Transforming IT Operations

# Zenoss Service Dynamics
# Service Impact Installation and Administration

Release 4.2.6

Zenoss, Inc.

www.zenoss.com

# Zenoss Service Dynamics Service Impact Installation and Administration

Copyright © 2014 Zenoss, Inc. All rights reserved.

Zenoss and the Zenoss logo are trademarks or registered trademarks of Zenoss, Inc., in the United States and other countries. All other trademarks, logos, and service marks are the property of Zenoss or other third parties. Use of these marks is prohibited without the express written consent of Zenoss, Inc., or the third-party owner.

Flash is a registered trademark of Adobe Systems Incorporated.

Oracle, the Oracle logo, Java, and MySQL are registered trademarks of the Oracle Corporation and/or its affiliates.

Linux is a registered trademark of Linus Torvalds.

RabbitMQ is a trademark of VMware, Inc.

SNMP Informant is a trademark of Garth K. Williams (Informant Systems, Inc.).

Sybase is a registered trademark of Sybase, Inc.

Tomcat is a trademark of the Apache Software Foundation.

VMware is a registered trademark or trademark of VMware, Inc. in the United States and/or other jurisdictions.

Windows is a registered trademark of Microsoft Corporation in the United States and other countries.

All other companies and products mentioned are trademarks and property of their respective owners.

Part Number: 41-032014-4.2-v30

Zenoss, Inc.
11305 Four Points Drive
Bldg 1 - Suite 300
Austin, Texas 78726

# Contents

# Preface

*Zenoss Service Dynamics Service Impact Installation and Administration* provides detailed instructions for installing, upgrading, using, and administrating Zenoss Service Dynamics Service Impact (Service Impact).

## Audience

This guide is designed for system administrators with Zenoss platform (Zenoss platform) experience. In addition, administrators need working knowledge of Linux system administration, and their data center environment.

## Related publications

| Title | Description |
|---|---|
| *Zenoss Service Dynamics Resource Manager Administration* | Provides an overview of Zenoss platform architecture and features, as well as procedures and examples to help use the system. |
| *Zenoss Service Dynamics Resource Manager Installation* | Provides detailed information and procedures for installing and upgrading Zenoss platform. |
| *Zenoss Service Dynamics Analytics Installation and Administration* | Provides conceptual and procedural information to help you install and use Analytics. |
| *Zenoss Service Dynamics Global Operations Management* | Provides conceptual and procedural information to help you install and use Global Operations Manager. |
| *Zenoss Service Dynamics Release Notes* | Describes known issues, fixed issues, and late-breaking information not already provided in the Zenoss Service Dynamics documentation set. |
| *Zenoss Service Dynamics Extended Monitoring* | Provides detailed information about extending monitoring and other capabilities provided by ZenPacks. |

## Additional information and comments

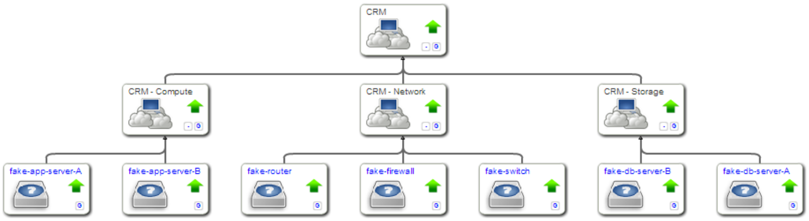
If you have technical questions about this product that are not answered in this guide, visit the *Zenoss Support* site.

Zenoss welcomes your comments and suggestions regarding our documentation. To share your comments, please send an email to `docs@zenoss.com`. In the email, include the document title and part number. The part number appears at the end of the list of trademarks, at the front of this guide.

# Introduction to Service Impact 1

Zenoss Service Dynamics Service Impact (Service Impact) reveals the impact of Zenoss platform events on the services your environment provides. Service Impact performs near-real-time dependency tracking, to maintain the availabilty and performance state information of all services. When new events occur, Service Impact knows the service context of the event, and automatically performs a root-cause analysis (RCA), so you can quickly diagnose and triage problems—often, before end-users even notice or call to complain.

Service Impact enables you to quickly create rich service models, and displays interactive graphs of them, like the following example.



Service Impact overcomes the challenges of manual RCA.

- Network operations center personnel do not have uniform levels of domain knowledge.

  In Service Impact, domain knowledge is captured in lightweight, context-sensitive policies.
- Events occur without context, eventually becoming a "sea of red" in the user interface.

  Service models provide the context of events, and Service Impact uses the resource and service dependencies in the models to perform its automated RCA analysis, and deliver notifications of significant events, in context.
- The process of filtering out "noise" events, identifying impacted services, and starting recovery procedures is time-consuming and error-prone.

  Service Impact maintains a near-real-time dependency map of each service model, and can be used to alert on service-level performance metrics, before users notice or costly fines are levied.

## Service models represented as graphs

In Service Impact, a service model is represented as a graph of the following nodes:

- *Service nodes*, representing either a service model as a whole, or a service on which other service nodes rely (a subservice). Service node roles are contextual.

- *Device nodes* and *component nodes*, representing resources monitored by Zenoss platform.
- *Logical nodes*, customizable objects that capture specific Zenoss platform event states. Logical nodes allow you to represent resources or services not monitored by Zenoss platform, or, for resources that are monitored by Zenoss platform, to capture event states from arbitrary classes.

Any node can be used in multiple service models. For example, network-related nodes are almost always shared, which greatly simplifies the process of creating new service models.

Graphs are built from model information stored in the Zope object database (ZODB), from other information available in Zenoss platform, and from nodes created in Service Impact.

Service Impact provides organizers (folders) to group nodes, and to enable summary dashboards. In addition, organizers provide valuable context in *service events* (state changes that affect service nodes) and in Zenoss Service Dynamics Analytics reports.

### Example use case

The following list outlines the flow of an example Service Impact use case.

1. A switch monitored by Zenoss platform goes down.
2. Zenoss platform generates an event about the switch.
3. Service Impact reads the switch down event, and changes the state of the switch node.
4. Service Impact evaluates state triggers in the service model graphs to which the switch node belongs.
5. Service Impact propogates state data according to the policies defined in the parent nodes of the switch.
6. If the switch node's state changes the state of a parent service node, Service Impact generates a service event, and sends it to Zenoss platform. (Like any other event, service event notifications can be customized.)
7. Service Impact performs a root cause analysis, generating confidence rankings of the impact of the switch down event, in each service model graph to which the switch belongs.

## State propagation controlled by policies

Policies determine the type of availability and performance state data passed from the bottom to the top of a service model graph, and the node or nodes that receive the data. State changes that affect service nodes result in service events, which are sent to Zenoss platform. Policies may be set on any node.

- A policy can include one or more user-defined *state triggers*, rules that examine the states of child nodes. State triggers select the type of state data to send to the parent level in the service model. All state triggers are evaluated before state data is sent.

  Nodes with *custom state providers* provide specialized options for defining state triggers. Service Impact always collects information from events belonging to most of the `/Status` subclasses, and all of the `/Perf` classes. Zenoss platform can be customized to gather state data from events belonging to other classes. For example, custom state providers are used to define state triggers for device and component nodes monitored through customized classes provided by the ZenVMware and CiscoUCS ZenPacks.
- The policy type determines which parent node or nodes receive the state data. The following list defines the policies, in their order of precedence:

  1. A node with a *contextual policy* sends different state data to different nodes in the next-higher level. However, contextual policies only apply to specific service models.
  2. A node with a *global policy* sends state data to its parent node. Global policies apply to all instances of a node, in all service models.
  3. A node with the *default policy* (neither of the preceding policies) sends the state data of the worst condition affecting it to its parent node.

## Service model definition process

The key to defining accurate service models is thorough dependency discovery. This section outlines a repeatable process for defining service models.

Note Defining service models is influenced by the following environmental factors:

- The level of organizational maturity (processes and tooling)
- The automation level of service life cycles (provisioning and management)
- The standardization level of applications and infrastructure

Nevertheless, by using the following process, you can create accurate service models.

**1** Define the service to model.

A service is defined by its boundaries and type.

- A web service (for example, a human resources portal) or an IaaS platform (an Amazon EC2 instance) depend on specific resources, which may be internal or external.
- An SaaS service (*ServiceNow*) is defined by its deployment architecture.

In Service Impact, a service model is represented as a *service node*.

**2** Define services nodes for subservices.

A subservice has a direct relationship with one or more services: If the subservice fails or degrades, the service fails or degrades. Often, a subservice represents an infrastructure tier, such as a gateway or database service. At a lower level, a tier may be configured with redundant elements for high availability. However, Zenoss recommends modeling subservices as single points of failure.

In Service Impact, subservices are represented as *service nodes*. (Service node roles are contextual.)

**3** Add device nodes, component nodes, and logical nodes to subservice nodes.

Once subservices are defined as service nodes, the infrastructure resources that make up the subservices can be added.

**4** Define global policies on subservice nodes.

Policies capture domain knowledge about services, and enable the automated dependency tracking and RCA computation that make Service Impact so valuable. Zenoss recommends using global policies instead of contextual policies wherever possible, to facilitate subservice re-use. Contextual policies are valuable, but their use cases are relatively rare.

The primary reason is that most deployment scenarios utilize shared resources. For example, a hypervisor hosts virtual machines that belong to separate services. One service node with a global policy can apply to all virtual machines, across all service models. The service relationship, not service ownership, is the key to determining the relevance of events and sending state data to parent nodes.

Note Steps 2, 3, and 4 may be performed iteratively.

**5** Perform gap analysis.

Identify gaps in monitoring processes by analyzing failure scenarios. Are the key measurement points of each node in the service model properly monitored? For example, are synthetic transactions in place for web servers? Are ping checks being performed against host operating systems? Service Impact can only act on events that flow through Zenoss platform.

**6** Test failure scenarios.

The `zensendevent` command generates synthetic events, which are invaluable for validating service relationships, policies, and even monitoring functions, before real events or event storms occur.

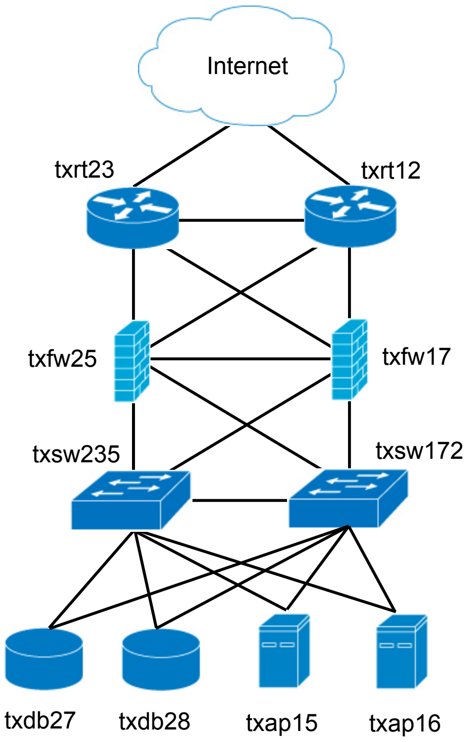
7. Refine the service model.

   If a test or gap analysis reveals missing policies or service nodes, add them and test again.

# Tutorial 2

This tutorial demonstrates how to create a service model for a simplified CRM application, how to view the model, and how to view and interpret the root cause analyses that Service Impact creates when events affect the application's availability state.

The following diagram shows the network topology of the CRM application featured in this tutorial.



| LEGEND |
|---|
| tx = Texas |
| rt = router |
| fw = firewall |
| sw = switch |
| db = database |
| ap = application |

Note The exercises in this tutorial are based on fake devices, which do not affect production environments. The final task erases the fake devices and the nodes created during the tutorial.

## 1. Define the service to model

The service to model is a CRM application, and it is defined by the resources on which it relies.

The CRM application relies on application and database hosts and processes, and on the network infrastructure that connects the service with its users. This tutorial creates a model of the development deployment of the CRM application, not the production or quality assurance deployments.

For this tutorial, all device and component resources are fake devices. *1.1. Fake device descriptions and roles* on page 12 describes the fake devices. The following procedure loads the fake devices into Zenoss platform, and sets up the tutorial environment.

1. Log in to the Zenoss platform master host as `zenoss`.
2. Create a soft link to the Service Impact scripts directory.

   ```
   ln -s ${ZENHOME}/ZenPacks/ZenPacks.zenoss.Impact-VERSION-py2.7.egg\
   /ZenPacks/zenoss/Impact/scripts/tutorial ${HOME}/impact_scripts
   ```
3. Change directory to the Service Impact scripts directory.

   ```
   cd ${HOME}/impact_scripts
   ```
4. Load fake devices and components into Zenoss platform.

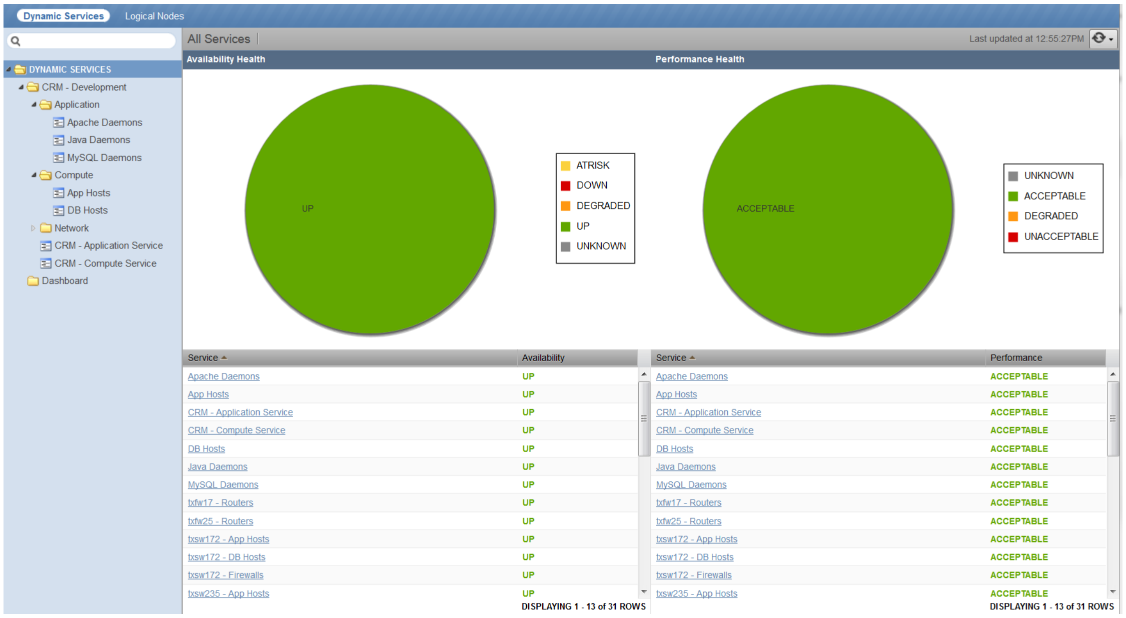
   ```
   zenbatchload --nomodel ./devices.txt
   ```
5. Check the values of the `ZENOSS_USERNAME` and `ZENOSS_PASSWORD` variables in `tutorial.sh`, and change them, if necessary.
   The values must be the username and password of a Zenoss platform user account.
6. Add execute permission to the `tutorial.sh` script.

