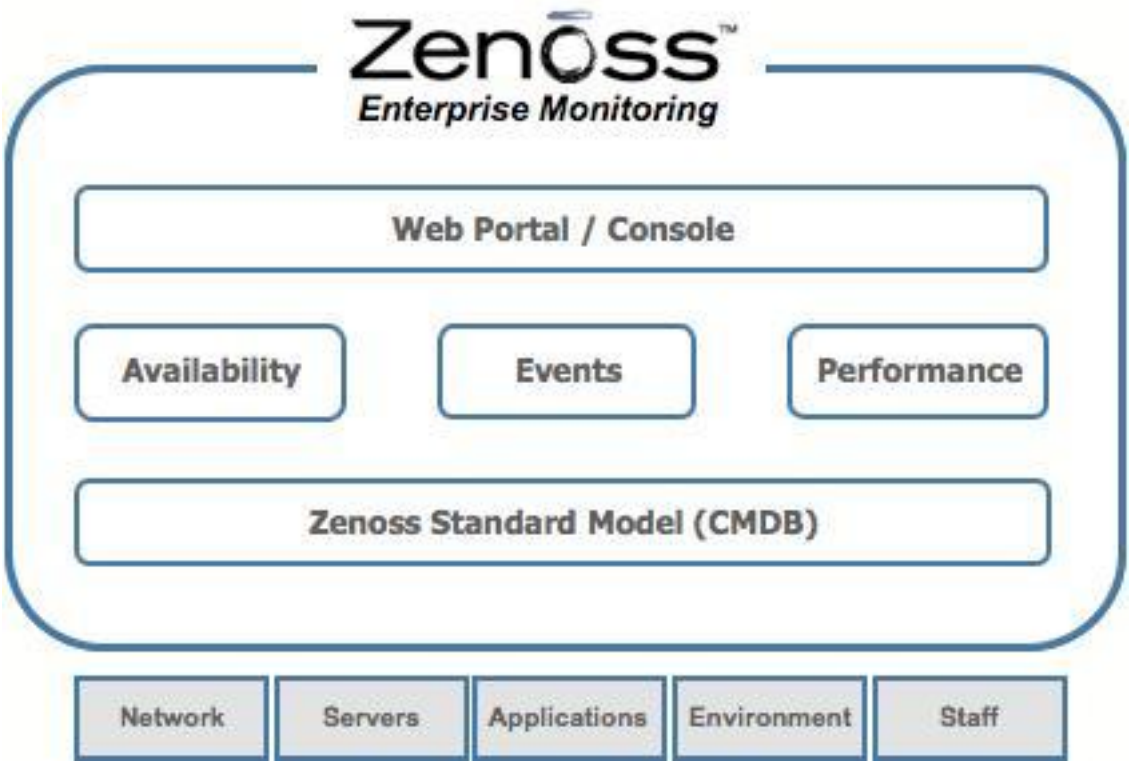
第一章. 介绍

1. Zenoss概览

Zenoss系统将各种监视和管理信息通过一个标准的web浏览器集中向用户展示，系统的各种功能都可以通过web界面进行访问而无需编写任何配置文件。从一个较高的层面来讲，Zenoss由四个主要部分组成。

图1.1. Zenoss 系统架构



1.1. Zenoss标准模型

Zenoss的核心是Zenoss标准模型，该模型详细地描述了Zenoss管理的设备，同时还描述了设备之间、Zenoss业务对象之间以及其它用户定义的重要分组之间的关系。由于该模型高度复杂，因此模型信息的来源也多种多样，其中一个最主要的来源称之为Zenoss自动发现进程，自动发现是指，Zenoss通过一个可用的传输通道来发现设备上的服务、接口等信息。通过这些发现的信息，Zenoss在系统中为设备建立一个模型。同时，用户可以通过Web界面手工输入设备相关数据的方式（或通过Zenoss的外部API）为设备建立模型。Zenoss的2.0版本增加了发现锁定功能，该功能使得自动发现的信息可以与手工录入的设备信息紧密地结合在一起，并为设备进行建模，而建立后的设备模型用于驱动Zenoss系统的所有监视元素。

1.2. Zenoss可用性监控

Zenoss的可用性测试包括针对IT基础架构的系统运行测试，通过测试可以判断系统是否在正常运行，这些测试通常在被监控的系统外部运行，测试手段包括： ping测试、进程测试和服务测试。.

