-ITSI-OS-OS_Hosts_Search** saved search uses the **tstats** command that retrieves host names from the **HOST_OS Inventory** data model object. This search references the **Inventory** object for events that have **version** or **vendor** product values. These usually come from **Splunk_TA_nix** or **Splunk_TA_windows**, so make sure those are configured when you install on prem.
26. Click the 🔍 button to run this search.

    Note that several columns of information are returned for each server, and that all servers in the environment have been returned. In practice, you will usually want to fine-tune this.
27. Add the following to the existing search and re-run:

    ```
    | search host=www* OR host=supp*
    ```

    Now you should only see our webserver hosts—the www* servers are for the public storefront website, and the supp* servers are for our support site.
28. Click **Next** to display the **Specify Columns** page. In the **Import Column As** column, select **Entity Title** for the **host** field, and select **Entity Information Field** for the other fields that have sample values. Leave the rest set to **Do Not Import**.

    Note that **Update Existing Entities** is the default action for Conflict Resolution. This is usually the best option. It adds new entities, but for existing entities it only adds new field values, but does not alter any existing fields.
29. In the Preview section below, click **Entities to be Imported** to see the entity titles and information that will be imported.
30. Click **Import**.

    This creates the new entities. The confirmation page is displayed. Note that this search can now be saved as a recurring import so you can repeat it in the future, including on a schedule to automatically add new entities that are created in your environment. You can also edit the search to fine-tune it to select specific hosts, and you can add more searches later to find different types of hosts.
31. Click **View All Entities** and note that the entities for the web servers have been added.

Task 3: Create the Web Farm service.

Based on your design, the IT team wants a Web Farm service to monitor the health of the web servers. For this service, you'll save time by using the OS Hosts Monitoring service template, which automatically creates the KPIs for you. Then, you'll add the generic KPI that the service design calls for—the Errors KPI.

32. Navigate to **Configure > Services** and click **New Service > New Service**.

    Title: Web Farm
    Description: Monitor web site servers
33. From the list of prebuilt KPIs, select the **OS Hosts Monitoring** category on the left.
34. Make sure only these three KPIs are checked:
    - CPU Utilization: %
    - Memory Free: %
    - Storage Free Space: %

    These are the best "fits" in the list of pre-built KPIs for the KPIs in your plan.
35. Click **Create**.

    The new service is created and you are taken to the service edit window, with the **Entities** tab active.

    The Entity Rules form for the service appears. This form controls which entities are part of this service. By default, there are two rules ANDed together: **host does not match blank**, AND **itsi_role matches operating_system_host**.
36. Change the first rule so that **host** – **matches** – **www*** and press [**tab**] to cause the rule to update.

    Only the three **www*** servers should appear in the list below.
37. Click **Save**, then **Save and Enable**.
38. On the **KPIs** tab, select the **Memory Free: % KPI** and examine the **Search and Calculate** settings. Note they are split by host (all the entities in the service, the three web servers) and are also filtered to only the entities in this service.

    The **OS Hosts** KPIs are set to update every minute and calculate their scores based on the search criteria over a 1 minute range. You could alter the time span for each KPI to match the requirements, but to save time you can leave these at default.
39. Click **Edit** for the **Source** section.

    Note that this KPI is set to use the **Performance.Memory** base search from the **DA-ITSI-OS** module.
40. Click the **Edit Base Search** link and inspect the base search in a new browser tab.

    Note that the search expression is:
    (`itsi_os_module_indexes` tag=oshost tag=performance tag=cpu)

    This search uses these tags to identify the correct events that correspond to the **Memory** object in the **Host_OS** data model.
41. Close the browser tab to return to the service editor, and click **Cancel** to close the KPI editor.
42. To add the **Errors** KPI, click **New**, scroll down to the **Templates** section near the end, and select **Count Based Ad Hoc KPI**.