   ```
   chmod +x ./tutorial.sh
   ```
7. Set up the tutorial environment.

   ```
   ./tutorial.sh
   ```
8. Log in to the Zenoss platform browser interface as a user with ZenManager or Manager privileges.
9. Click **SERVICES**.

   The Service Impact feature of Zenoss platform adds a tab named **SERVICES** to the Zenoss platform menu bar.
10. In the tree view, open the **CRM - Development** organizer, and then open the **Application** and **Compute** organizers.

    The tree view displays the service nodes created by the `tutorial.sh` script, as shown in the following example.



| Service ▲ | Availability |
|---|---|
| Apache Daemons | UP |
| App Hosts | UP |
| CRM - Application Service | UP |
| CRM - Compute Service | UP |
| DB Hosts | UP |
| Java Daemons | UP |
| MySQL Daemons | UP |
| txfw17 - Routers | UP |
| txfw25 - Routers | UP |
| txsw172 - App Hosts | UP |
| txsw172 - DB Hosts | UP |
| txsw172 - Firewalls | UP |
| txsw235 - App Hosts | UP |

DISPLAYING 1 - 13 of 31 ROWS

| Service ▲ | Performance |
|---|---|
| Apache Daemons | ACCEPTABLE |
| App Hosts | ACCEPTABLE |
| CRM - Application Service | ACCEPTABLE |
| CRM - Compute Service | ACCEPTABLE |
| DB Hosts | ACCEPTABLE |
| Java Daemons | ACCEPTABLE |
| MySQL Daemons | ACCEPTABLE |
| txfw17 - Routers | ACCEPTABLE |
| txfw25 - Routers | ACCEPTABLE |
| txsw172 - App Hosts | ACCEPTABLE |
| txsw172 - DB Hosts | ACCEPTABLE |
| txsw172 - Firewalls | ACCEPTABLE |
| txsw235 - App Hosts | ACCEPTABLE |

DISPLAYING 1 - 13 of 31 ROWS

### Related Links

*Service Impact home page* on page 24
The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

*Overview* on page 25
The **Overview** view provides details about a service node, including the list of nodes that are associated with it.

## 1.1. Fake device descriptions and roles

A description of each fake device used in the tutorial, and the role (Zenoss platform device or component) each fake device plays.

### Application host 15

| Fake device name | Description | Role |
|---|---|---|
| fake-txap15 | Application host 15 | Device |
| fake-txap15-httpd | Apache daemon | Component |
| fake-txap15-java | Java/JRE daemon | Component |
| fake-txap15-nic-0 | Network interface card 0 | Component |
| fake-txap15-nic-1 | Network interface card 1 | Component |

### Application host 16

| Fake device name | Description | Role |
|---|---|---|
| fake-txap16 | Application host 16 | Device |
| fake-txap16-httpd | Apache daemon | Component |
| fake-txap16-java | Java/JRE daemon | Component |
| fake-txap16-nic-0 | Network interface card 0 | Component |
| fake-txap16-nic-1 | Network interface card 1 | Component |

### Database host 27

| Fake device name | Description | Role |
|---|---|---|
| fake-txdb27 | Database host 27 | Device |
| fake-txdb27-mysqld | MySQL daemon | Component |
| fake-txdb27-nic-0 | Network interface card 0 | Component |
| fake-txdb27-nic-1 | Network interface card 1 | Component |

### Database host 28

| Fake device name | Description | Role |
|---|---|---|
| fake-txdb28 | Database host 28 | Device |
| fake-txdb28-mysqld | MySQL daemon | Component |
| fake-txdb28-nic-0 | Network interface card 0 | Component |

| Fake device name | Description | Role |
|---|---|---|
| fake-txdb28-nic-1 | Network interface card 1 | Component |

### Firewall 17

| Fake device name | Description | Role |
|---|---|---|
| fake-txfw17 | Firewall 17 | Device |
| fake-txfw17-10g-0 | Port 0 (10GB capacity) | Component |
| fake-txfw17-10g-1 | Port 1 (10GB capacity) | Component |
| fake-txfw17-10g-2 | Port 2 (10GB capacity) | Component |
| fake-txfw17-10g-3 | Port 3 (10GB capacity) | Component |
| fake-txfw17-1g-0 | Port 0 (1 GB capacity) | Component |

### Firewall 25

| Fake device name | Description | Role |
|---|---|---|
| fake-txfw25 | Firewall 25 | Device |
| fake-txfw25-10g-0 | Port 0 (10GB capacity) | Component |
| fake-txfw25-10g-1 | Port 1 (10GB capacity) | Component |
| fake-txfw25-10g-2 | Port 2 (10GB capacity) | Component |
| fake-txfw25-10g-3 | Port 3 (10GB capacity) | Component |
| fake-txfw25-1g-0 | Port 0 (1 GB capacity) | Component |

### Router 12

| Fake device name | Description | Role |
|---|---|---|
| fake-txrt12 | Router 12 | Device |
| fake-txrt12-100g-0 | Port 0 (100GB capacity) | Component |
| fake-txrt12-10g-0 | Port 0 (10GB capacity) | Component |
| fake-txrt12-10g-1 | Port 1 (10GB capacity) | Component |
| fake-txrt12-1g-0 | Port 0 (1GB capacity) | Component |

### Router 23

| Fake device name | Description | Role |
|---|---|---|
| fake-txrt23 | Router 23 | Device |
| fake-txrt23-100g-0 | Port 0 (100GB capacity) | Component |
| fake-txrt23-10g-0 | Port 0 (10GB capacity) | Component |
| fake-txrt23-10g-1 | Port 1 (10GB capacity) | Component |
| fake-txrt23-1g-0 | Port 0 (1GB capacity) | Component |

### Switch 172

| Fake device name | Description | Role |
|---|---|---|
| fake-txsw172 | Switch 172 | Device |
| fake-txsw172-10g-0 | Port 0 (10GB capacity) | Component |
| fake-txsw172-10g-1 | Port 1 (10GB capacity) | Component |
| fake-txsw172-1g-0 | Port 0 (1GB capacity) | Component |
| fake-txsw172-1g-1 | Port 1 (1GB capacity) | Component |
| fake-txsw172-1g-2 | Port 2 (1GB capacity) | Component |
| fake-txsw172-1g-3 | Port 3 (1GB capacity) | Component |
| fake-txsw172-1g-4 | Port 4 (1GB capacity) | Component |

### Switch 235

| Fake device name | Description | Role |
|---|---|---|
| fake-txsw235 | Switch 235 | Device |
| fake-txsw235-10g-0 | Port 0 (10GB capacity) | Component |
| fake-txsw235-10g-1 | Port 1 (10GB capacity) | Component |
| fake-txsw235-1g-0 | Port 0 (1GB capacity) | Component |
| fake-txsw235-1g-1 | Port 1 (1GB capacity) | Component |
| fake-txsw235-1g-2 | Port 2 (1GB capacity) | Component |
| fake-txsw235-1g-3 | Port 3 (1GB capacity) | Component |
| fake-txsw235-1g-4 | Port 4 (1GB capacity) | Component |

## 1.2. Introduction to the tutorial environment

The `tutorial-setup.sh` script creates the following service nodes and organizers to initialize the CRM service model graph in Service Impact.

- The **Dashboard** organizer.

  This root-level organizer is for service nodes that represent service models as a whole, a best practice. Initially, the organizer is empty.
- The root-level **CRM - Development** organizer, containing additional organizers.

  Zenoss recommends using a single root-level organizer to contain the subservices of each service model, and using standardized names (and contents) for sub-organizers.
- The **CRM - Application Service** and **CRM - Compute Service** service nodes, children of the **CRM - Development** organizer.

  These service nodes summarize the application and compute services associated with the CRM application, and are easily located without having to open the organizers in which their constituent subservices are located. This is a best practice.
- The service nodes in the **Network** organizer that start with **zfake** represent the network connections between fake devices.

Because these fake devices are not modeled, Zenoss platform cannot discern their relationships, and Service Impact cannot create device or component nodes for us. So the setup script creates service nodes to represent the connections.

None of these nodes have either contextual or global policies, so the default policy applies: The state of the worst condition affecting child nodes becomes the state of the **zfake** service nodes, which is the correct policy for these connections.

- All of the DNS and interface names follow a naming convention, a best practice.
- The subservice nodes in the **Network** organizer that start with **tx** contain redundant resources, and have standardized, global availability policies defined.

  These subservice nodes embody several best practices.

  - Each subservice node contains homogeneous child nodes. Global policies work best when child nodes are homogeneous.
  - Each subservice uses global policies. Global policies can be re-used across service model boundaries. Contextual policies are restricted to specific service models.
  - Each global policy contains the following, standardized state triggers: The availability state is ATRISK if 50% or more child nodes are down, and DOWN if 100% of child nodes are down. By using percentage thresholds, the policies do not need adjustment if additional resources are deployed at a later date.

  Note The standardized state triggers used in this case are not intrinsically best practices. Rather, they are examples of thinking about and using global policies systematically. For example, if a resource pool contains more than two members, additional state triggers may be defined.

The remaining procedures in this tutorial demonstrate how to complete the CRM service model, and how to test it.

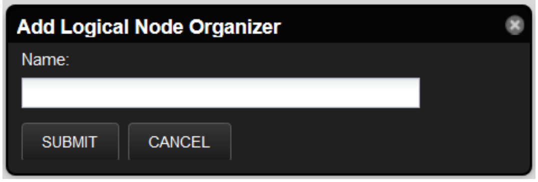
## 2. Create logical node and subservice nodes for internet connections

All of the network connections are modeled, except the connection to the internet. Since the internet is not modeled in Zenoss platform, we create a logical node to represent it. For this tutorial, the logical node is simply an object that reacts to fake events. A "real" logical node could be configured to react to events from a `zencommand` that pings an internet resource.

### 2.1. Create logical node

1 In the Zenoss platform browser interface, select **SERVICES > Logical Nodes**.

2 From the **Add** menu at the bottom of the tree view, select **Add Logical Node Organizer**.



3 In the **Add Logical Node Organizer** dialog, enter `CRM - Development`, and then click **SUBMIT**.

4 From the **Add** menu at the bottom of the tree view, select **Add Logical Node**.

5 In the **Add Logical Node** dialog, enter `zenoss.com`, and then click **SUBMIT**.

6 In the `zenoss.com` details view, enter values to match the following table, and then click **Save**.

| Field | Description |
|---|---|
| **Description** | A node to help represent a route to the Internet. |
| **Criteria** | all, Summary, contains, fakeInternet |

| Field | Description |
|---|---|
| **Events for this node in this event class** | /Status/Ping |
| **will result in these availability states** | Critical: DOWN, Error: DOWN, Warning: ATRISK, Clear: UP |

### Related Links

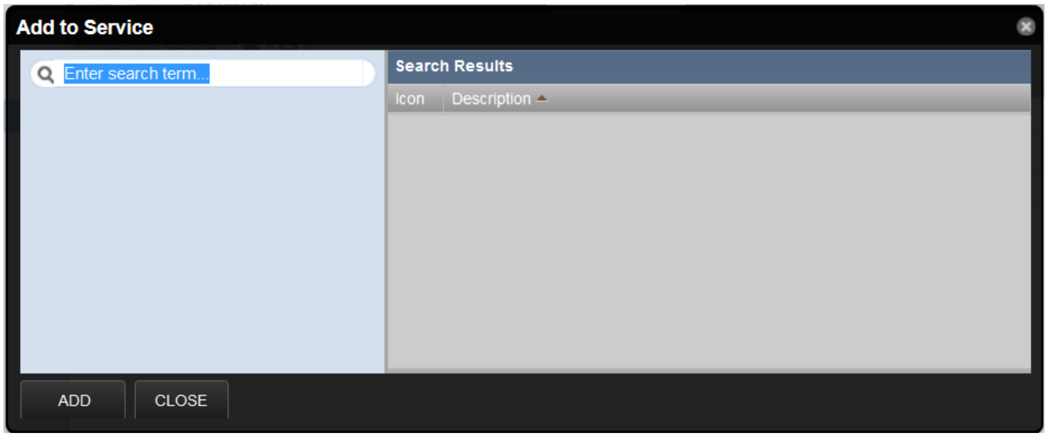
*Service Impact home page* on page 24
The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

*Logical node details view* on page 33
The logical node details view allows you to create and edit logical nodes.

## 2.2. Create a subservice node for one internet connection

Create a service node for the connection from router 12 to the `zenoss.com` logical node.

1 In the Zenoss platform browser interface, select **SERVICES > Dynamic Services**.
2 In the tree view, open the **CRM - Development** organizer, and then open the **Network** organizer.
3 In the tree view, select the **Network** organizer.
When an organizer is not selected, new service nodes are created at the root level. If that happens, simply drag the new service node into the correct organizer.
4 From the **Add** menu at the bottom of the tree view, select **Add Dynamic Service**.
5 In the **Add Dynamic Service** dialog, enter `txrt12 - Internet`, and then click **SUBMIT**.
6 In the **Overview** view, click the **Add** button.
7 In the **Add to Service** dialog, enter `fake-txrt12` in the search field.



Service Impact begins searching for matches after the first 3 characters are entered.
8 Select **Device** in the left column, and **fake-txrt12-100g-0** in the results list, then click **ADD**.
9 In the search field, enter `zenoss.com`.
10 From the search results list, select `zenoss.com`.
11 Click **ADD** and then **CLOSE**.

### Related Links

*Service Impact home page* on page 24
The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

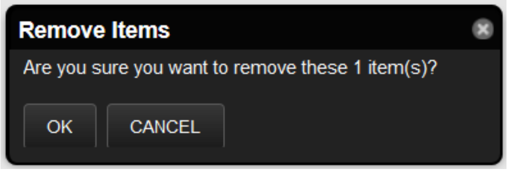
*Add to Service dialog* on page 26
The interface for finding and adding nodes to a service node.

*Overview* on page 25
The **Overview** view provides details about a service node, including the list of nodes that are associated with it.

## 2.3. Create a subservice node for the other internet connection

Clone the service node for the connection through router 12 to create the service node for the connection through router 23.