1.3. Zenoss 事件管理系统

当Zenoss的监视进程检测到有失败信息或者门限值被突破后，系统就产生一个事件，该过程与目前市面上绝大多数可用的监视系统一样。Zenoss的事件管理是Zenoss系统各部分状态信息以及受其监视系统信息的一个整合。Zenoss还可接入来自IT基础设施其它部分的事件，这其中包括Syslog和SNMP Traps。Zenoss收到这些事件后，通过一套规则进行处理并最终将这些事件整合进Zenoss模型。

1.4. Zenoss 性能监视系统

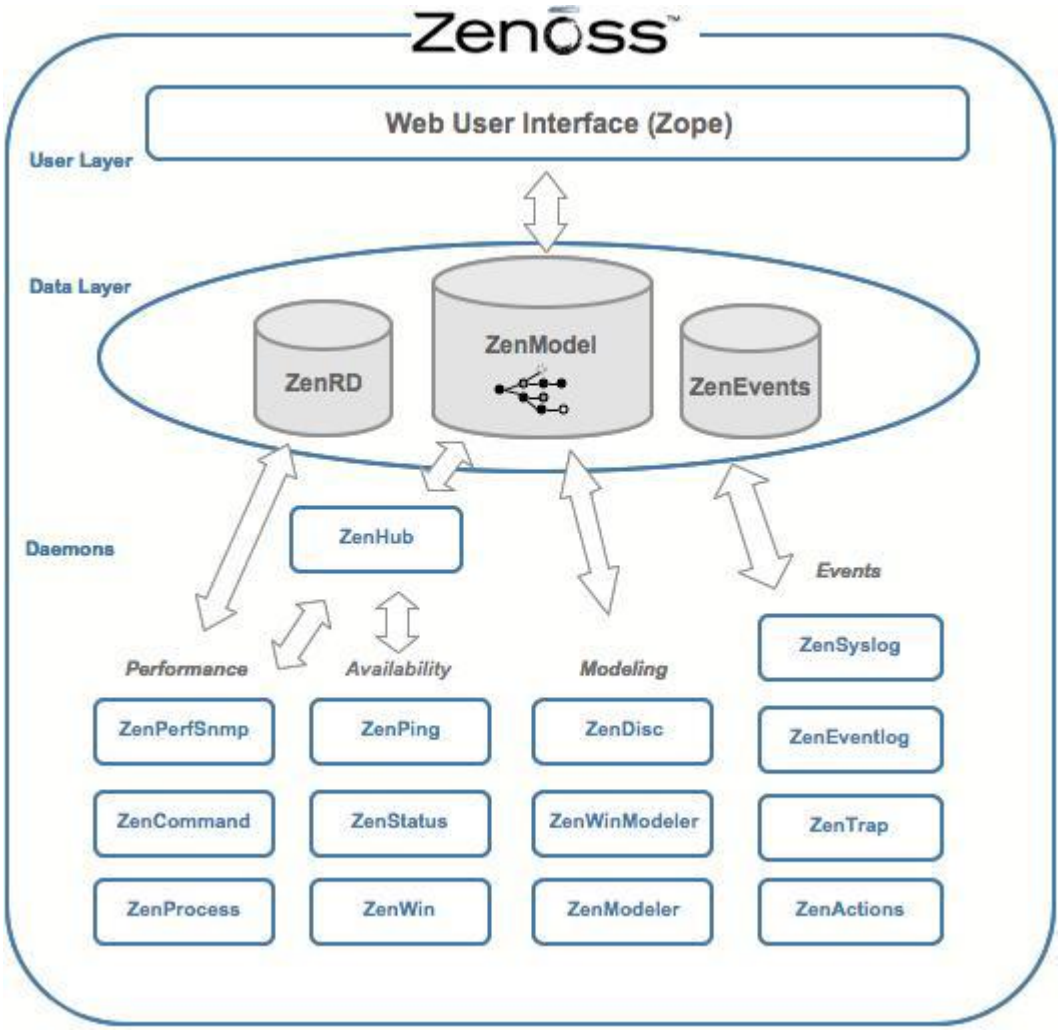
Zenoss性能监视系统的作用是，跟踪重要的IT资源信息并随时记录其变化。对系统管理员来说，随时了解磁盘可用率、CPU负载以及Web页面载入时间都相当重要。而Zenoss就可以通过SNMP、自定义脚本（ZenCommands）或XML-RPC来采集这些数据。由于性能信息被整合在Zenoss模型中，因此浏览在其它Zenoss信息时，用户也可以根据上下文获得有关设备的性能数据。