43. Update the following parameters for Steps 1-3:

    Title: Errors
    Description: Errors in web farm.
    Search: `sourcetype=access_combined_wcookie status>=400`
    Threshold Field: _time
    Split by Entity: Yes
    Filter to Entities in Service: Yes

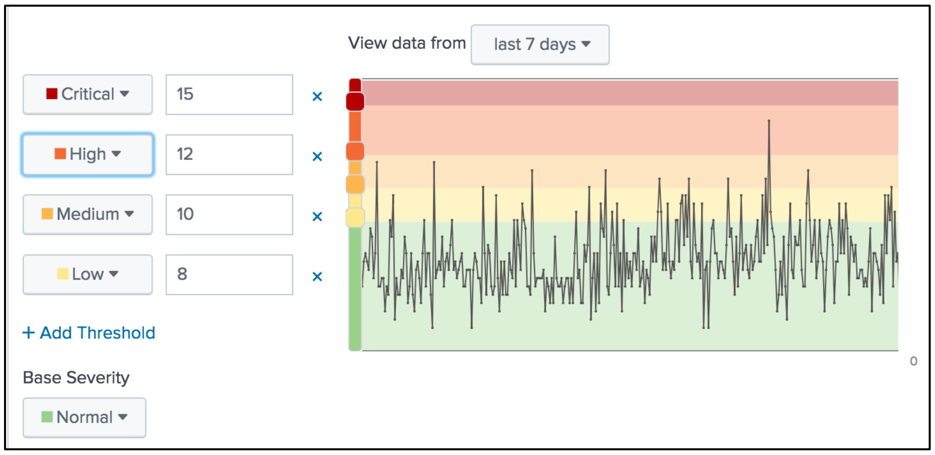
    The **host** field is automatically added to the **Entity Filter Field**. This is the field in the event that you will split by. This generates KPIs for each server in the service—all three "www*" servers in this case.
44. On step 3, click the **Generated Search** link to expand the search window and examine the search generated so far. This search becomes the KPI saved search when you're done.
45. On step 4, configure the Calculation Options as follows:

    KPI Search Schedule: Every Minute
    Entity Calculation: Count
    Service/Aggregate Calculation: Sum
    Calculation Window: Last 15 Minutes
    Fill Data Gaps with: Null values
    Threshold level for Null values: Unknown

    The explanation should read:

    "Every 1 minute take the count of host for each entity as the entity value then take the sum of all entity values as the service/aggregate value all over the last 15 minutes. Fill gaps in data with Null values and use a unknown threshold level for them."
46. On step 5, the **Unit** will show up next to the KPI value in numerous places. Clear the default value "ct." Leave the **monitoring lag** at 30 (default).
47. Backfill the KPI for the **last 7 days**.

    On step 6, the threshold configuration appears. Note that it has arranged a "low normal" threshold pattern automatically.
48. Set thresholds as follows (your sample data may be different):



🟥 Critical	15
🟧 High	12
🟧 Medium	10
🟨 Low	8

Base Severity
🟩 Normal

49. Click **Finish**, then **Save**.
50. Click the **Settings** tab and configure the **Errors** KPI as a minimum health score KPI (importance of 11).
51. Click **Save**.

End of Lab Exercise 10

Lab Exercise 11: Templates and Dependencies

Description

In this exercise, you will first address the known association between Online Sales and the Web Farm service. Then, address a new requirement thrown at us: The customer support team has learned about ITSI and wants us to implement a service for them!

Task 1: Make the Online Sales service dependent upon the Web Farm service.

1. Navigate to **Configure > Services** and edit the **Online Sales** service.
2. Select the **Service Dependencies** tab and click **Add dependencies**.
3. Select the **Web Farm** service, and then the **ServiceHealthScore** KPI.
4. Click **Done**. The Web Farm's service health score is now a dependency of the Online Sales service.
5. Click **Save**.

   Scenario: New service

   The IT support team has a customer support portal running on a set of servers. They mainly want the same level of monitoring as the web IT team. In fact, the new Support service would be a copy of the web farm service, but using different entities for the web servers. Looking ahead, you can see that there are other web resources that would likely also be added in the future, so this is a good time to implement a Web Farm service template.

Task 2: Create a new service template.