1 In the tree view, select service node **txrt12 - Internet**.
2 From the **Action** menu at the bottom of the tree view, select **Clone Service...**.
3 In the **Clone Service** dialog, enter `txrt23 - Internet`, and then click **SUBMIT**.
The new service is created and its contents are displayed in the **Overview** view.
4 In the list of nodes, select **fake-txrt12**, and then click the remove button.
If you click on the name of the node, Zenoss platform displays the device overview page. Click the browser's back button to return to the correct page.



Remove Items

Are you sure you want to remove these 1 item(s)?

5 In the **Remove Items** dialog, click **OK**.
6 In the **Overview** view, click the **Add** button.
7 In the **Add to Service** dialog, enter `100g` in the search field.
8 In the left column, select **Device**. In the results list, select **fake-txrt23-100g-0**. Then click **ADD** and **CLOSE**.

Note You may double-click an entry in the results list to add it to a service node.

### Related Links

*Service Impact home page* on page 24
The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

*Add to Service dialog* on page 26
The interface for finding and adding nodes to a service node.

*Overview* on page 25
The **Overview** view provides details about a service node, including the list of nodes that are associated with it.

## 2.4. Create service node to represent redundant paths to the internet

The previous tasks created subservices to represent the two WAN connections to the internet. This task creates a subservice node to represent the redundant WAN tier.

1 In the tree view, select the **Network** organizer.
2 From the **Add** menu at the bottom of the tree view, select **Add Dynamic Service**.
3 In the **Add Dynamic Service** dialog, enter `Routers - Internet`, and then click **SUBMIT**.
4 In the **Overview** view, click the **Add** button.
5 In the **Add to Service** dialog, enter `Internet` in the search field.
6 In the left column, select **DynamicService**.
7 In the results list, select **txrt12-Internet** and **txrt23-Internet**, and then click **ADD** and **CLOSE**.

### Related Links

*Service Impact home page* on page 24

The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

*Add to Service dialog* on page 26
The interface for finding and adding nodes to a service node.

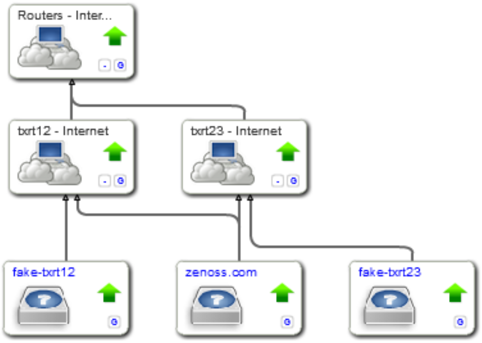
*Overview* on page 25
The **Overview** view provides details about a service node, including the list of nodes that are associated with it.

## 2.5. Add a policy to the internet connection node

Add a global availability policy to the internet connection subservice node.

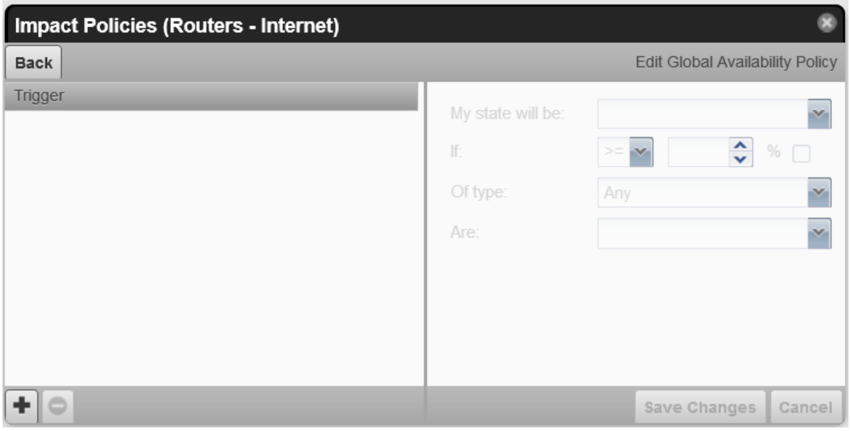
1 In the tree view, select **Routers - Internet**.

2 In the **Overview** view, select **Impact View**.



3 In the service element widget of the **Routers - Internet** service (the widget at the top of the hierarchy), click the **Policy Gate** link, located in the lower-right corner.

4 In the **Impact Policies** dialog, click the **Add** button directly to the right of the **Global Policy** entry.


Impact Policies (Routers - Internet)
Back
Edit Global Availability Policy
Trigger
My state will be:
If:
>=
%
Of type:
Any
Are:
Save Changes
Cancel

5 In the lower-left corner of the **Edit...Policy** tab, click the **Add** button, and then add the following triggers.

- `ATRISK if >= 50% DynamicService is DOWN`
- `DOWN if >= 100% DynamicService is DOWN`

Note When you select a type for a state trigger, that selection excludes other types. In this case, the only options are **Any** and **DynamicService**, and both child nodes are service nodes, so exclusivity is not a concern.

### Related Links

*Service Impact home page* on page 24

The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

*Impact View* on page 27
The **Impact View** displays the interactive graph of a service node.

*Impact Policies dialog* on page 31
The **Impact Policies** dialog provides options for defining global and contextual policies, by creating or editing state triggers.

*Edit...Policy tab* on page 32
The **Edit...Policy** tab of the **Impact Policies** dialog is the interface for adding or editing state triggers for contextual or global policies.

## 3. Create service nodes for major network segments

All of the connections between devices are modeled, and all of the redundant resources are modeled. This procedure creates service nodes for the major network segments.

1. In the Zenoss platform browser interface, select **SERVICES > Dynamic Services**.
2. In the tree view, open the **CRM - Development** organizer, and then open and select the **Network** organizer.
3. Create new service nodes for the major network segments.

   The following table matches new and existing service nodes.

   | New service node | Existing service nodes |
   |---|---|
   | App Hosts - Switches | txsw235 - App Hosts, txsw172 - App Hosts |
   | DB Hosts - Switches | txsw235 - DB Hosts, txsw172 - DB Hosts |
   | Firewalls - Routers | txfw25 - Routers, txfw17 - Routers |
   | Switches - Firewalls | txsw235 - Firewalls, txsw172 - Firewalls |

   Refer to the preceding steps for detailed instructions.
4. Add the following availability state triggers to each new service node.

   - `ATRISK if >= 50% DynamicService is DOWN`
   - `DOWN if >= 100% DynamicService is DOWN`

   Refer to the preceding steps for detailed instructions.

### Related Links

*Service Impact home page* on page 24
The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

*Add to Service dialog* on page 26
The interface for finding and adding nodes to a service node.

*Overview* on page 25
The **Overview** view provides details about a service node, including the list of nodes that are associated with it.

## 4. Create summary nodes for critical paths

Services nodes for all of the major network segments are in place, so now it's time to consider nodes to represent the critical paths. The following, minimal conditions must be satisfied to characterize the CRM application as available.

- One connection between internet users and an application server is UP.
- One connection between an application server and a database server is UP.

If nodes for these paths already existed, we could create the service node that summarizes the network service for CRM. However, only one segment is defined, by the **Routers - Internet** subservice. Two additional subservices are required, for the following paths:

- the path between the routers and the application hosts
- the path between the application and database hosts

1 In the Zenoss platform browser interface, select **SERVICES > Dynamic Services**.
2 In the tree view, open the **CRM - Development** organizer, and then open and select the **Network** organizer.
3 Create new service nodes for the network paths that affect users.

The following table matches new and existing service nodes.

| New service node | Existing service nodes |
|---|---|
| Routers - App Hosts | App Hosts - Switches, Firewalls - Routers, Switches - Firewalls |
| App Hosts - DB Hosts | App Hosts - Switches, DB Hosts - Switches |

Refer to the preceding steps for detailed instructions.

Note The correct policy for these new service nodes is the default policy, because each subservice is critical.

### Related Links

*Service Impact home page* on page 24
The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

*Add to Service dialog* on page 26
The interface for finding and adding nodes to a service node.

*Overview* on page 25
The **Overview** view provides details about a service node, including the list of nodes that are associated with it.

## 5. Create a node for the network service

The critical network paths are defined; now, create a service node to represent the network service for the CRM application.

1 In the Zenoss platform browser interface, select **SERVICES > Dynamic Services**.
2 In the tree view, open the **CRM - Development** organizer, and then select it.
Service nodes that represent a category should be peers of their sub-organizers.
3 Create a new service node, named `CRM - Network Service`.
4 Add the following subservice nodes to the new service node.

- App Hosts - DB Hosts
- Routers - App Hosts
- Routers - Internet

The correct policy for this node is the default policy.

### Related Links

*Service Impact home page* on page 24

The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

*Add to Service dialog* on page 26
The interface for finding and adding nodes to a service node.

*Overview* on page 25
The **Overview** view provides details about a service node, including the list of nodes that are associated with it.

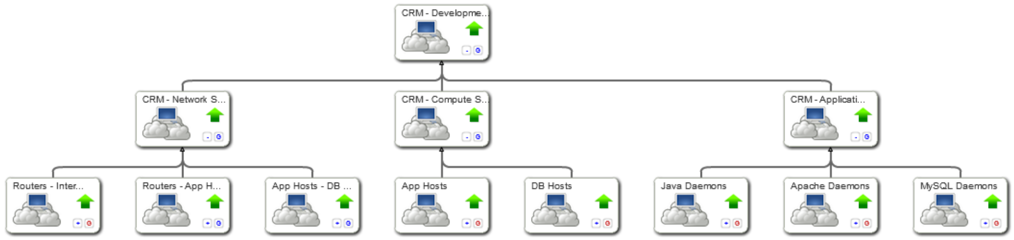
## 6. Create a service node for the service model

With all of the resource categories modeled, the application as a whole can be modeled.

1 In the Zenoss platform browser interface, select **SERVICES > Dynamic Services**.
2 In the tree view, select the **Dashboard** organizer.
Service nodes that represent a service model as a whole should be kept in a separate, root-level organizer. This way, you can quickly determine the state of all service models in the environment.
3 Create a service node named `CRM - Development Service` and add the following subservice nodes to it.

- CRM - Application Service
- CRM - Compute Service
- CRM - Network Service

To view the model, select the new service in the tree view, and then select **Impact View**. The graph looks similar to the following image.



If the graph is off-center, use the overview toolbar to center it.

### Related Links

*Service Impact home page* on page 24
The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

*Add to Service dialog* on page 26
The interface for finding and adding nodes to a service node.

*Overview* on page 25
The **Overview** view provides details about a service node, including the list of nodes that are associated with it.

## 7. Send events to fake devices

Send events to the fake devices to see how they affect the availability of the CRM application.

1 In the tree view, open the **Dashboard** organizer, and then select **CRM - Development Service**.
2 In the main view area, select **Impact View**.
3 Log in to the Zenoss platform master host as `zenoss`.

4 Send ping down events to the fake network interface card components in the `txap15` and `txap16` hosts.

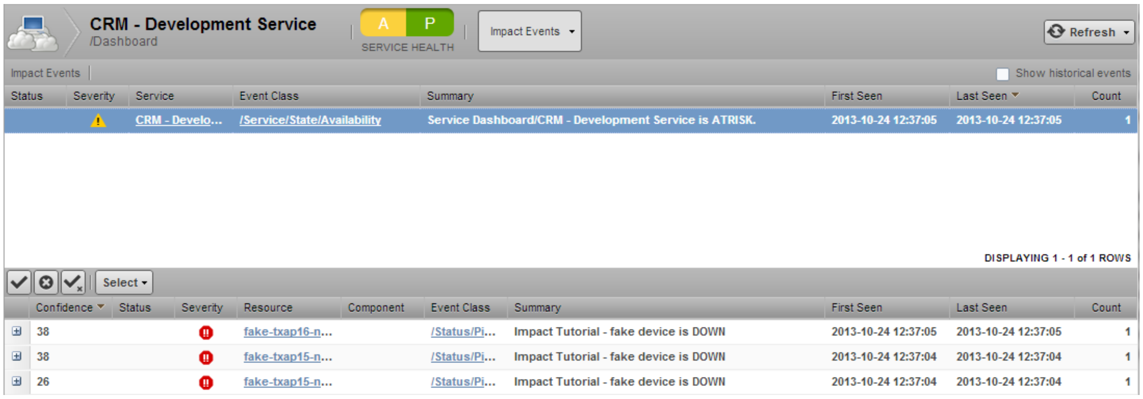
```
zensendevent -d fake-txap15-nic-0 -c /Status/Ping -s Critical \
   "Impact Tutorial - fake device is DOWN"
zensendevent -d fake-txap15-nic-1 -c /Status/Ping -s Critical \
   "Impact Tutorial - fake device is DOWN"
zensendevent -d fake-txap16-nic-0 -c /Status/Ping -s Critical \
   "Impact Tutorial - fake device is DOWN"
```

5 In the browser, click the **Refresh** button.

The **Impact View** shows the CRM application availability state changed to ATRISK, and the nodes involved in the state change highlighted with yellow and red.

To see all of the nodes in the service model graph, click **Expand All**.

6 Change the view to **Impact Events**, and then click **CRM - Development Service** in the **Impact Events** list.



| Status | Severity | Service | Event Class | Summary | First Seen | Last Seen | Count |
|---|---|---|---|---|---|---|---|
|  | ⚠ | CRM - Develo... | /Service/State/Availability | Service Dashboard/CRM - Development Service is ATRISK. | 2013-10-24 12:37:05 | 2013-10-24 12:37:05 | 1 |

DISPLAYING 1 - 1 of 1 ROWS

|  | Confidence | Status | Severity | Resource | Component | Event Class | Summary | First Seen | Last Seen | Count |
|---|---|---|---|---|---|---|---|---|---|---|
| ⊞ | 38 |  | ● | fake-txap16-n... |  | /Status/Pi... | Impact Tutorial - fake device is DOWN | 2013-10-24 12:37:05 | 2013-10-24 12:37:05 | 1 |
| ⊞ | 38 |  | ● | fake-txap15-n... |  | /Status/Pi... | Impact Tutorial - fake device is DOWN | 2013-10-24 12:37:04 | 2013-10-24 12:37:04 | 1 |
| ⊞ | 26 |  | ● | fake-txap15-n... |  | /Status/Pi... | Impact Tutorial - fake device is DOWN | 2013-10-24 12:37:04 | 2013-10-24 12:37:04 | 1 |

The events that contribute to the current state of the CRM application are weighted by the root cause analysis that Service Impact performs (the **Confidence** column). To view the impact chain of an event, click the plus button in the left column.