第二章. 具体结构

1. Zenoss 具体结构

下图更为详细地展示了Zenoss系统的结构。

图 2.1. Zenoss 具体结构图



图中显示，Zenoss系统分三个不同的层面：用户层、数据层以及采集与控制层。Zenoss守护进程主要在采集与控制层。

2. 用户层

用户层以门户（WebPortal）的形式出现，该层包含用户图形界面(GUI)，用户可通过GUI访问如下信息：

Dashboard（仪表板）	Events（事件）	Locations（位置）
Devices（设备）	Manufacturers（制造商）	Reports（报告）
Services（服务）	Systems（系统）	Users（用户）
Networks（网络）	Groups（分组）	Administration（管理）

用户层负责与数据层进行交互，同时将数据层的信息翻译出来并通过GUI向用户表达。

3．数据层

数据层存放了所有的系统信息，该层包含了各种Zenoss的守护进程以及用于运行系统核心的zeoctl和zopectl进程。 Zeoctl 是后台的对象数据库，该数据库保存着配置模型，zopectl用于控制Zope Web应用程序开发环境。

进程	描述
ZenrRRD	ZenRRD 采集时序数据，作用类似于RRDtool.
Zenevents	Zenevents 负责与MySQL事件数据库进行交互
Zenmodel	Zenmodel 是Zope对象数据库的统一配置模型
Zenhub	数据层与采集层之间的信息中介

4．采集与控制层

采集与控制服务层的进程负责采集数据，同时向数据层提供数据。该层进程可被分为五类：自动化建模进程、可用性监视进程、事件采集进程、性能监视进程和自动响应进程，详细信息如下所示。

4.1. 自动建模进程

进程	描述
Zendisc	Zendisc是zenmodeler的子类，该进程用于发现新的网络资源。该进程通过遍历路由表来发现网络拓扑，之后通过Ping来发现网络和寻找活动的IP和设备。
ZenwinModeler	ZenWinModeler 用于自动发现运行于windows主机上的服务。
ZenModeler	ZenModeler 是一个配置采集和配置进程。ZenModeler使用SNMP,SSH,Telnet进行高性能的自动化的建模工作，Zenmodeler 针对加载了DMD的设备进行工作。

4.2.可用性建模进程

进程	描述
Zenping	Zenping 是Zenoss的ping状态监视进程，主要对ICMP状态进行高性能异步测试。
Zenstatus	Zenstatus 进程会主动发起对远程进程的TCP连接测试。
Zenprocess	Zenprocess 通过使用SNMP主机资源mib对进程进行监控。

4.3. 事件采集进程

进程	描述
Zensyslog	Zensyslog对syslog事件进行分类恶化采集。
Zeneventlog	Zeneventlog用于采集(WMI) event log事件.
Zentrap	Zentrap 采集 SNMP Traps并将其转换为事件。

4.4. 性能监视进程

进程	描述
ZenperfSNMP	ZenperfSNMP 主要执行高性能异步SNMP性能采集。
ZenperfXMLrpc	ZenperfXMLrpc 用于XML RPC采集.
Zencommand	Zencommand 进程用于XML RPC采集，尤其是该进程的运行使得在本地主机或者远程主机（通过SSH）上运行Nagios©和Cactii插件成为可能。.