First, you'll rename the existing web farm service to make it more identifiable.

6. Navigate to **Configure > Services**, and click **Web Farm** to edit the service.
7. Change the name of the service to: **Storefront Web Farm**
8. Click **Done**, then **Save**.
9. Navigate back to **Configure > Services**.
10. For the Storefront Web Farm service, click **Edit > Create Service Template**.
11. Enter the title: **Web Farm Service Template**
12. Click **Create**.

    You have created the template, and the template editor UI is opened. You can see that it contains the KPIs from the Storefront Web Farm service.
13. Select the **Linked Services** tab.

    Notice the **Storefront Web Farm** service is a linked service. Any future changes you make to the **Web Farm Service Template** affects the Storefront Web Farm and any other linked service(s).
14. Select the **Entities** tab; note the entity rule from the Storefront Web Farm is replicated here. You don't want this template to select `www*` hostnames for all possible web farm services, so change the **matches www*** rule to **matches a value to be defined in the service**.
15. Click **Save**, and on the **Save Service Template** modal, note first that you have options to overwrite linked services configurations (although you will leave these off) and also to schedule the changes for later (but you'll push it now to save time.)
16. Click **Save**.
17. Open the **KPIs** tab.

18. Select the **Errors** KPI and expand the **Search and Calculate** section.
19. Click **Edit**.
20. Notice the KPI is now using a new base search and metric, which was automatically created when you created the service template.
21. Click **Cancel**.

Task 3: Create a new service using a service template.

Now you'll create a new service using the Web Farm Service Template.

22. Navigate to **Configure > Services** and select **New Service > New Service**.
23. On the Create Service modal, enter the title **Support Web Farm**, and select **Link service to a service template**.
24. For Link to template, make sure Web Farm Service Template is selected.

    Note the list of entity rules, KPIs and other linked services.
25. Click **Create** to create the new service.

    The service editor UI is opened with the **Entities** tab selected. Note the rule has the blank value for matching host aliases and a yellow warning icon indicating the missing value.
26. Enter **supp*** for the **matches** value and press **[TAB]** to rerun the entity rule preview.

    You should see a set of matching host names for **supp*** hosts in the preview list at the bottom.
27. Click **Save**, then **Save and Enable**.

    Your new service is up and running! In a few minutes, you can check the default service analyzer to confirm the service is active and see the initial values for its KPIs.

End of Lab Exercise 11

Lab Exercise 12: Anomaly Detection and Predictive Analytics

Description

In this lab exercise, you'll add a new KPI to the Support Web Farm and enable anomaly detection. You'll also create some predictive analytics models for analysts to use on the Predictive Analytics dashboard.

Task 1: Add the ticketing load metric.

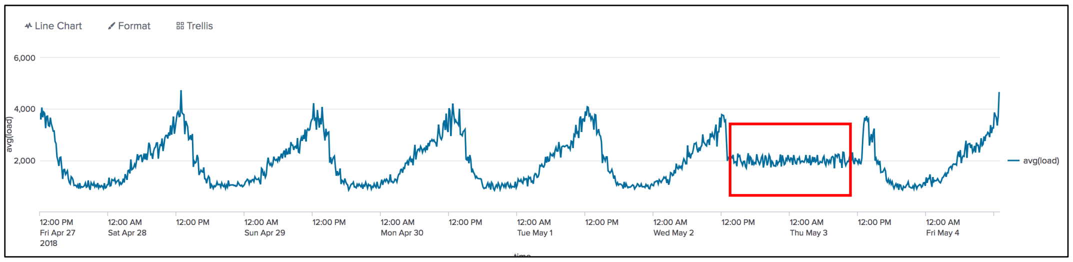
The support team has an additional metric to track: the load on their ticketing system. They want to be alerted if the load behaves "abnormally," so you'll add this KPI to the service and enable anomaly detection.

First, evaluate the source events. Talking to the support team, you learn that the required sourcetype is **support**, and in the support events, a field named **load** contains a value expressing the total load on the ticketing system.