7 Send ping down events to the fake network interface card components in the `txdb27` and `txdb28` hosts, and then click the **Refresh** button.

```
zensendevent -d fake-txdb27-nic-0 -c /Status/Ping -s Critical \
   "Impact Tutorial - fake device is DOWN"
zensendevent -d fake-txdb27-nic-1 -c /Status/Ping -s Critical \
   "Impact Tutorial - fake device is DOWN"
zensendevent -d fake-txdb28-nic-0 -c /Status/Ping -s Critical \
   "Impact Tutorial - fake device is DOWN"
```

The list of contributing events grows, and the rankings change to reflect the added events.

8 Send clear events to all of the fake devices that are down.

```
zensendevent -d fake-txap15-nic-0 -c /Status/Ping -s Clear \
   "Impact Tutorial - fake device is UP"
zensendevent -d fake-txap15-nic-1 -c /Status/Ping -s Clear \
   "Impact Tutorial - fake device is UP"
zensendevent -d fake-txap16-nic-0 -c /Status/Ping -s Clear \
   "Impact Tutorial - fake device is UP"
zensendevent -d fake-txdb27-nic-0 -c /Status/Ping -s Clear \
   "Impact Tutorial - fake device is UP"
zensendevent -d fake-txdb27-nic-1 -c /Status/Ping -s Clear \
   "Impact Tutorial - fake device is UP"
zensendevent -d fake-txdb28-nic-0 -c /Status/Ping -s Clear \
   "Impact Tutorial - fake device is UP"
```

### Related Links

*Service Impact home page* on page 24
The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

*Impact Events* on page 26
The **Impact Events** view shows summary and detail information about events that are affecting a service node.

## 8. Remove tutorial service elements

This step removes all of the tutorial-defined nodes.

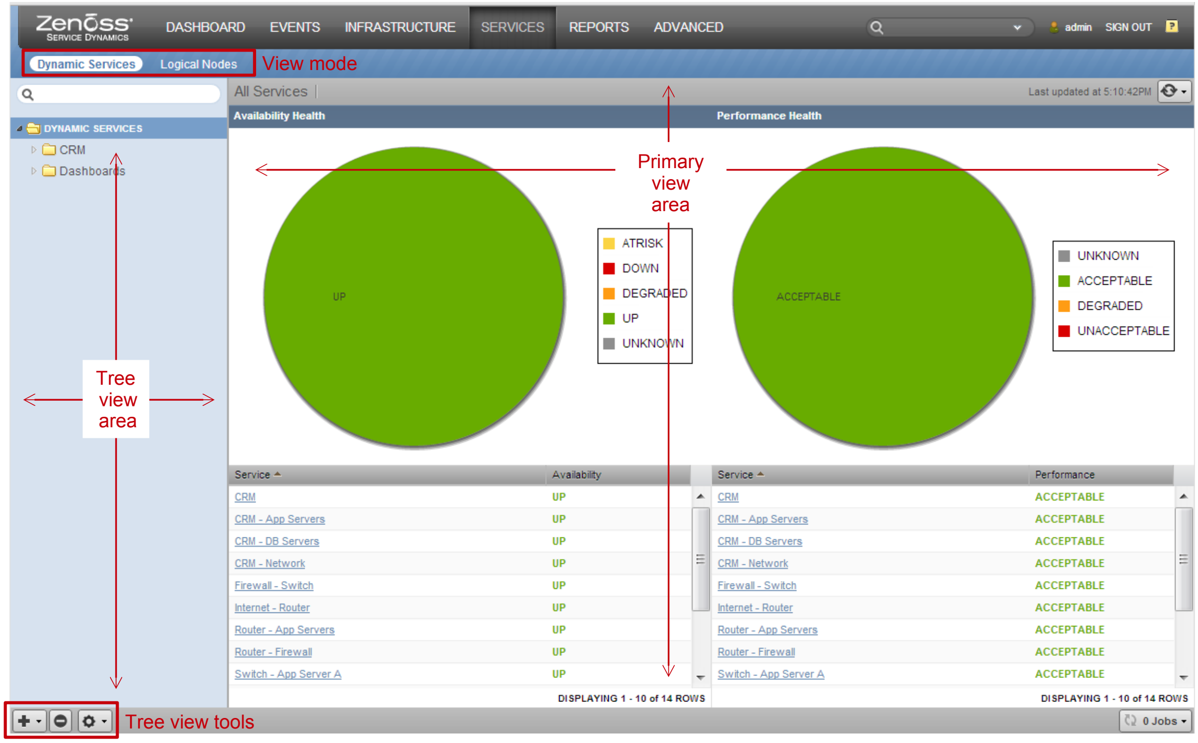
1. Log in to the Zenoss platform browser interface as a user with ZenManager or Manager privileges.
2. Click **SERVICES**.
3. Remove the dynamic services and logical node.
   a. In the tree view, select the **Dashboard** organizer.
   b. At the bottom of the tree view, click the **Delete** button.
   c. In the tree view, select the **CRM - Development** organizer.
   d. At the bottom of the tree view, click the **Delete** button.
   e. Select the **Logical Nodes** view mode.
   f. In the tree view, select the **CRM - Development** organizer.
   g. At the bottom of the tree view, click the **Delete** button.
4. Remove the fake devices.
   a. In the Zenoss platform browser interface, select **INFRASTRUCTURE**.
   b. In the tree view, select **FakeImpactDevices**.
   c. At the bottom of the tree view, click the **Delete** button.

# Browser interface reference 3

## Service Impact home page

The home page of the Service Impact feature of Zenoss Service Dynamics displays the health summaries of all services.

From the home page, you can view information in two modes: **Dynamic Services** and **Logical Nodes**. In both view modes, the Service Impact home page includes the tree view area and its tools, and the primary view area.



| Service | Availability |
|---|---|
| CRM | UP |
| CRM - App Servers | UP |
| CRM - DB Servers | UP |
| CRM - Network | UP |
| Firewall - Switch | UP |
| Internet - Router | UP |
| Router - App Servers | UP |
| Router - Firewall | UP |
| Switch - App Server A | UP |

DISPLAYING 1 - 10 of 14 ROWS

| Service | Performance |
|---|---|
| CRM | ACCEPTABLE |
| CRM - App Servers | ACCEPTABLE |
| CRM - DB Servers | ACCEPTABLE |
| CRM - Network | ACCEPTABLE |
| Firewall - Switch | ACCEPTABLE |
| Internet - Router | ACCEPTABLE |
| Router - App Servers | ACCEPTABLE |
| Router - Firewall | ACCEPTABLE |
| Switch - App Server A | ACCEPTABLE |

DISPLAYING 1 - 10 of 14 ROWS

### Tree view area

The tree view area displays service nodes and logical nodes in alphabetical order. You may create organizers and order them as you wish in both view modes. In addition, you may move service nodes and logical nodes into organizers by dragging them in the tree view.

| Tool | Menu | |
|---|---|---|
| | **Dynamic Services view mode** | **Logical Nodes view mode** |
| + ▾ | **Add Dynamic Service**<br>**Add Dynamic Service Organizer** | **Add Logical Node**<br>**Add Logical Node Organizer** |
| ⊖ | (no menu) | (no menu) |
| ⚙ ▾ | **View and Edit Details**<br>**Clone Service...**<br>**Export Selected** | (no menu) |

## Primary view area

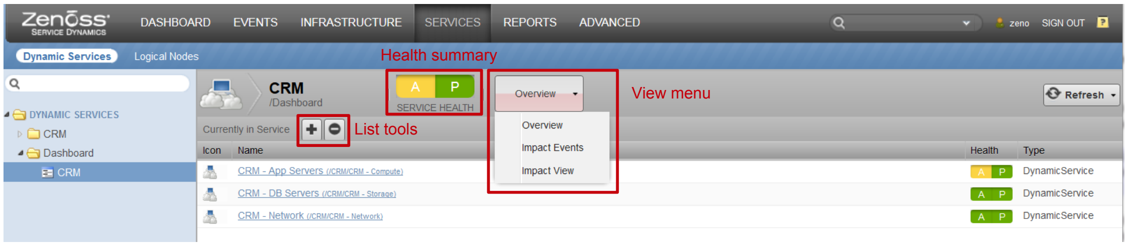
The contents of the primary view area depend on the view mode and the item selected in the tree view area.

| View mode | Tree view selection | Primary view contents |
|---|---|---|
| **Dynamic Services** | **DYNAMIC SERVICES** (the root organizer) | The availability and performance health summaries of all services. |
| | An organizer | The availability and performance health summaries of the services contained in the organizer. |
| | A dynamic service | The **Overview** of the selected service. From this view, you may select **Impact Events** or **Impact View**. |
| **Logical Nodes** | ■ **LOGICAL NODES** (the root organizer)<br>■ An organizer | A blank logical node details view. |
| | A logical node | The details view of the selected logical node. There are no other views associated with logical nodes. |

# Overview

The **Overview** view provides details about a service node, including the list of nodes that are associated with it.

The following example shows a service node **Overview**, with key features highlighted, and the view menu selected.



Health summary

View menu

List tools

| Icon | Name | Health | Type |
|---|---|---|---|
| | CRM - App Servers (/CRM/CRM - Compute) | A P | DynamicService |
| | CRM - DB Servers (/CRM/CRM - Storage) | A P | DynamicService |
| | CRM - Network (/CRM/CRM - Network) | A P | DynamicService |

### List tools

The list tools add or remove nodes from a service node.

| Tool | Function |
|---|---|
| ➕ | Display the **Add to Service** dialog. |
| ⊖ | Remove a selected node from the service node. |

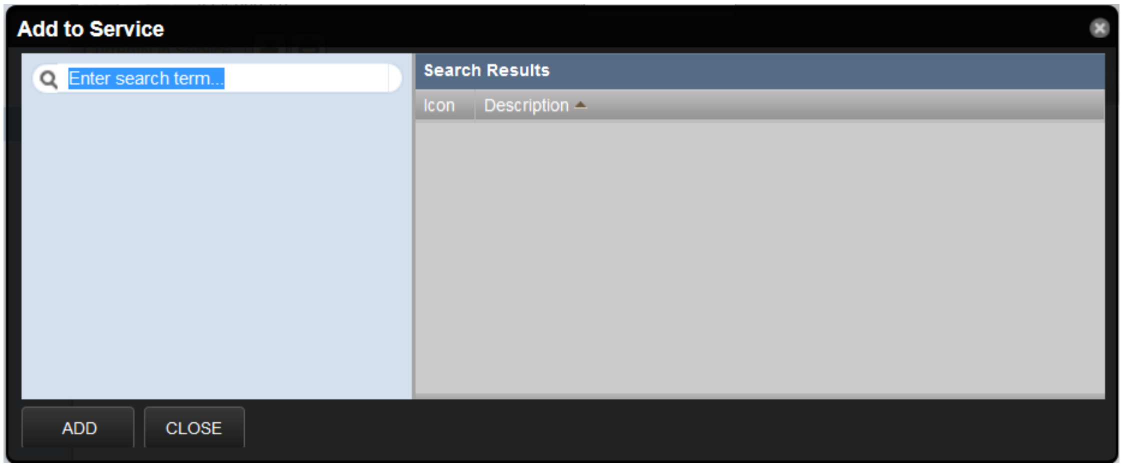
### Health summary

The health summary shows a service node's color-coded availability and performance states. The same summary information is provided for each node associated with a service node, in the **Health** column.

| Availability ("A") states | Performance ("P") states |
|---|---|
| 🟨 ATRISK<br>🟥 DOWN<br>🟧 DEGRADED<br>🟩 UP<br>◼ UNKNOWN | ◼ UNKNOWN<br>🟩 ACCEPTABLE<br>🟧 DEGRADED<br>🟥 UNACCEPTABLE |

## Add to Service dialog

The interface for finding and adding nodes to a service node.



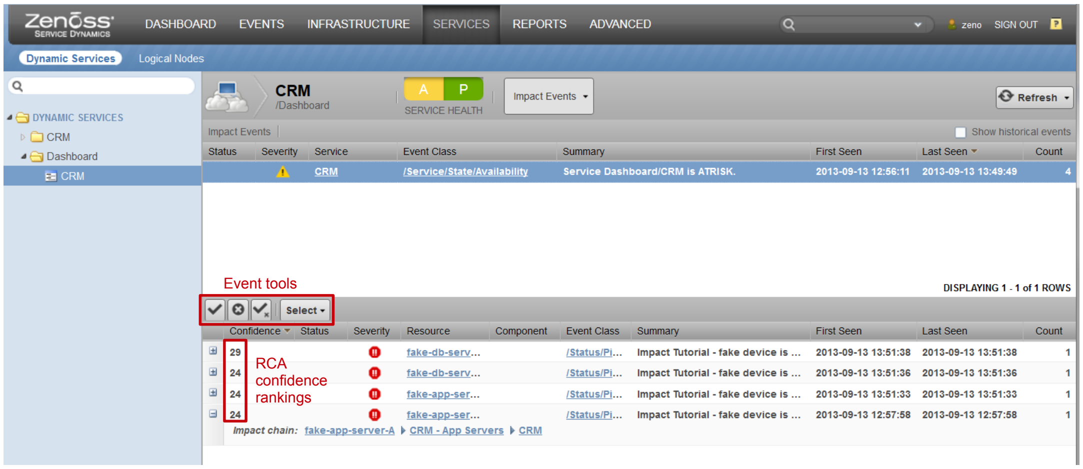
Enter the name of the node to find in the search field. After 3 characters, Service Impact displays matching nodes.

## Impact Events

The **Impact Events** view shows summary and detail information about events that are affecting a service node.

The following example shows an **Impact Events** view, with key features highlighted.

Note The last event in the details area is expanded, showing the **Impact chain**, which is the hierarchy of nodes associated with the event.



| Status | Severity | Service | Event Class | Summary | First Seen | Last Seen | Count |
|---|---|---|---|---|---|---|---|
| | ⚠ | CRM | /Service/State/Availability | Service Dashboard/CRM is ATRISK. | 2013-09-13 12:56:11 | 2013-09-13 13:49:49 | 4 |

DISPLAYING 1 - 1 of 1 ROWS

| | Confidence | Status | Severity | Resource | Component | Event Class | Summary | First Seen | Last Seen | Count |
|---|---|---|---|---|---|---|---|---|---|---|
| ⊞ | 29 | | ● | fake-db-serv... | | /Status/Pi... | Impact Tutorial - fake device is ... | 2013-09-13 13:51:38 | 2013-09-13 13:51:38 | 1 |
| ⊞ | 24 | | ● | fake-db-serv... | | /Status/Pi... | Impact Tutorial - fake device is ... | 2013-09-13 13:51:36 | 2013-09-13 13:51:36 | 1 |
| ⊞ | 24 | | ● | fake-app-ser... | | /Status/Pi... | Impact Tutorial - fake device is ... | 2013-09-13 13:51:33 | 2013-09-13 13:51:33 | 1 |
| ⊟ | 24 | | ● | fake-app-ser... | | /Status/Pi... | Impact Tutorial - fake device is ... | 2013-09-13 12:57:58 | 2013-09-13 12:57:58 | 1 |

*Impact chain:* fake-app-server-A ▸ CRM - App Servers ▸ CRM

### Event tools

The event tools provide options for manipulating the list of events.