4.5. 自动响应进程

进程	描述
Zenactions	Zenactions 进程主要用于告警 (SMTP, SNPP 以及 Mainten-ance Windows).

第三章. 关键概念

1.分类（Classification）

Zenoss的层次结构主要用来对IT实体进行分类,比如设备（计算机）或事件（设备发出的状态信息）。对IT实体的正确分类有助于系统对IT实体的了解，这使得正确分类成为系统中一项极为重要的活动。

2. zProperties

Zenoss允许通过使用其结构化的组织系统来制定配置信息。zProperties是被应用到设备或者分组上的属性，这有助于对Zenoss的不同模块进行控制。用户可以在Zenoss层次结构中的任何一层设置zProperties，在较低层上设置的zProperties的值将覆写其父层zProperties的值。

3.继承与路径导航（Inheritance and Path Navigation）

Zenoss使用层次结构来组织信息，就如文件系统一样。用户可以通过使用路径来浏览这些层次结构，这种操作方式与Unix文件系统或者使用一个web URL来导航的方式如出一辙。Zenoss使用路径将用户导航至数据库中的对象，但路径并非一个实际的文件系统。

第4章. Zenoss界面与导航

1. Zenoss 仪表板

一旦用户完成了Zenoss安装并在用户界面中输入URL，用户将会看到仪表板。仪表板是在Zenoss系统中进入到设备、事件和活动视图的主要窗口。仪表板显示系统级的事件汇总，也显示至少有“错误”级别事件的设备，同时也显示基础设施事件。仪表板中有一个搜索框，用户可以在搜索框中输入主机名、IP地址或者partial name来进行搜索。用户可以通过屏幕右侧上部的“Preferences”链接来访问用户信息。“Preferences”链接下方是Zenoss最近一次更新的日期和时间。默认情况下，系统通过AJAX调用每60秒钟刷新一次数据，如果该操作失败，系统将在用户信息区域附近显示“Lost Connection to Zenoss”（至Zenoss连接失败）。

图 4.1. Zenoss仪表板

2. 左侧导航菜单

左侧的导航菜单包含三个主要部分：主要视图（MainViews）、分类浏览部分（Classes）以及管理功能（Management）部分。

隐藏左侧导航菜单用户可以使用左侧导航菜单上侧的三角形来隐藏导航菜单，之后用户就可以使用别针图标将导航菜单钉在用户想要的位置，然后导航菜单将会重新出现。

2.1. 主要视图

导航菜单的主要视图部分包含以下一些视图：用于展示主要视图的仪表板、显示所有当前事件的事件控制台（Event Console）、显示系统中所有设备的设备列表（Device List），同时还有网络拓扑图链接，用户可以通过该链接看到网络中设备的图形化展示界面。

2.2. 分类区（Classes ）

分类区包含以下链接：点击事件链接，用户可以进入到事件管理页来监视事件状态、事件、事件历史、zProperties、事件转换以及事件变化的跟踪记录；设备连接允许用户进入到设备管理页面，同时还可以看到按照重要程度进行分类的事件列表、设备的事件历史、性能配置页、设备的 zProperties 页以及设备最近的变化；点击服务链接后，用户可以看到服务分类、基于服务的管理员命令和 zProperties，同时用户还可以跟踪服务监视的任何变更；进程链接允许用户创建一个新的进程分组，这里我们成为子组，同时还可以向系统中添加需要监视的进程，用户还可以设定收集进程监视信息的顺序。同时在进程分类中，用户还可以运行针对进程的命令或者访问进程的 zProperties。产品链接向用户显示 Zenoss 数据库中的所有设备制造商列表。