1. Run this search over the **Last 7 days** to examine the events:

   ```
   sourcetype=support | timechart span=10m avg(load)
   ```

2. Viewing the results as a line chart visualization, you should see something like this:

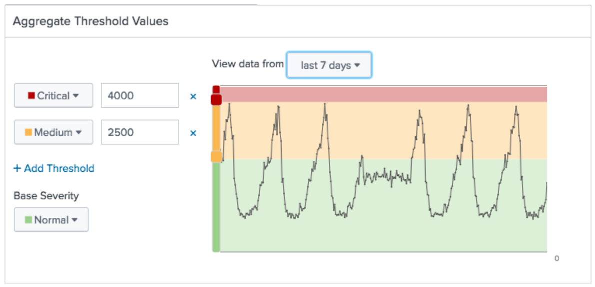


   Since these events have a regular pattern, it's a very good candidate for anomaly analysis. If the KPI deviates from this expected pattern, you'll get an alert. However, note the area in the red box (this should be in your data too—about 1 day before the start of your class). This looks odd; a day where there was no regular high-low cycle. The load level is not too high nor too low, so it wouldn't be caught by simple thresholding. But it is anomalous—it indicates something is interfering with your measurement of the ticketing system. You should be on the lookout for it in the future.

3. Run a search over all time on the **anomaly_detection** index. Note that it is empty—no anomaly detection is configured.

4. Edit the **Support Web Farm** service, and add a new generic KPI with the following settings (accept default values for all other parameters):

   Title: Load
   Search: `sourcetype=support`
   Threshold field: load
   Split by Entity: Yes
   Filter to Entities in Service: Yes
   KPI Search Schedule: Every Minute
   Backfill: checked
   Backfill period: Last 30 days (to provide a long baseline for analysis)
   Aggregate Thresholds:


Aggregate Threshold Values

View data from last 7 days

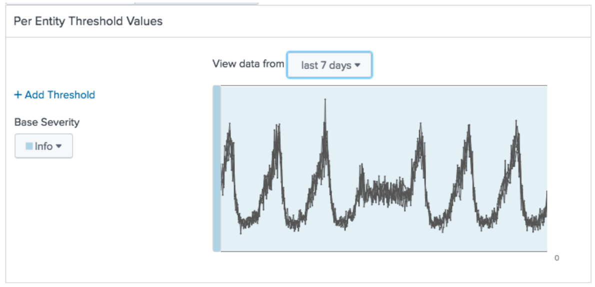
Critical 4000

Medium 2500

+ Add Threshold

Base Severity: Normal

Per Entity Thresholds > Base Severity: Info


Per Entity Threshold Values

View data from last 7 days

+ Add Threshold

Base Severity: Info

5. Click **Finish** to add the new KPI and then click **Save** to update the service.

   The backfill will now start and will take a few minutes to finish. When it completes, proceed to the next task.
6. Wait for the backfill to finish before proceeding. The backfill must complete to provide the necessary baseline for anomaly detection.

Task 2: Configure anomaly detection.

7. In the service editor, open the **Anomaly Detection** section of the **Load** KPI.
8. For **Analysis Time Window**, select **Last 30 days**, and leave the algorithm sensitivity set to 8.
9. Click **Analyze KPI Data**.

   The AD system audits the existing KPI data in **itsi_summary**, looking to see if the **alert_values** for the **Load** KPI indicate a usable statistical model for the algorithms used for anomaly detection. Both trending and entity cohesion are tested. Since this KPI has more than 3 entities, it is a candidate for both entity cohesion analysis and trend analysis.

   The analysis result should return with a recommendation to enable anomaly detection for both the trending and cohesion models.
10. Expand the analysis breakdown and note the anomalies the algorithm found in the past. This indicates a departure from the normal 24 hour pattern for this metric. If this happens again in the future, AD will alert us with a notable event.
11. Enable the **Trending AD** and **Cohesive AD** algorithms.
12. Leave the sensitivity sliders set to **8**.