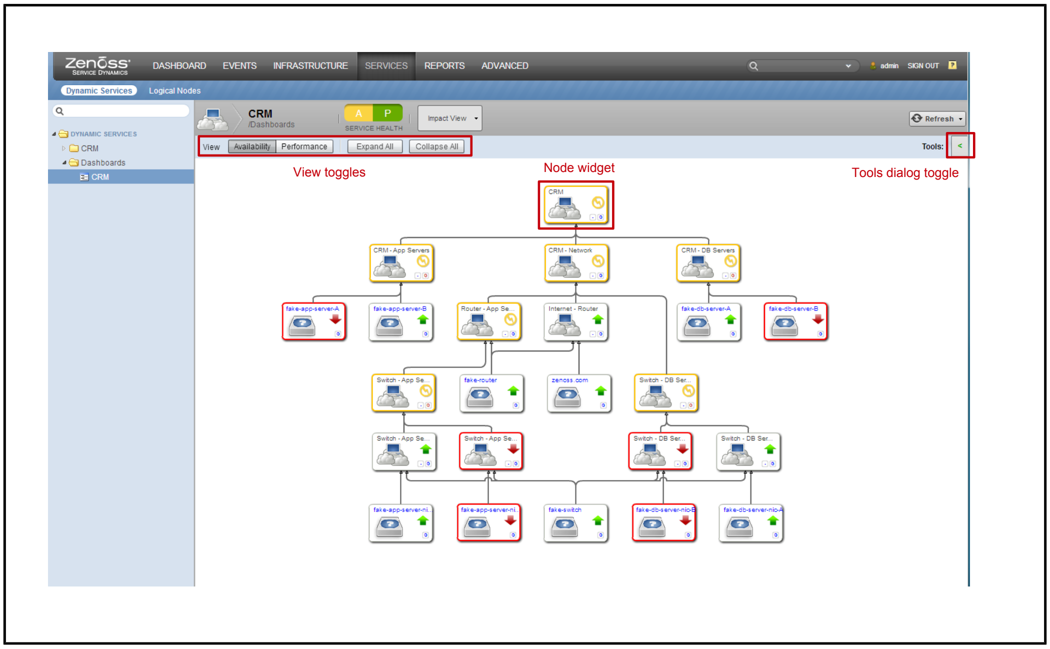
| Tool | Function |
|---|---|
| ✓ | Acknowledge the selected event. |
| ⊗ | Close the selected event. The event is moved to the archive. |
| ✓ₓ | Undo acknowledgement of the selected event. |
| Select ▾ | Display the event selection menu.<br>**Select All**<br>Select all events in the list.<br>**None**<br>Deselect all of the selected events. |

### Root cause analysis (RCA) confidence rankings

The second column of the details list contains the event's confidence ranking. Service Impact knows which nodes affect which service nodes, and automatically performs root cause analysis when an event occurs. The analysis yields a probability value that an event is the cause of the service node's current state. Often, more than one event contributes to a state, so the confidence rankings allow you to focus resources on the most likely cause right away.

## Impact View

The **Impact View** displays the interactive graph of a service node.



### View toggles

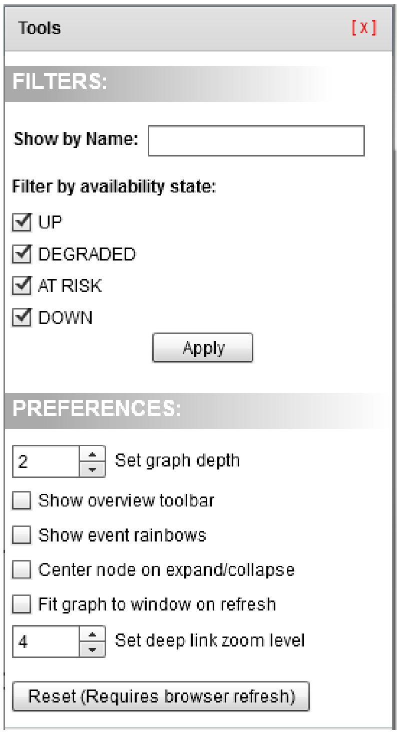
| Toggle | Function |
|---|---|
| Availabilty | Display availabilty states in the interactive graph. |
| Performance | Display performance states in the interactive graph. |
| Expand all | Display all levels of the interactive graph. |
| Collapse all | Display the minimum number of levels of the interactive graph. |

## Tools dialog

The **Tools** dialog provides controls for features of interactive graphs.

The tools dialog provides options for

- selecting nodes to display (FILTERS)
- configuring display options (PREFERENCES)



Tools [x]

FILTERS:

Show by Name:

Filter by availability state:

☑ UP
☑ DEGRADED
☑ AT RISK
☑ DOWN

Apply

PREFERENCES:

2 Set graph depth
☐ Show overview toolbar
☐ Show event rainbows
☐ Center node on expand/collapse
☐ Fit graph to window on refresh
4 Set deep link zoom level

Reset (Requires browser refresh)

## FILTERS

Select nodes to display in the interactive graph, by name and availability state.

| Option | Description |
|---|---|
| **Show by Name** | Specify a string to select nodes that contain the string in their names. Strings are case-sensitive; regular expressions are not supported. When no string is specified (the default) all nodes are selected. |
| **Filter by availability state** | Check availability states to select nodes that match the checked states. When all states are checked (the default) all nodes are selected. |

Click the **Apply** button to effect the selections.

## PREFERENCES

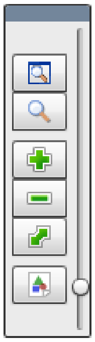
Configure display options for the interactive graph.

| Option | Description |
|---|---|
| **Set graph depth** | Increase or decrease the number of levels displayed in the interactive graph. |
| **Show overview toolbar** | Display the **Overview Toolbar**, which provides options for manipulating interactive graphs.<br>Note The default location of the **Overview Toolbar** is in the upper-right corner (behind the **Tools** dialog). Close the **Tools** dialog to see the **Overview Toolbar**. |
| **Show event rainbows** | Display counts of the critical, error, and warning events inside node widgets. |

| Option | Description |
|---|---|
| | Note Event rainbows may slow the rendering speed of large or complex graphs. |
| **Center node on expand/collapse** | When the branch toggle of a node is clicked, the node widget remains in the center of the interactive graph. |
| **Fit graph to window on refresh** | At each refresh of the interactive graph, adjust its size to fit inside the display area. |
| **Set deep link zoom level** | Set the initial zoom level when linking to a node from outside the graph. |
| **Reset (Requires browser refresh)** | Restore the defaults of all settings at the next refresh. |

## Overview Toolbar

The **Overview Toolbar** provides options for manipulating interactive graphs.



- The slider (on the right side) zooms the view in or out.
- The buttons (on the left side) are described in the next section.

The overview toolbar may be moved anywhere in an interactive graph. The default location is the upper-right corner.

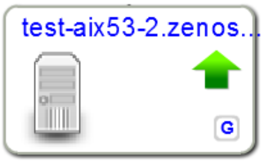
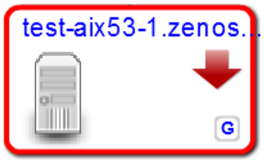
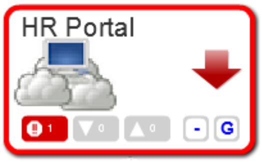
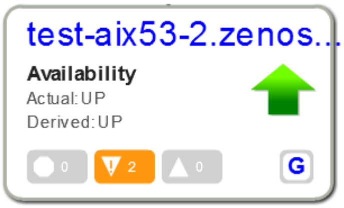
### Overview toolbar buttons

| Button | Description |
|---|---|
| ▣ | Toggle the overview, a miniature representation of the interactive graph. Drag the focus area to change the portion of the graph that appears in the view area. |
| 🔍 | Toggle the magnifier, which allows you to look more closely at a portion of the interactive graph. |
| ➕ | Zoom in to the interactive graph. |
| ➖ | Zoom out of the interactive graph. |
| ⤢ | Fit the interactive graph to the window. |
| 🖼 | Save an image of the current view. |

## Node widget

A node widget is the primary feature of interactive graphs.

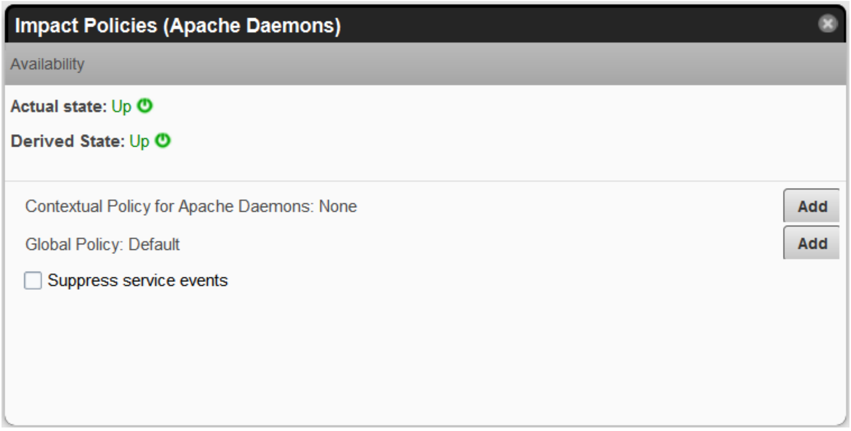
In an interactive graph, each node is represented by a rectangular widget with rounded corners.

| Widget | Description |
|---|---|
|  | This widget represents an AIX server named `test-aix53-2.zenoss.loc`. (The name is shortened to fit inside the widget.)<br>■ The computer icon represents the node's type, server.<br>■ The green arrow represents the node's availability state, UP.<br>■ The upper-case "G" in the lower-right corner is a link to the **Impact Policies** dialog. When a policy is configured on a node, the "G" is red.<br>■ When you click the name of a node, Zenoss platform displays the node's overview page. |
|  | This widget represents an AIX server named `test-aix53-1.zenoss.loc`. The red arrow and the red border of the widget represent the node's availability state, DOWN. |
|  | This widget represents a service node, and The **Show event rainbows** preference is selected.<br>■ The event rainbow is the 3 rounded rectangles at the bottom of the widget, showing the counts of critical, error, and warning events associated with the node. When you click an event rainbow rectangle, Zenoss platform displays the overview page of the node.<br>■ This widget includes the branch toggle, immediately to the left of the link to the **Impact Policies** dialog, at the bottom-right. The branch toggle collapses or expands the row of nodes immediately below the widget. |
|  test-aix53-2.zenos..<br>Availability<br>Actual: UP<br>Derived: UP<br>0<br>2<br>0<br>G | This widget is the same node as the first widget, with two differences.<br>■ The **Show event rainbows** preference is selected.<br>■ The view is zoomed in enough to fit the details of the availability states, in place of the node's icon. |

## Impact Policies dialog

The **Impact Policies** dialog provides options for defining global and contextual policies, by creating or editing state triggers.

Use the **Impact Policies** dialog to add or edit state triggers for contextual or global policies. If a custom state provider is associated with a node, an additional option is displayed in this dialog as well.



Impact Policies (Apache Daemons)

Availability

Actual state: Up

Derived State: Up

Contextual Policy for Apache Daemons: None

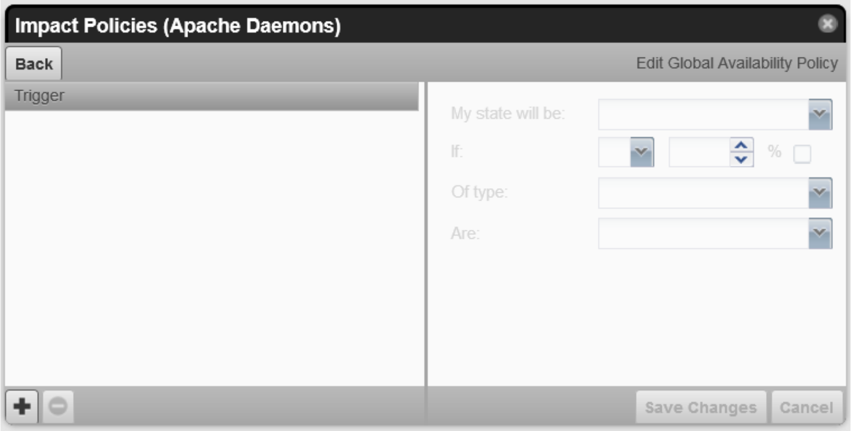
Global Policy: Default

□ Suppress service events

Check the **Suppress service events** box to prevent sending service events when changes affect a node. Choose this option when a service node is used solely to group child nodes.

## Edit...Policy tab

The **Edit...Policy** tab of the **Impact Policies** dialog is the interface for adding or editing state triggers for contextual or global policies.



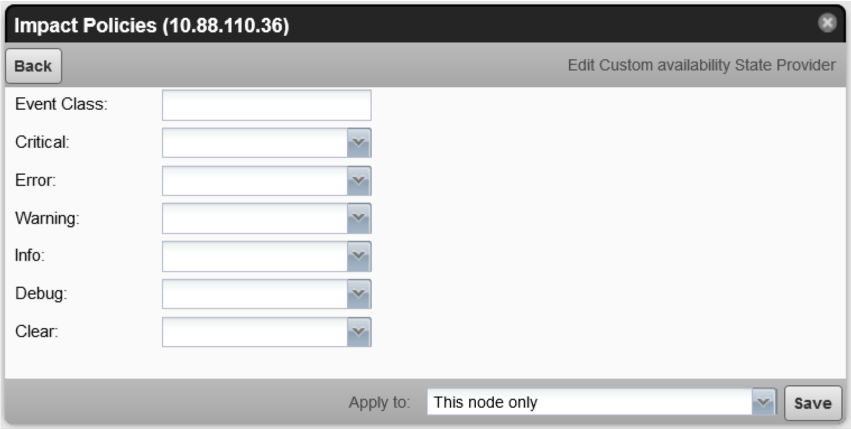
Impact Policies (Apache Daemons)

To add a trigger, click the **Add** button in the lower-left corner, modify the trigger fields on the right side, and the click the **Save Changes** button.

| Field | Description |
|---|---|
| **My state will be** | The new state, if the trigger applies. Selections are DOWN, DEGRADED, ATRISK, and UP. |
| **If** | The conditions that trigger a state change. To specify a percentage rather than an absolute value, click the percent (%) check box. |
| **Of type** | Restrict the type of child node to which the trigger applies. Types other than **Any** are exclusive. |
| **Are** | The state of child nodes that cause asn evaluation of this trigger. Selections are DOWN, DEGRADED, ATRISK, and UP. |

## Edit Custom...State Provider

The **Custom...State Provider** tab of the **Impact Policies** dialog is the interface for adding or editing state triggers for custom device and component objects.