2.3. Browse By

在 "Browse by"功能菜单区域中，用户可以选择按照几种不同的逻辑分组来查看 Zenoss 的相关数据。比如"Systems"视图，用户可以按照系统来查看与该系统相关的设备或者网络状况，用户可以查看某系统的性能数据，同时还可以查看与该系统相关的事件以及事件历史，用户还可以使用系统视图中的管理页标签来定义命令、创建维护窗口或者定义一个托管对象；"Locations"视图使得用户能够基于物理位置来查看与该物理位置相关的 Zenoss 数据。"Group"(分组)视图允许用户按照某种分组来浏览数据，当然这种分组是用户定义的分组。按照位置(Location)浏览是指，浏览物理位置在一起的设备的相关数据，在这里，所有的设备可以在一个位置上，但不同的设备可以有不同的子位置。系统中的任何一个设备都可以属于多个不同的分组(group)，但同时只能有一个位置。但一个设备同时安装在一个位置和该位置的一个子位置是被允许的，比如，一台服务器可以安放在马里兰州，同时又可以安放在安纳波利斯（马里兰州首府）。用户还可以通过 Google-map 来访问设备的物理位置以及设备的相关状态；按照网络浏览是指，按照 IP 地址对设备进行分组，用户可以查看某一网络下所有的设备及其子网，同时用户还可访问基于 IP 的 zProperties。Report 选项允许用户查看和定义报告。

2.3.1. 管理

Add Device（添加设备）

Mibs - 添加新的分类以及新的MIB库（Add New organizer, Add new MIBS）

Monitors - 状态监视器和性能监视器

Settings - Settings, commands users ZenPacks Menus Daemons Versions

Event Manager - 修改数据库连接信息，事件缓存、事件历史

2.4. 隐藏左侧导航菜单

用户可以通过点击左侧导航菜单上面的三角形来隐藏导航菜单，同时还可以使用图钉图

标将导航菜单钉在界面上的某个位置，或者取消图钉功能。

3. 目录路径

目录路径是设备显示的路径，该路径显示数据在用户界面中出现的位置信息。

4.设备/IP搜索框

用户可以在该搜索框中输入设备名称或者IP地址，并以此为手段来搜索具体设备。.

5.用户信息区

当前用户的ID现实在屏幕顶部的右侧，紧邻着Preferences链接。同时还可以通过点击导航区域左侧的Settings链接来访问用户设置，或者点击LogOut链接退出当前登录。

6. 自定义Zenoss仪表板布局

用户可以通过选择不同的页面插件（Portlets）以及将这些页面插件摆放在不同的位置来自定义Zenoss仪表板的布局。要设定仪表板页面布局方式的话，用户可以在仪表板右上侧选择配置布局（Configure layout.），之后系统将列出几种不同的页面布局风格。

图 4.2. 列布局对话框

给仪表板选择列布局，一旦选定布局之后，用户就可以对页面插件进行随意拖拽，直至用户满意度位置。用户可以在仪表板右上侧上选择添加页面插件链接（Add Portlet.）来控制在仪表板上显示哪些页面插件，下图是添加页面插件对话框：

图 4.3. 添加页面插件对话框

可以添加到页面插件包括：

- 设备事件（Device issues）
- 顶级分类（Top level Organizers）
- 观察列表（Watch List）
- Google地图（Google Maps）
- Zenoss事件（Zenoss Issues）
- 生产状态（Production States）

6.1. Devices Issues 页面插件

该插件显示设备的名称、用户已确认的设备状态以及该设备上严重和错误级别的事件。点击设备名称，可以查看设备的事件日志 Event Log，事件日志以已确认/总数进行分类。

图 4.4. Device Issues 页面插件

6.1.1. 配置 Device Issues 页面插件

如果想要配置在Device Issues 页面插件上显示的信息的话，可以点击页面插件窗口右上侧的*号，之后系统将弹出下拉对话框，用户可以在该对话框中更改页面插件显示的信息，同时还可以修改页面插件的抬头标题以及刷新率，同时也可以在该窗口中控制该插件在页面上不显示。