    This is a good general setting for AD; in practice, you can adjust this up or down as needed to deal with false positives or false negatives.
13. **Save** the service.
14. Wait a few moments, then run this search over the **Last 24 hours**:

    ```
    index=anomaly_detection
    ```

You should see events created here beginning from when you configured AD on the Load KPI. Two sources should be present: `itsi_mad_context` and `itsi_mad_cohesive_context`. The `alert` field is **True** for detected anomalies.

You might see notable events created, if any new anomalies are detected. If so, you'll see these in the next lab exercise.

Task 3: Configure predictive analytics.

15. Navigate to **Apps > Manage Apps** and filter for **toolkit**. The **Machine Learning Toolkit** is installed and enabled.
16. Filter for **scientific**. The **Python for Scientific Computing** app is installed and enabled.
17. Navigate back to the IT Service Intelligence app.
18. Navigate to **Dashboards > Predictive Analytics**.
19. Select the **Online Sales** service.

    Note there are no models available yet.
20. Click the **Service Configuration** link to open the Online Sales service and display the **Predictive Analytics** tab.

    Note the two top graphs, one showing values for the service health score and KPI values over time, the other showing the distribution of service health score values from 0 to 100.
21. Leave **Time Period** set to **Last 14 days**, and **Split for Training/Test** at **70/30**.
22. Use the **Algorithm Type** and **Algorithm** selectors to select each of the 4 model types one at a time, and then use the **Train** button to train a model for each of the 4 algorithms.
23. After training all four models, scroll down to the **Test a Model** section.
24. In the **Regression Models** panel, click the name of each model to see its metrics.

    Note that one of the models has a check-mark in the **Recommended** column. This model shows the best predictive behavior.
25. **Save** the service.
26. From the **Regression Models** panel, select the recommended model.
27. Scroll down to the **Predicted Worst Case Service Health Score** panel and click the **Create Correlation Search** 🔔 icon.
28. Accept all default values and click **Create**, then **OK**. You will now be notified with an alert in **Episode Review** indicating the Online Sales service is predicted to hit or drop below XX in 30 minutes.
29. Navigate to **Dashboards > Predictive Analytics**.
30. Select the **Online Sales** service.
31. In the **Model** selector, note the models are now available for analysts. Select the **Recommended** model.
32. The 30-minute prediction for the health score is shown.
33. Navigate to **Episode Review**.
34. If an episode appears, click the episode.
35. Select the **Common Fields** tab.

36. Examine the current value(s) of the **next30m_worst_hs** field. The predicted health score value(s) is/are lower than the threshold value defined in the notable event title.
37. *Optional*: After 30 minutes elapses, navigate to the Online Sales service's default deep dive, select Last 60 minutes for the timerange. Do the actual health scores match the predicted health scores?

End of Lab Exercise 12

Lab Exercise 13: Implementing Correlation Searches and Multi-KPI Alerts

Description

The customer has asked for the following notifications:

- The IT team wants to be alerted if the number of available web servers falls below a certain level.
- The sales operations team wants to watch the conversion rate. This is roughly computed as the ratio of page views to purchases. High page views correlated with low purchases indicates there's something wrong with the marketing or messaging on the web site. This should be a high severity notable event.
- The anomaly detection you configured for the support ticketing system load KPI should generate notable events.

Task 1: Create the web server alert for the IT team.

You will configure the correlation search to find any errors in a one-hour interval for testing purposes.

1. Select **Configure > Correlation Searches**.
2. Click **Create New Search > Create Correlation Search** and define the Search Properties, Association, and Schedule as follows (accept all other defaults):

   Search Name: Storefront Web Farm Service Level
   Description: Too few servers are running to support the web site
   Search Type: Ad hoc
   Search:
   ```
   sourcetype=access_combined_wcookie
   | eval service_level = 4
   | stats dc(host) as num_servers, last(service_level) as
   service_level
   | where num_servers < service_level
   ```
   This search is intentionally designed to raise alerts immediately, since there are only three servers in the storefront web farm. But for now, you'll use this for testing to ensure notable events are generated.