Impact Policies (10.88.110.36)

Back

Edit Custom availability State Provider

Event Class:

Critical:

Error:

Warning:

Info:

Debug:

Clear:

Apply to: This node only

Save

In the **Event Class** field, specify the Zenoss platform event class to monitor.

In the event severity fields, specify the state for this node, if the event severity is observed. Selections are DOWN, DEGRADED, ATRISK, and UP.

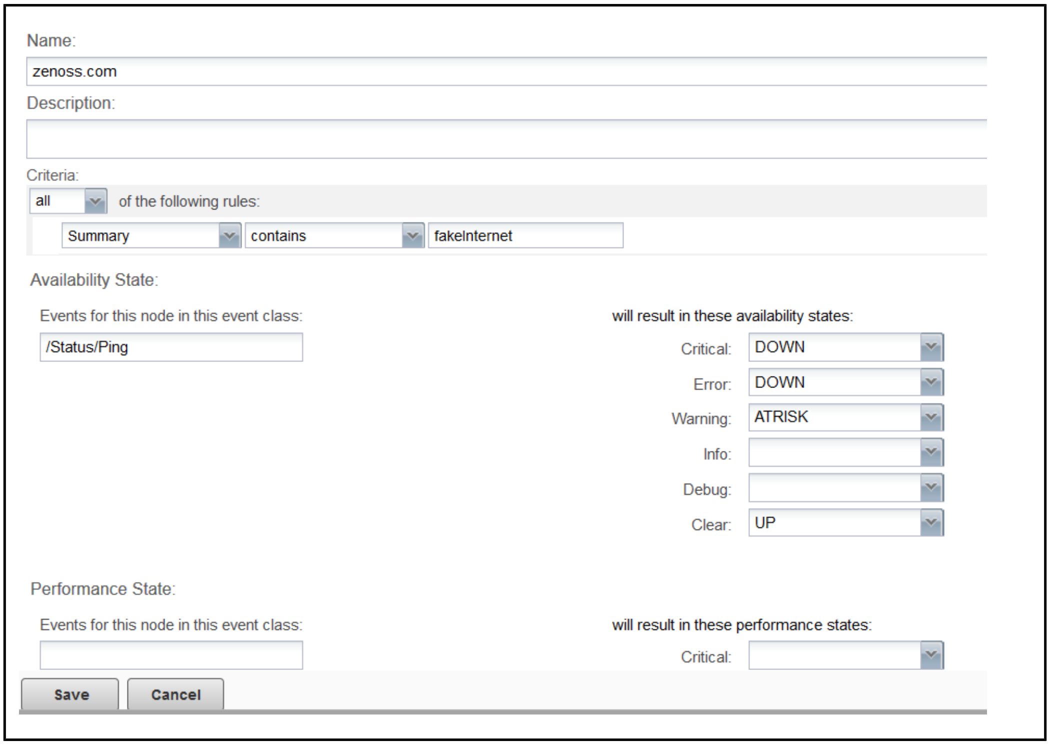
In the **Apply to** field, specify the nodes to which the state applies.

## Logical node details view

The logical node details view allows you to create and edit logical nodes.

A logical node is a customizable object that captures specific Zenoss platform event states. Logical nodes allow you to represent resources or services not monitored by Zenoss platform, or, for resources that are monitored by Zenoss platform, to capture event states from arbitrary classes.

A logical node may be a child of a service node, but no other node type.



Name: zenoss.com

Description:

Criteria: all of the following rules:

Summary contains fakeInternet

Availability State:

Events for this node in this event class: /Status/Ping

will result in these availability states:

| | |
|---|---|
| Critical: | DOWN |
| Error: | DOWN |
| Warning: | ATRISK |
| Info: | |
| Debug: | |
| Clear: | UP |

Performance State:

Events for this node in this event class:

will result in these performance states:

Critical:

Save Cancel

### Criteria

The **Criteria** rules allow you to define event triggers, and to associate the triggers with availability and performance states.

### Availability State

Use the **Events for this node in this event class** field to specify the event class or subclass associated with the trigger.

The following examples illustrate how to specify event classes.

**`/Status`**

Only the `/Status` class.

**`/Status/Web`**

Only the `Web` subclass of `/Status`

**`/Status/`**

The `/Status` class and all of its subclasses.

Use the **will result in these availability states** fields to map event severity levels to availability states.

### Performance State

The fields and options in this area of the page differ only in that the mapping of event severity levels to performance states uses different states.

# Installing Service Impact 4.2.6

## Overview

Zenoss recommends installing the Service Impact server on its own, dedicated host, rather than the Zenoss platform master host, to avoid contention with core Zenoss platform services. Impact is packaged for dedicated-host deployment, and Zenoss recommends it as a best practice.

The instructions in this section are valid for both dedicated-host and single-host installations. Installing the Service Impact server on the Zenoss platform master host can be appropriate for development and testing environments, and in very limited cases, production environments. If you are unsure which deployment strategy is best for your environment, consult Zenoss Support.

Zenoss recommends choosing a network location for the Service Impact server host that enables low-latency SSH and API access from the Zenoss platform master host. A ZenPack containing the **zenimpactstate** daemon is installed on the Zenoss platform master host, to enable communication with the server.

The Service Impact server requires a host with the following features.

- Red Hat Enterprise Linux or CentOS Linux, version 5 or 6
- 4 GB main memory
- 8 CPU cores
- 50 MB disk space for software
- 50 MB disk space for log files
- 1-20 GB disk space for database files

In addition, you may mount a separate filesystem for the Service Impact server database, if desired.

This release requires Zenoss platform version 4.2.4, plus the most recent recommended patch set (RPS), or a more recent version of Zenoss platform 4.2, and its most recent RPS. For more information about compatibility with Zenoss platform, contact your Zenoss representative.

Note Once the Service Impact server is installed, Zenoss platform and Service Impact are interdependent. Zenoss platform is unable to perform modeling if the Service Impact server is unavailable. However, Zenoss platform is able to continue to perform monitoring.

## Preparing the Zenoss platform version 4.2.4 master host

1 Log in to the Zenoss platform master host as `zenoss`.

2 Verify that ZenUp is installed, and Zenoss platform is registered.
`zenup status`

- If the result is `zenup: command not found`, ZenUp is not installed. Stop this procedure and install ZenUp.
- If the result is similar to the following example, continue this procedure.

```
Product: zenoss-resmgr-4.2.4 (id = zenoss-resmgr-4.2.4)
Home: /opt/zenoss
Revision: 179
Updated On: Fri Aug 12 13:42:36 2013
```

The value of the `Revision` field is the recommended patch set (RPS) number.

3 Log in to the *Zenoss Support* site.

4. Compare the RPS number returned by the `zenup status` command with the revision number of the latest available RPS for Zenoss platform.

   - If the revision number is less than 179, stop this procedure, and download and install the latest RPS.
   - If the revision number is 179 or greater, continue this procedure.
5. (Upgrade only) If you have `cron` jobs to manage Service Impact log files, delete them. The new version manages its log files efficiently.
6. Append the following lines to `$ZENHOME/etc/global.conf`.

   ```
   impactport 8083
   impacthost Impact-Host-FQDN
   impactuser zenossimpact
   ```

   Replace *Impact-Host-FQDN* with the fully-qualified domain name of the Service Impact server host.

   Note Zenoss recommends using a fully-qualified domain name for the `impacthost` property in all deployment scenarios.

## Preparing the Zenoss platform version 4.2.5 master host

1. Log in to the Zenoss platform master host as `zenoss`.
2. Append the following lines to `$ZENHOME/etc/global.conf`.

   ```
   impactport 8083
   impacthost Impact-Host-FQDN
   impactuser zenossimpact
   ```

   Replace *Impact-Host-FQDN* with the fully-qualified domain name of the Service Impact server host.

   Note Zenoss recommends using a fully-qualified domain name for the `impacthost` property in all deployment scenarios.

## Preparing the Service Impact server host

1. Log in to the Service Impact server host as `root`, or as a user with superuser privileges.
2. Configure firewall ports.

   *Firewall port requirements* on page 37 identifies the port requirements of the Service Impact server.
3. Disable Security-Enhanced Linux (SELinux).

   a. Disable enforcing mode permanently.

   ```
   /bin/sed -i.bak -e 's/^SELINUX=.*/SELINUX=disabled/g' \
     /etc/selinux/config
   ```

   b. Disable enforcing mode immediately.
   `/bin/echo 0 > /selinux/enforce`
4. Determine whether incompatible Java packages are installed.

   ```
   rpm -qa | egrep -i '(jdk|jre|java)'
   ```

   - If the command returns no result, continue to the next step.
   - If the command returns a result, remove the packages.

   ```
   yum -y remove $(rpm -qa | egrep -i '(jdk|jre|java)')
   ```

**5** Install the Oracle JRE.

- If you are installing the Service Impact server on its own, dedicated host, continue with this step.
- If you are installing the Service Impact server on the Zenoss platform master host, proceed to *Installing Java 7 on the Zenoss platform master host* on page 37.

**a** Download the RPM file of Oracle Java SE Runtime Environment 7u51.

The file to download is `jre-7u51-linux-x64.rpm`, located on the *Java SE 7 Downloads* page.

**b** Install the package.

```
yum -y --nogpgcheck localinstall jre-7u51-linux-x64.rpm
```

## Firewall port requirements

| Daemon | Port/App. | Protocol | Direction | Source/Dest. | Notes |
|---|---|---|---|---|---|
| sshd | 22/SSH | TCP | IN | Zenoss platform master host | Required. Used for backup, restore, and reset operations. |
| avahi | 53/DNS | UDP | OUT | DNS server(s) | Very strongly recommended. |
| ntpd | 123/NTP | UDP | IN/OUT | Zenoss servers, network time servers | Very strongly recommended. |
| bind | 161/SNMP | UDP | IN | `zenperfsnmp` on Zenoss platform master host | Recommended. |
| bind | 162/ SNMPTrap | UDP | OUT | `zentrap` on Zenoss platform master host | Recommended. |
| rabbitmq | 5672/ AMQP | TCP | IN | Service Impact server | Required. |
| java | 12345/RMI | TCP | OUT | `zenjmx` on Service Impact server host | Recommended. |
| jetty | 8083/HTTP | TCP | IN/OUT | `zenwebserver` on Zenoss platform master host, and `zenhub`, on any host it is present | Required. Also, any host that has ZODB access needs to use this port. |
| Neo4J backup | 6362/HTTP | TCP | IN/OUT | `zenbackup` and `zenrestore` on Zenoss platform master host | Required. |

# Installing Java 7 on the Zenoss platform master host

Note Perform this procedure only if you are installing or upgrading the Service Impact server on the Zenoss platform master host.

Zenoss platform version 4.2.x requires Java 6, and this release of the Service Impact server requires Java 7. This procedure installs Java 7 alongside Java 6.

**1** Log in to the Zenoss platform master host as `root`, or as a user with superuser privileges.

**2** Download the RPM file of Oracle Java SE Runtime Environment 7u51.

The file to download is `jre-7u51-linux-x64.rpm`, located on the *Java SE 7 Downloads* page.

3 Install the Java 7 package, except the jexec utility.

```
rpm -ivh --excludepath /etc/init.d/jexec jre-7u51-linux-x64.rpm
```

The Service Impact server does not use the jexec utility, so the Java 6 version can remain in place.

4 Set the default version to Java 6.

- For Zenoss platform version 4.2.4, use the following commands.

```
rm /usr/java/default; ln -s /usr/java/jre1.6.0_31/ /usr/java/default
```

- For Zenoss platform version 4.2.5, use the following commands.

```
rm /usr/java/default; ln -s /usr/java/jre1.6.0_45/ /usr/java/default
```

5 Create a link to Java 7 for the Service Impact server.

```
ln -s /usr/java/jre1.7.0_51 /usr/java/zenoss_impact
```

## Installing the Service Impact server

1 Log in to the Service Impact server host as root, or as a user with superuser privileges.

2 Download the Service Impact server RPM file from the *Zenoss Support* site.

Contact your Zenoss representative for login credentials.

3 Install the Service Impact server.

```
yum -y --nogpgcheck localinstall zenoss_impact*
```

The installation process adds a new user, zenossimpact, and places the Service Impact server software in /opt/zenoss_impact.

4 Note Perform this step only if you are installing the Service Impact server on the Zenoss platform master host.

Optional: Change the value of the JAVA_HOME variable that the Service Impact server uses.

```
sed -i 's|JAVA_HOME=.*$|JAVA_HOME=/usr/java/zenoss_impact|' \
  /etc/default/zenoss_impact
```

5 Mount a separate filesystem at /opt/zenoss_impact/var/db, if desired.

The Service Impact server database can become very large, and filesystem performance affects the overall performance of Service Impact.

6 Start a shell as user zenossimpact.

```
su - zenossimpact
```

7 Configure connection properties.

a Change directory to $IMPACT_HOME/etc.

```
cd $IMPACT_HOME/etc
```

This directory contains the Service Impact configuration files.

b Create one variable for the fully-qualified domain name of the Zenoss platform master host, and another for the password of the Zenoss platform admin user.

```
z_master=Master-Host-FQDN
admin_pw=Admin-Password
```

c Edit the RabbitMQ properties file.

```
sed -i.bak -e '/^#amqphost/ s/#//' \
  -e '/^amqphost/ s/localhost/'$z_master'/' \
  ./zenoss-dsa-amqpconf.properties
```

d. Edit the Service Impact properties file.

```
sed -i.bak -e '/^#dsa.amqp.uri/ s/#//' \
  -e '/^dsa.amqp.uri/ s/localhost/'$z_master'/' \
  -e '/^#dsa.zenoss.host/ s/#//' \
  -e '/^dsa.zenoss.host/ s/localhost/'$z_master'/' \
  -e '/^#dsa.zenoss.password/ s/#//' \
  -e '/^dsa.zenoss.password/ s/=zenoss/='$admin_pw'/' \
  ./zenoss-dsa.properties
```

# Configuring authentication

The `zenoss` user on the Zenoss platform master host requires access to the Service Impact server host as user `zenossimpact`, through Secure Shell (SSH). You may configure key-based authentication or password authentication to enable access (but not both).

Note When Zenoss platform is started for the first time, the Distributed Collector ZenPack generates a new, unique key pair for user `zenoss`, with the OpenSSH `ssh-keygen` command. All remote hosts use the same SSH key pair for user `zenoss`.