图 4.5. Device Issues 页面插件配置

6.2. Top Level Organizers顶级分类

顶级分类页面插件显示Zenoss层次结构中的顶级分类，如下图所示：

图 4.6. 顶级分类页面插件

6.2.1. 配置顶级分类页面插件

如要配置在顶级分类页面插件中显示的信息，用户可以点击顶级分类页面插件右上侧的*号，在系统弹出的对话框中，用户可以更改顶级分类页面插件的显示信息，可以更改该页面插件的抬头标题、刷新率，还可以选择想要显示的根分类，还可以控制该页面插件在页面上不显示。

图 4.7. 顶级分类配置

6.3. Watch List 页面插件

该插件显示一个观察设备列表，用户可以定义需要观察到设备列表以及从被观察设备采集数据的频率。

图 4.8. Watch List 页面插件

6.3.1. 配置Watch List 页面插件

要配置 Watch List 页面插件，用户可以点击该页面插件右上侧的*号，在系统弹出的对话窗口中，用户可以修改该页面插件显示的信息、该页面插件的抬头标题以及该页面插件的刷新率，同时还可以控制该页面插件在页面上不显示

图 4.9. Watch List 插件配置

6.4. Google Maps 页面插件

Google Maps页面插件使得用户能够看到设备以及设备连接的位置图，如下图所示：

图 4.10. Google Maps 页面插件

6.4.1.配置 Google Maps 页面插件

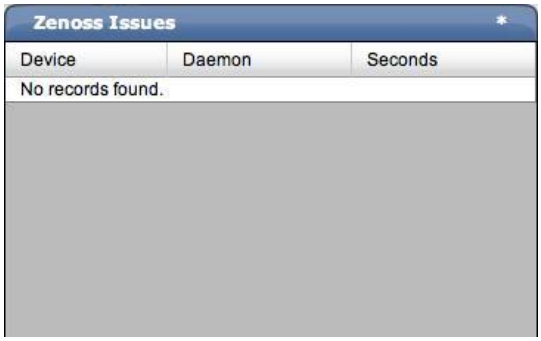
要配置 Google Maps 页面插件中显示的信息，用户可以点击该页面插件右上侧的*号，在系统的弹出对话框中，用户可以修改该页面插件显示的信息、抬头标题以及该页面插件的刷新率，同时还可以控制该页面插件在页面上不显示。

图 4.11. 配置Google Maps 页面插件

6.5. Zenoss Issues 页面插件

Zenoss 基础设施事件主要显示Zenoss进程出现的问题，比如Zenoss进程的心跳进程如果出现问题，将在这里显示一个事件，如果在这里显示有事件，说明与该事件相关的Zenoss进程停止了运行。

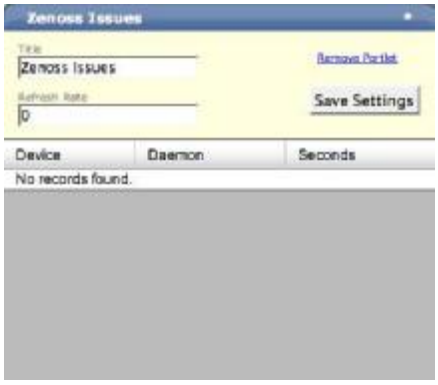
图 4.12. Zenoss Issues 页面插件



6.5.1. 配置 Zenoss Issues 页面插件

要配置 Zenoss Issues 页面插件中显示的信息，用户可以点击该页面插件右上侧的*号，在系统的弹出对话框中，用户可以修改该页面插件显示的信息、抬头标题以及该页面插件的刷新率，同时还可以控制该页面插件在页面上不显示。