   Time range: Last 60 minutes
   Service: Storefront Web Farm
   Schedule Type: Basic
   Run Every: minute
3. Notable Events settings (accept all other defaults):

   Notable Event Title: There are only %num_servers% servers running in the storefront web farm
   Notable Event Description: Minimum service level is %service_level% servers
   Severity: High
   Drilldown Search Name: Examine events
   Drilldown Search: `sourcetype=access_combined_wcookie`
   Drilldown earliest offset: Last 60 minutes
   Drilldown latest offset: Next minute
4. Click **Save**, and **OK**.

5. The correlation search begins to run. After a few minutes, you can check **Episode Review**. You should have an episode for this correlation. You may also have one or more episodes for the predicted health of the Online Sales service, from the previous lab.

Task 2: Create the multi-KPI alert for the online sales team.

A multi-KPI alert is well suited for the online sales operations team's needs. They want to raise an alert when purchases are low but views are high.

6. Navigate to **Multi-KPI Alerts**.
7. Click **Status over time** in the upper right corner.
8. Select **Last 60 minutes** for the search time range.
9. Select the **Online Sales** service in the Services panel.

   The service's KPIs appear in the KPIs in selected services panel.
10. Click the **+Add** link for the **Purchases** KPI to add it to the **Selected KPIs** panel.

    The Triggers for Correlation Search modal is displayed.
11. Set the following trigger conditions:

    Critical >= 25%
    High >= 50% (make sure all others are unchecked)
12. Click **Apply**.
13. Repeat the above steps for the **Views** KPI, using these trigger values:

    Low >= 50 %
    Normal >= 25% (make sure all others are unchecked)

    To summarize: "Over the last 60 minutes, if Views are good (alert_level = normal or low), but Purchases are bad (alert_level = critical or high), raise an alert."
14. Click **Save**. The **Create Correlation Search** modal appears. Configure as follows:

    Search Name: Conversion rate
    Notable Event Title: Conversion rate was low
    Notable Event Description: %event_description% (auto-populates the description field)
    Schedule Type: Basic
    Run Every: minute (for testing)
    Severity: Medium
15. Click **Save**.
16. Navigate to **Episode Review**. An episode for the correlation search you created earlier should appear. It could take a few minutes for the Multi-KPI alert to generate an epidode. You might also see episodes for the Online Sale's health score predictive analytics alert and/or the anomaly detection alert you configured in the previous lab exercise.

End of Lab Exercise 13

Lab Exercise 14: Aggregation Policies

Description

In this lab exercise, you'll work with aggregation policies to achieve some of the customer's requirements:

- The IT team wants to be alerted when errors happen in the storefront web farm.
- They work on these errors differently depending on error type, so they want to be able to group by type of error in the Episode Review display.
- Errors during customer purchase activity are critical, so the severity for these notables needs to be increased.

Task 1: Add a new correlation search for storefront web farm errors.

1. Create a new correlation search with the following settings, leaving other settings at the default:

   Search Name: Storefront Web Farm Error
   Search: `sourcetype=access_combined_wcookie status >= 400`
   Time Range: Last 60 minutes
   Service: Storefront Web Farm
   Entity Lookup Field: host

   Note: After you enter this value, the search field is appended with:
   ``| `match_entities(host,sec_grp)` | `filter_maintenance_entities` ``
   This changes the host field to `entity_title`, and suppresses this correlation search for entities in maintenance mode.

   Run Every: minute
   Notable Event Title: Storefront Web Farm Error on %entity_title%
   Notable Event Description: Error %orig_status% occurred on server %entity_title% during %action% process

   Note: The Status field becomes orig_status in the notable event, because status is used for notable event state.