For more information about the Distributed Collector ZenPack, refer to *Zenoss Service Dynamics Extended Monitoring*.

## Configuring key-based access

This procedure enables communication between the Zenoss platform master host and the Service Impact server host.

Note If you configure key-based access, do not configure password access.

1. Log in to the Service Impact server host as `root`, or as a user with superuser privileges.
2. Reset the password of user `zenossimpact`.
   `passwd -uf zenossimpact`
3. Start a shell as user `zenossimpact`.
   `su - zenossimpact`
4. Create a new password for user `zenossimpact`.
   `passwd`

   The `passwd` command prompts you for the new pasword.
5. Create a directory and file to hold public keys.
   a. Create the directory.
      `mkdir $HOME/.ssh`
   b. Set the directory access permissions.
      `chmod 700 $HOME/.ssh`
   c. Create the file.
      `touch $HOME/.ssh/authorized_keys`
   d. Set the file access permissions.
      `chmod 600 $HOME/.ssh/authorized_keys`
6. Log in to the Zenoss platform master host as `zenoss`.
7. Copy the `zenoss` user's public key to the Service Impact server host.
   a. Enter the following command. Substitute the fully-qualified domain name of the Service Impact server host for *Impact-Host-FQDN*.

```
cat $HOME/.ssh/id_rsa.pub | ssh -l zenossimpact Impact-Host-FQDN \
    "cat - >> /home/zenossimpact/.ssh/authorized_keys"
```

The `ssh` command prompts you to confirm the connection.

**b** Enter `yes`.

The `ssh` command adds the Service Impact server host to the known hosts file of the `zenoss` user on the master host, and then prompts you for the password of the `zenossimpact` user on the Service Impact server host.

**c** Enter the password.

The `ssh` command invokes `cat` on the Service Impact server host to append the public key of user zenoss on the Zenoss platform host to the authorized keys file of user `zenossimpact`.

### Configuring password access

This procedure enables communication between the Zenoss platform master host and the Service Impact server host.

Note If you configure password access, do not configure key-based access.

1. Log in to the Service Impact server host as `root`, or as a user with superuser privileges.
2. Reset the password of user `zenossimpact`.
   `passwd -uf zenossimpact`
3. Start a shell as user `zenossimpact`.
   `su - zenossimpact`
4. Create a new password for user `zenossimpact`.
   `passwd`

   The `passwd` command prompts you for the new pasword.
5. Log in to the Zenoss platform master host as `zenoss`.
6. Append the following line to `$ZENHOME/etc/global.conf`. Substitute the password of the `zenossimpact` user on the Service Impact server host for *Remote-Password*.
   `impactpassword` *`Remote-Password`*
7. Start a shell on the Service Impact server host as user `zenossimpact`. Substitute the fully-qualified domain name of the Service Impact server host for *Impact-Host-FQDN*.
   `ssh -l zenossimpact` *`Impact-Host-FQDN`*

   The `ssh` command prompts you to confirm the connection.
8. Enter `yes`.

   The `ssh` command adds the Service Impact server host to the known hosts file of the `zenoss` user on the master host, and then prompts you for the password of the `zenossimpact` user on the Service Impact server host.
9. Enter the password.

   The shell is started on the Service Impact server host.
10. Log out of the Service Impact server host: `exit`

## Starting Service Impact processing

1. Log in to the Service Impact server host as `root`, or as a user with superuser privileges.
2. Start the Service Impact server.
   `service zenoss_impact start`
3. Verify the Service Impact server is started.
   `service zenoss_impact status`
4. Log in to the Zenoss platform master host as `zenoss`.
5. Download and install the Service Impact ZenPack.

*Installing and upgrading ZenPacks* on page 52 describes how to install ZenPacks.

Note This ZenPack includes a daemon, **zenimpactstate**. If you are using `$ZENHOME/etc/daemons.txt`, add **zenimpactstate** to the file. If you are upgrading from version 4.2.4, remove **zenimpactserver** and **zenimpactgraph** from `$ZENHOME/etc/daemons.txt`.

## Service Impact server configuration files

The `$IMPACT_HOME/etc` directory contains Service Impact server configuration files.

**`$IMPACT_HOME/etc/zenoss-dsa-amqpconf.properties`**

This file contains properties for the Service Impact host's connection to the RabbitMQ Server.

**`$IMPACT_HOME/etc/zenoss-dsa.env`**

This file contains the `JVM_ARGS` variable definition.

**`$IMPACT_HOME/etc/zenoss-dsa.properties`**

This file contains general Service Impact configuration properties, including properties for backups, log file management, and remote debugging.

Please consult with Zenoss Support before changing values in the preceding files or any other file in this directory.

# Upgrading to Service Impact 4.2.6

## Overview

You may upgrade Service Impact version 4.2.5 to Service Impact version 4.2.6. To upgrade to version 4.2.5, refer to the previous release of this document.

Zenoss recommends installing the Service Impact server on its own, dedicated host, rather than the Zenoss platform master host, to avoid contention with core Zenoss platform services. Impact is packaged for dedicated-host deployment, and Zenoss recommends it as a best practice.

The Service Impact server requires a host with the following features.

- Red Hat Enterprise Linux or CentOS Linux, version 5 or 6
- 4 GB main memory
- 8 CPU cores
- 50 MB disk space for software
- 50 MB disk space for log files
- 1-20 GB disk space for database files

In addition, you may mount a separate filesystem for the Service Impact server database, if desired.

This release requires Zenoss platform version 4.2.4, plus the most recent recommended patch set (RPS), or a more recent version of Zenoss platform 4.2, and its most recent RPS. For more information about compatibility with Zenoss platform, contact your Zenoss representative.

Note Zenoss recommends upgrading a development or testing environment before upgrading a production environment.

## Preparing to upgrade

1 Log in to the Zenoss platform master host as `zenoss`.

2 Stop Zenoss platform.

   - *Stopping Zenoss platform without remote collector or hub hosts* on page 50
   - *Stopping Zenoss platform with remote collector or hub hosts* on page 51

   At the end of both procedures, you are logged in to the Zenoss platform master host as `zenoss`.

3 Log in to the Service Impact server host as `root`, or as a user with superuser privileges.

4 Stop the Service Impact server.

```
service zenoss_impact stop
```

5 Verify the Service Impact server is stopped.

```
service zenoss_impact status
```

6 Upgrade the Oracle JRE.

   - If you are upgrading the Service Impact server on its own, dedicated host, continue with this step.
   - If you are upgrading the Service Impact server on the Zenoss platform master host, skip this step and proceed to the next section.

   a Download the RPM file of Oracle Java SE Runtime Environment 7u51.

   The file to download is `jre-7u51-linux-x64.rpm`, located on the *Java SE 7 Downloads* page.

   b Install the package.

```
yum -y --nogpgcheck localinstall jre-7u51-linux-x64.rpm
```

   c Verify the installation.

```
java -version
```

```
java version "1.7.0_51"
Java(TM) SE Runtime Environment (build 1.7.0_51-b13)
Java HotSpot(TM) 64-Bit Server VM (build 24.51-b03, mixed mode)
```

## Installing Java 7 on the Zenoss platform master host

**Note** Perform this procedure only if you are installing or upgrading the Service Impact server on the Zenoss platform master host.

Zenoss platform version 4.2.x requires Java 6, and this release of the Service Impact server requires Java 7. This procedure installs Java 7 alongside Java 6.

**1** Log in to the Zenoss platform master host as `root`, or as a user with superuser privileges.

**2** Download the RPM file of Oracle Java SE Runtime Environment 7u51.

The file to download is `jre-7u51-linux-x64.rpm`, located on the *Java SE 7 Downloads* page.

**3** Install the Java 7 package, except the `jexec` utility.

```
rpm -ivh --excludepath /etc/init.d/jexec jre-7u51-linux-x64.rpm
```

The Service Impact server does not use the `jexec` utility, so the Java 6 version can remain in place.

**4** Set the default version to Java 6.

- For Zenoss platform version 4.2.4, use the following commands.

```
rm /usr/java/default; ln -s /usr/java/jre1.6.0_31/ /usr/java/default
```

- For Zenoss platform version 4.2.5, use the following commands.

```
rm /usr/java/default; ln -s /usr/java/jre1.6.0_45/ /usr/java/default
```

**5** Create a link to Java 7 for the Service Impact server.

```
ln -s /usr/java/jre1.7.0_51 /usr/java/zenoss_impact
```

## Upgrading the Service Impact server

**1** Log in to the Service Impact server host as `root`, or as a user with superuser privileges.

**2** Verify the Service Impact server is stopped.

```
service zenoss_impact status
```

**3** Download the Service Impact server RPM file from the *Zenoss Support* site.

Contact your Zenoss representative for login credentials.

**4** Install the Service Impact server.

```
yum -y --nogpgcheck localinstall zenoss_impact*
```

**5** **Note** Perform this step only if you are upgrading the Service Impact server on the Zenoss platform master host.

Optional: Change the value of the `JAVA_HOME` variable that the Service Impact server uses.

```
sed -i 's|JAVA_HOME=.*$|JAVA_HOME=/usr/java/zenoss_impact|' \
  /etc/default/zenoss_impact
```

**6** Edit the `zenoss-dsa.properties` file.

**a** Start a shell as user `zenossimpact`.

```
su - zenossimpact
```

**b** Change directory to `$IMPACT_HOME/etc`.

```
cd $IMPACT_HOME/etc
```

**c** Create one variable for the fully-qualified domain name of the Zenoss platform master host, and another for the password of the Zenoss platform `admin` user.

```
z_master=Master-Host-FQDN
admin_pw=Admin-Password
```

**d** Edit the file.

```
sed -i.bak -e '/^#dsa.amqp.uri/ s/#//' \
  -e '/^dsa.amqp.uri/ s/localhost/'$z_master'/' \
  -e '/^#dsa.zenoss.host/ s/#//' \
  -e '/^dsa.zenoss.host/ s/localhost/'$z_master'/' \
  -e '/^#dsa.zenoss.password/ s/#//' \
  -e '/^dsa.zenoss.password/ s/=zenoss/='$admin_pw'/' \
  ./zenoss-dsa.properties
```

**7** Return to the `root` user shell.

```
exit
```

**8** Start the Service Impact server.

```
service zenoss_impact start
```

**9** Verify the Service Impact server is started.

```
service zenoss_impact status
```

## Upgrading the Service Impact ZenPack

**1** Log in to the Zenoss platform master host as `zenoss`.

**2** Download the Service Impact ZenPack file from the *Zenoss Support* site.

Contact your Zenoss representative for login credentials.

**3** Download and install the Service Impact ZenPack.

*Installing and upgrading ZenPacks* on page 52 describes how to install ZenPacks.

# Exporting and Importing Service Organizers

## Overview

Exporting and Importing Service Organizers allows you to migrate and reconcile Impact service organizers between different instances of Resource Manager. For example, you could export from a development environment and import into a production environment.

## Export a Dynamic Service Organizer or Single Service

Exporting a service organizer or single service is the first step in moving this information from one Resource Manager instance to another. For example, from a development environment to a production environment.

1 In the Resource Manager browser interface, select the **SERVICES** tab
2 In the tree view, select the Organizer or Service that you want to export. To export all services, select the **Dynamic Services** folder.
3 From the **Action** menu at the bottom of the tree view, select **Export Selected**.
4 Save the file to your local machine. The generated file name consists of the following four sections:

- `export` - (action)
- `impact` - (plugin)
- Name of the exported selection
- `YYYYMMDDHHMMSS` - (formatted date)

An example export file name is `export_impact_DynamicServices_20140224094302.graphml`

This export file is then used as the import file on the other Zenoss instance. For ease of understanding, this export file has been renamed `ImportFile.xml` in the following Import instructions.

## Initial Import of a Service Organizer

The initial import peforms an initial reconciliation between the two Impact environments. Output files are created and must be edited based on the outcome of the reconciliation.

1 Copy your Impact import file to the Resource Manager master host.
2 If desired, you can create new services, logical nodes, policies, and custom state providers by defining them in the import file and completing the import process.

Note Importing a service will not create devices or components.

3 Log in to the Resource Manager master host as `zenoss`.
4 Execute the initial import of the import file.

```
zenimpactimport -i ImportFile.xml
```

where `ImportFile.xml` is the name of your initial import file.

Note The `-i` flag indicates the initial import action.

5 This command imports and performs a first Reconcile attempt. The results are output to two files:

- `ImportFile.0001.txt` - this file name is sequentially numbered on subsequent reconciliation runs for historical purposes. Do not edit this file.
- `ImportFile.latest.txt` - this file should used as the source file for the next reconciliation run. Edits should be made in this file based on the output of the reconciliation.

## Reconciliation Run Output

The reconciliation run produces an output file with three distinct columns:

1. [ACTION]

   | Action | Description |
   |---|---|
   | UNRECONCILED | Could not find a matching device in Resource Manager. |
   | MAP | Found a matching element in Resource Manager. |
   | CREATE | Could not find a matching element in Resource Manager, but it can be created through import. |
   | DELETE | Found a matching element in Resource Manager and it is marked to be deleted in the import file. |
   | IGNORE | Do nothing with the imported node. |

2. [IMPORT NODE] - This column should not be edited in most cases. It contains the ID of the node that was imported by the input file.
3. [OPERATIONAL NODE] - This column is the DMD ID, Element Name, or GUID of the Resource Manager element that you want to map to the imported node.

   **Note** This column only applies to MAP and DELETE actions.

## Import Reconciliation Phase

The reconciliation phase of an import attempts to find devices in the import file and match them to devices found in Resource Manager.

Each import is referenced using an import file and the timestamp of when the information was imported. Reconciliation automatically uses the `ImportFile.latest.txt` file. This file can be edited and re-used as input for the next reconciliation run. The import is considered fully reconciled when all nodes have been satisfactorily reconciled. As such, there should be no UNRECONCILED nodes in the report.

1. Edit the `ImportFile.latest.txt` file that was generated during your initial import to resolve some of the discrepancies between the two Impact environments.

   **Note** You do not have to make all your changes at once. You can perform this reconciliation phase multiple times until you are satisfied with the outcome.

2. Start a reconciliation run to execute the commands in the edited `ImportFile.latest.txt` file.

   ```
   zenimpactimport -r ImportFile.xml
   ```

   where `ImportFile.xml` is the name of your initial import file.

   **Note** The `-r` flag indicates a reconciliation action. The script will look for the `ImportFile.latest.txt` file.

3. The result of this reconciliation will be a new `ImportFile.latest.txt` file and a sequentially numbered `ImportFile.0002.txt` file. If another reconciliation run is needed, use the new `ImportFile.latest.txt` and repeat these steps.
4. Once you have fully reconciled your import, you can continue to the commit phase where you will be able to see your changes take effect in Resource Manager.