   Severity: Medium
   Drilldown Search Name: Storefront Web Farm Events
   Drilldown Search: `sourcetype=access_combined_wcookie`
2. **Save** the correlation search.
3. Navigate to the **Episode Review** page.
4. Refresh the page until a Storefront Web Farm Error episode appears.
5. Click **Add Filter** and select **Status**, then **New**.
6. In the search box, type: **Storefront Web Farm Error**
7. Save this as the **Storefront Web Farm Errors** view. You can use this view to focus on new notable events.
8. Select the **Storefront Web Farm Error** episode when it appears.
9. Note the list of impacted entities on the **Impact** tab. Notice the latest error code for each entity is displayed.
10. Click the **Common Fields** tab, and expand the list of values for **orig_status**. These are the status codes being returned. There are a range of 4xx and 5xx error codes. Different error types are triaged differently by the IT team, so they'd like to be able to group the notables by error type.

Task 2: Create an aggregation policy to group notables from the web farm error correlation by error type.

11. Navigate to **Configure > Notable Event Aggregation Policies**.
12. Click **Create Notable Event Aggregation Policy**.
13. Under **Include the events if**, click **+ Add Rule (OR)**.
14. In the field name (Select…), type: **source**
15. Make sure **matches** is selected as the operator, then type: **Storefront Web Farm Error** in the right-hand field.
16. Press **tab** to register your field value entry.

    At this point, the preview pane should populate with a single row showing the number of matching notable events. Expand the single row to inspect the individual events in the group.
17. In the **Split events by field** section, remove the default value (host) and type **orig_status**.
18. Press **tab**.

    The preview pane should now change, showing one row per value of **orig_status**.
19. In the **Break episode** section, remove the default (If an event occurs for which severity = Normal).
20. Click **+ Add Breaking Condition (OR)**.
21. From the **If** dropdown, select **the number of events in this episode is**.
22. In the **count limit** field, type: **10**
23. Press **tab**.

    The preview should change again, showing no more than 10 events in any group.
24. Click **Episode information** to expand the section.
25. For **Episode Title**, select **Static value** and type: **Storefront Web Farm: Error %orig_status%**
26. Press **tab**.

    The **Title** column values should update in the preview pane.
27. Click **Next** to display the Action Rules page. You'll change this later. For now, click **Next** to display the **Policy Info** page.
28. In the **Policy Title** field, type: **Storefront Web Farm Errors by Type**
29. Make sure **Status** is **Enabled**, and click **Next**.

    Your policy is created and saved.
30. Wait a few moments, then navigate to the **Episode Review** page.

    Pre-existing notable events will not be re-grouped, but new Storefront Web Farm Error notable events should now start being grouped by error type. You might need to refresh the page a few times.

Task 3: Define an automated action to escalate notable severity for errors in storefront purchase workflow.

The customer has one additional requirement: Errors that occur during the purchase workflow are urgent, and should be escalated to critical severity.

31. Navigate back to the **Notable Event Aggregation Policies** configuration page.
32. Click **Storefront Web Farm Errors by Type** to edit the policy.
33. Select the **Action Rules** tab.
34. Click **+ Add Rule**.
35. From the **If** dropdown, select **the following event occurs**.
36. From the **Select…** dropdown, select the **action** field.
37. Tab to the value field and type: **purchase**

    This rule triggers now for any notable event where the **action** field is **purchase**.
38. In the **Then** pane, make sure **change severity to** is selected (but have a look at other options in that list) and change the severity value from Info to **Critical**.
39. **Save** the edited policy.
40. Navigate back to **Episode Review**.

    Any new notable events for web farm errors that happen during a purchase activity will now be set to critical severity. It may take some time before an example of this scenario is generated. If you don't see any in the next few minutes, return to this view later in the course to verify.

End of Lab Exercise 14

Lab Exercise 15: Service Access Control

Description

In this lab exercise, you'll create a Sales Operations team to restrict the Online Sales service to only sales people.

Task 1: Restrict the Online Sales service to the Sales Operations team.

1. Navigate to **Settings > Access controls > Roles**.
2. Create a new role as follows:

   Role name: salesops_admin
   Default app: itsi
   Inheritance: itoa_team_admin
3. Click **Save**.
4. Create a new role as follows:

   Role name: salesops_analyst
   Default app: itsi
   Inheritance: itoa_analyst
5. Click **Save**.
6. Navigate to **Settings > Access controls > Users**.
7. Edit **analyst1**. This will be your test Sales Ops analyst.
8. Add **salesops_analyst** to the list of selected roles**.**
9. Click **Save**.
10. Navigate to the IT Service Intelligence app, and select **Configure > Teams**.
11. Click **New Team** and configure as follows:

    Title: sales_ops
    Give **salesops_admin** read and write privileges
    Give **salesops_analyst** read privilege
12. Click **Create**.
13. Navigate to **Configure > Services** and, for the **Online Sales** service, select **Edit > Edit Team**.
14. Change the team setting to **sales_ops**.
15. Click **Save**.

    Now only members of the two **salesops** roles can access the service, and only the **salesops_admin** role (or **itoa_admin**) can modify the service.
16. Test this by logging out, and logging into Splunk as **analyst1** (your instructor will supply passwords).

    You should see the **Online Sales**, **Storefront Web Farm**, and **Support Web Farm** services in your Service Analyzer.
17. Log out, and log in as **analyst2**.

    You should only see the **Storefront Web Farm** and **Support Web Farm** services, as well as their KPIs, but not the **Online Sales** service. If you were to create a new glass table, you would not see any KPIs from the **Online Sales** service.

End of Lab Exercise 15

splunk>

Lab Exercise 16: Customization, Backup and Maintenance Mode

Description

In this lab exercise, you'll modify the default daily backup, manually backup the KV store, download the archive to your workstation to examine its contents, and put the Web Farm service into maintenance mode.

Task 1: Modify the daily backup schedule and initiate an immediate backup of the ITSI KV store.

1. Make sure you're logged in to Splunk as **admin**
2. Navigate to **Configure > Backup/Restore** in the ITSI app.

   Note there is one job, **Default Scheduled Backup**. The last backup start and end timestamps are shown, along with the status, **Scheduled Daily**.
3. For the default job, select **Edit** dropdown.

   Note that you can download the most recent scheduled backup, and also restore from the most recent backup. You can also disable the scheduled backup, or edit its settings.
4. Select **Edit > Edit**.

   You can alter the scheduled backup name and description, and also change the schedule to be daily or weekly.
5. Change the schedule to **Weekly**, on **Friday** at **23:00**, and click **Save**.

   The scheduled backup will now run at the end of each week.
6. To run an ad-hoc backup, click **Create Job > Create Backup Job**.
7. Name the job: **End of class**
8. Click **Create**.
9. Refresh the web page until the Status changes to **Completed** for the **End of class** job.
10. Select **Edit > Download Backup** to download a zip file of the backup.

    Note: If the **Download Backup** option is not active in the Edit menu, reload the web page.
11. Expand the zip file on your workstation.

    You'll see a set of JSON files storing the ITSI configuration for the server.

    Note: You could now use this backup package to perform a restore on a different ITSI server.

Task 2: Put the Storefront Web Farm service into maintenance mode.

12. Navigate to **Configure > Maintenance Windows**.
13. Click **Create Maintenance Window**.
14. Configure the parameters as follows:

    Title: Update storefront web servers
    Start Time: the current time is the default, so leave that.
    Duration: 4 hours
    Objects: Services
15. Click **Next**.

16. Select the **Storefront Web Farm** service and click **Create**.
17. Click **Back to Maintenance Windows**.
18. Click **Update storefront web servers**.
19. To see a list of KPIs impacted by this maintenance window, select the **Impacted KPIs** tab. Keep refreshing the page until the KPIs display.
20. Navigate to the default service analyzer. If the **Storefront Web Farm** service and its KPIs have not already changed to grey, refresh the page until they do.

    Great news, the storefront web server update has completed early!
21. Navigate back to **Configure > Maintenance Windows**.
22. For the **Update storefront web servers** maintenance window, select **Edit > End Now**.
23. Click **End Now**.
24. Navigate back to the default service analyzer.

    It will take a few minutes for the **Storefront Web Farm** service and its KPIs to return to normal.

End of Lab Exercise 16 (and end of all lab exercises!)