Note If you want to delete the import from the system without going to the commit phase, use the `--abort` flag.

Note If you want to list all the active imports, use the `--list` flag.

## Import Commit Phase

Once you have fully reconciled your Impact import, you are ready to commit the changes to the Operational Graph and see your changes take effect in Zenoss.

Committing your changes cannot be undone. Be sure that you have fully reconciled your import.

1 Run the reconciliation command with the `--commit` flag.

```
zenimpactimport -r ImportFile.xml --commit
```

where `ImportFile.xml` is the name of your initial import file.

Note The `--commit` flag indicates that after the reconciliation action, which takes the `ImportFile.xml` and the edited `ImportFile.latest.txt` as input, the changes will be committed.

2 You will see your Services, Logical Nodes, Policies, and Custom State Providers displayed under the **SERVICES** tab in Resource Manager. These items are now editable.

3 If you realize that you need to make additional changes after committing, you will need to start a new export/ import procedure. You will not be able to edit the old `ImportFile.latest.txt`.

# Additional administrative tasks

## Starting Service Impact

Note Zenoss platform is dependent on Service Impact. Start Zenoss platform after starting the Service Impact server.

1. Log in to the Service Impact server host as `root`, or as a user with superuser privileges.
2. Start the Service Impact server.
   ```
   service zenoss_impact start
   ```

## Stopping Service Impact

Note Zenoss platform is dependent on Service Impact. Stop Zenoss platform before stopping the Service Impact server.

1. Stop Zenoss platform.
   - *Stopping Zenoss platform without remote collector or hub hosts* on page 50
   - *Stopping Zenoss platform with remote collector or hub hosts* on page 51

   At the end of both procedures, you are logged in to the Zenoss platform master host as `zenoss`.
2. Log in to the Service Impact server host as `root`, or as a user with superuser privileges.
3. Stop the Service Impact server.
   ```
   service zenoss_impact stop
   ```

## Backing up Service Impact

Zenoss platform and Service Impact may be up or down when a backup is performed.

Prerequisite: The Service Impact database path specified in `$IMPACT_HOME/etc/zenoss-dsa.properties` is the default value, `var/db`.

1. Log in to the Zenoss platform master host as `zenoss`.
2. Start the backup.
   ```
   zenbackup
   ```
   - The `zenbackup` command retrieves a copy of the Service Impact database from the Service Impact server host, and backs up Zenoss platform, as well.
   - For more information about `zenbackup`, refer to *Zenoss Service Dynamics Extended Monitoring*.

   ```
   INFO:zenbackup:Getting ZEP dbname, user, password, port from
    configuration files.
   INFO:zenbackup:Backing up the events database.
   INFO:zenbackup:Backup of events database completed in 0 seconds.
   INFO:zenbackup:Not backing up event indexes - it is currently running.
   INFO:zenbackup:Backing up config files.
   INFO:zenbackup:Backup of config files completed.
   INFO:zenbackup:Backing up the ZODB.
   INFO:zenbackup:Backup of ZODB database completed in 10 seconds.
   INFO:zenbackup:Backing up ZenPacks.
   INFO:zenbackup:Backup of ZenPacks completed.
   INFO:zenbackup:Backing up bin dir.
   INFO:zenbackup:Backup of bin completed.
   timestamp INFO zenbackup: Backing up ZenPack contents.
   ```

```
timestamp INFO zen.catalogservice: Performing an online backup of
 zencatalogservice.
timestamp INFO zenbackup: Backup of zencatalogservice complete.
timestamp INFO zenbackup: ZenPacks.zenoss.Impact backing up zenoss-
impact.example.com
timestamp INFO zenbackup: ZenPacks.zenoss.Impact back up succeeded
timestamp INFO zenbackup: Backup of ZenPack contents complete.
timestamp INFO zenbackup: Backup of ZenPacks completed in 7 seconds.
timestamp INFO zenbackup: Backing up performance data (RRDs).
timestamp INFO zenbackup: Backup of performance data completed in 0
 seconds.
timestamp INFO zenbackup: Packaging backup file.
timestamp INFO zenbackup: Backup written to /opt/zenoss/backups/
zenbackup_20130822.tgz
timestamp INFO zenbackup: Cleaning up staging directory /tmp/tmp4l8akq
timestamp INFO zenbackup: Backup completed successfully in 30 seconds.
```

Note If the backup requires more than 60 seconds, or if the backup fails due to commit lock timeout errors, refer to *Increasing the commit lock timeout interval* on page 49.

## Increasing the commit lock timeout interval

The Service Impact portion of a backup takes more than 60 seconds, or the backup fails due to commit lock timeout errors.

Prerequisite: The duration of the Service Impact portion of a backup. The duration is written to STDOUT by `zenbackup`.

1. Add 30 seconds to the duration of the Service Impact portion of a backup.
2. Log in to the Zenoss platform master host as `zenoss`.
3. Change the values of the commit lock timeout options to the result of the first step, in both of the configuration files in which it occurs.

   The default value of both options is 60 seconds.

| File | Option |
|---|---|
| `$ZENHOME/etc/global.conf` | `zodb-commit-lock-timeout` |
| `$ZENHOME/etc/zope.conf` | `commit-lock-timeout` |

# Restoring Service Impact

Zenoss platform and Service Impact must be down when a restore is performed.

Prerequisite: The backup image to restore was created by the `zenbackup` command.

1. Log in to the Service Impact server host as `root`, or as a user with superuser privileges.
2. Stop the Service Impact server.
   `service zenoss_impact stop`
3. Stop Zenoss platform.

   - *Stopping Zenoss platform without remote collector or hub hosts* on page 50
   - *Stopping Zenoss platform with remote collector or hub hosts* on page 51

   At the end of both procedures, you are logged in to the Zenoss platform master host as `zenoss`.
4. Start the restore.
   `zenrestore`

5 Log in to the Service Impact server host as `root`, or as a user with superuser privileges.

6 Start the Service Impact server.

```
service zenoss_impact start
```

7 Log in to the Zenoss platform master host as `zenoss`.

8 Start Zenoss platform.

```
zenoss start
```

## Increasing the Java heap size

The amount of main memory devoted to the Java process may influence the Service Impact server processing rate. If the Service Impact server is very slow or unable to operate, increase the heap size.

1 Log in to the Service Impact server host as `zenossimpact`.

2 Open the `$IMPACT_HOME/etc/zenoss-dsa.env` file with a text editor.

3 Change the value of the maximum heap size argument (`-Xmx`) from the default, `2048m`, to `3072m`.

Pre-release testing indicates that heap sizes larger than 3 GB do not improve performance.

4 Restart the Service Impact server.

```
$IMPACT_HOME/bin/zenoss_impact restart
```

## Removing Service Impact

**Note** This procedure removes the Service Impact server and its database. Create a backup of the data if you wish to maintain it.

1 Stop Zenoss platform.

- *Stopping Zenoss platform without remote collector or hub hosts* on page 50
- *Stopping Zenoss platform with remote collector or hub hosts* on page 51

At the end of both procedures, you are logged in to the Zenoss platform master host as `zenoss`.

2 Start the event server and catalog service daemons.

```
zeneventserver start; zencatalogservice start
```

3 Remove the Service Impact ZenPack.

```
zenpack --remove=ZenPacks.zenoss.Impact
```

4 Start Zenoss platform.

```
zenoss start
```

5 Log in to the Service Impact server host as `root`, or as a user with superuser privileges.

6 Stop the Service Impact server.

```
service zenoss_impact stop
```

7 Determine the Service Impact server package name.

```
rpm -qa | grep -i impact
```

8 Remove the Service Impact server package.

```
yum -y remove Package-Name
```

## Stopping Zenoss platform without remote collector or hub hosts

When a Zenoss platform deployment does not include remote hub or collector hosts, stopping all daemons is simple.

1 Log in to the Zenoss platform master host as `zenoss`.

2 Stop all Zenoss platform daemons.

```
zenoss stop
```

Occasionally, the `stop` command does not terminate all of the daemons.

3 Check for daemons that are not stopped.

```
pgrep -fl ${ZENHOME}
```

- If the command returns no result, Zenoss platform is stopped.
- If the command returns a result, stop the remaining daemons.

```
pkill -f ${ZENHOME}
```

## Stopping Zenoss platform with remote collector or hub hosts

Stop all daemons on remote Zenoss platform collector or hub hosts before stopping them on the master host.

1 Log in to the Zenoss platform master host as `zenoss`.

2 Stop the `zenwebserver` daemon.

```
zenwebserver stop
```

3 Stop Zenoss platform daemons on all collector hosts.

  a Log in to each collector host as `zenoss`.

```
ssh zenoss@Remote-Collector-Host
```

  b Stop all Zenoss platform daemons.

```
zenoss stop
```

  c Check for daemons that are not stopped.

```
pgrep -fl ${ZENHOME}
```

  - If the command returns no result, Zenoss platform is stopped.
  - If the command returns a result, stop the remaining daemons.

```
pkill -f ${ZENHOME}
```

4 Stop Zenoss platform daemons on all hub hosts.

  a Log in to each hub host as `zenoss`.

```
ssh zenoss@Remote-Hub-Host
```

  b Stop all Zenoss platform daemons.

```
zenoss stop
```

  c Check for daemons that are not stopped.

```
pgrep -fl ${ZENHOME}
```

  - If the command returns no result, Zenoss platform is stopped.

- If the command returns a result, stop the remaining daemons.

```
pkill -f ${ZENHOME}
```

5 Stop all Zenoss platform daemons on the master host.

a Log in to the Zenoss platform master host as `zenoss`.

b Stop all Zenoss platform daemons.

```
zenoss stop
```

c Check for daemons that are not stopped.

```
pgrep -fl ${ZENHOME}
```

- If the command returns no result, Zenoss platform is stopped.
- If the command returns a result, stop the remaining daemons.

```
pkill -f ${ZENHOME}
```

## Installing and upgrading ZenPacks

The ZenPack installation and update procedure requires stopping and starting Zenoss platform.

1 Log in to the Zenoss platform master host as `zenoss`.

2 Download the ZenPack or ZenPacks you wish to install or upgrade from the *Zenoss Support* site.

Contact your Zenoss representative for login credentials.

3 Stop Zenoss platform.

- *Stopping Zenoss platform without remote collector or hub hosts* on page 50
- *Stopping Zenoss platform with remote collector or hub hosts* on page 51

At the end of both procedures, you are logged in to the Zenoss platform master host as `zenoss`.

4 Start the event and catalog servers.

```
zeneventserver start; zencatalogservice start
```

5 Install the new ZenPacks.

```
zenpack --install ZenPack.Name-Version.egg
```

6 If a ZenPack introduces a new daemon, and you are using `$ZENHOME/etc/daemons.txt`, add the new daemon to the file.

7 Start Zenoss platform.

```
zenoss start
```

8 Update remote collectors, if deployed. Repeat the following steps for each remote collector.

a Log in to the Zenoss platform user interface as a user with ZenManager or Manager privileges.

b Click **ADVANCED**, and then **Collectors**.

c Display the collector's overview page.

d In the **Performance Collector Configuration** panel, select **Update Collector...** from the **Action** menu.

e In the **Update Collector** dialog, click **OK**.

9 Update remote hubs, if deployed. Repeat the following steps for each remote hub.

a Log in to the Zenoss platform user interface as a user with ZenManager or Manager privileges.

b Click **ADVANCED**, and then **Collectors**.

c Display the hub's overview page.

d In the **Hub Configuration** panel, select **Update Hub...** from the **Action** menu.

e In the **Update Hub** dialog, click **OK**.

## Removing a ZenPack

The ZenPack removal procedure requires stopping and starting Zenoss platform.

Note Removing a ZenPack can have unexpected consequences.

- Removing a ZenPack removes all objects provided by the ZenPack, as well as all objects that depend on code provided by the ZenPack. Review the details page of the ZenPack to remove, before removing it.
- Removing a ZenPack that installs a device class removes the device class, any contained device classes, and all devices in that class.

To avoid negative consequences, Zenoss recommends the following steps before removing a ZenPack.

- Delete data sources provided by the ZenPack to remove, if any.
- Back up Zenoss platform data with the `zenbackup` command.

1 Log in to the Zenoss platform master host as `zenoss`.

2 From the list of installed ZenPacks, identify the name of the ZenPack to remove.

```
zenpack --list
```

The output includes the full name of the ZenPack; a blank space; and the ZenPack location, in parentheses.

3 Stop Zenoss platform.

- *Stopping Zenoss platform without remote collector or hub hosts* on page 50
- *Stopping Zenoss platform with remote collector or hub hosts* on page 51

At the end of both procedures, you are logged in to the Zenoss platform master host as `zenoss`.

4 Start the event and catalog servers.

```
zeneventserver start; zencatalogservice start
```

5 Remove the ZenPack. Replace *Name* with the full name of the ZenPack to remove.

```
zenpack --remove=Name
```

6 If the removed ZenPack provides a daemon, and you are using `$ZENHOME/etc/daemons.txt`, delete the daemon from the file.

7 Start Zenoss platform.

```
zenoss start
```

8 Update remote collectors, if deployed. Repeat the following steps for each remote collector.

a Log in to the Zenoss platform user interface as a user with ZenManager or Manager privileges.

b Click **ADVANCED**, and then **Collectors**.

c Display the collector's overview page.

d In the **Performance Collector Configuration** panel, select **Update Collector...** from the **Action** menu.

e In the **Update Collector** dialog, click **OK**.

9 Update remote hubs, if deployed. Repeat the following steps for each remote hub.

a Log in to the Zenoss platform user interface as a user with ZenManager or Manager privileges.

b Click **ADVANCED**, and then **Collectors**.

c Display the hub's overview page.

d In the **Hub Configuration** panel, select **Update Hub...** from the **Action** menu.

e In the **Update Hub** dialog, click **OK**.

# Service Impact terms and definitions

**component node**

An Service Impact object representing a Zenoss platform component.

**contextual policy**

A policy to send different state data to different nodes in the parent level. Affects only one service model.

**custom state provider**

Customized nodes that gather state data from classes other than the standard `/Status/Ping` and `/Perf` classes.

**device node**

An Service Impact object representing a Zenoss platform device.

**global policy**

A policy to send state data to to the parent node. Affects all service models in which a node with this policy occurs.

**logical node**

An Service Impact object that captures specific Zenoss platform event states.

**service event**

A Zenoss platform event, generated by Service Impact when an availability or performance state change affects a service node.

**service node**

An Service Impact object that represents either a service model as a whole, or a service on which other service nodes rely (a subservice).

**state trigger**

A rule that examines the states of child nodes and selects the type of state data to send to the parent level in the service model.