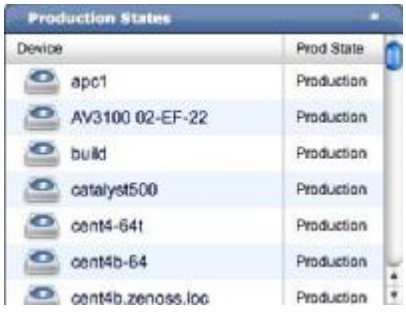
图 4.13. 配置 Zenoss Issue 页面插件



6.6. Production States 页面插件

Production States 显示所有设备的生产状态。

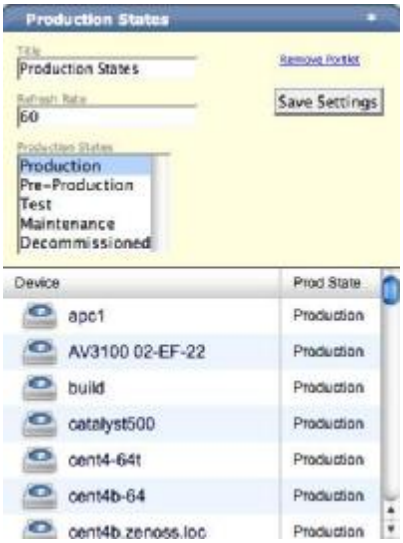
图 4.14. Production State Portlet



6.6.1. 配置Production States页面插件

要配置Production State页面插件中显示的信息，用户可以点击该页面插件右上侧的*号，在系统的弹出对话框中，用户可以修改该页面插件显示的信息、抬头标题以及该页面插件的刷新率，同时还可以控制该页面插件在页面上不显示。

图 4.15. 配置 Production State页面插件



7. Zenoss 网络拓扑图

网络拓扑图通过Flash的方式向用户展示用户网络的第三层拓扑结构。用户可以在图中双击途中一个设备或者网络的图标来查看详细信息，

如果要装载/重新装载特定某节点的数据，用户可以双击该节点，选择跳数，点击刷新按钮。同时用户还可以通过设备类别来过滤想要显示的设备：比如，用户可以在设备分类过滤条件中选择/Server/Linux，之后点击刷新按钮，这样网络拓扑图中将仅显示Linux设备。

用户可以用顶部右侧的Number of Hops滑动条来调整所选设备的可见跳数，如果想要查看所选网络或者设备的更详细的信息，用户可以点击顶部左上角的状态页链接。

图 4.16. Zenoss 网络拓扑图

8.菜单化元素（"Menu-ized" Elements）

Zenoss 有两种不同类型的菜单。

8.1.页菜单（Page Menus）

页菜单扩展了页面顶部的 Tab 页，页菜单通常会影响该页显示对象的整体，该对象可能是一台设备、一个时间或者是这些元素的一个分组。在下图中您将会看到页菜单的展开效果。

图 4.17. 页菜单

8.2. 表菜单（Table Menus）

如下图所示，用户可以通过点击表头附近的三角形来使用表菜单，下图中展示的是子设备表菜单。

图 4.18. 子设备表菜单

第5章. 用户管理

1. 关于Zenoss用户帐号

Zenoss系统中的每个用户都有唯一的用户ID，这些用户ID通过权限以及告警规则进行分组，并以此成为zenoss系统安全的基础。要在zenoss系统中创建或者管理用户帐号，就必须以Zenoss Admin帐号登录。通过帐号管理，可以对用户进行事件视图和告警规则的定制，这些内容将在本手册的其它章节进行讨论。

2. 创建新用户

新建帐号：

1. 从Zenoss仪表板左侧的菜单中选择Settings(设置)。

2. 点击Users Tab（用户页），系统将显示用户管理界面。

图 5.1. 用户管理页

3. 点击用户文件文件夹中的表菜单，将显示用户管理选项，包括添加用户、删除用户以及添加至Zenpacks。.

4. 选择“添加用户”

系统将显示“添加用户”对话框，如下图所示：

图 5.2. 添加用户对话框

5. 输入用户名.

6. 输入该用户帐号的一个电子邮件地址。该电子邮件帐号将是未来接收系统告警的邮件帐号。

7. 点击OK按钮,新建的用户帐号将出现在用户列表中。

至此，一个新的用户帐号创建完毕。管理员还可以继续编辑用户帐号以设置用户帐号口令以及其它更详细的信息，具体操作详见本用户手册的编辑用户帐号部分。

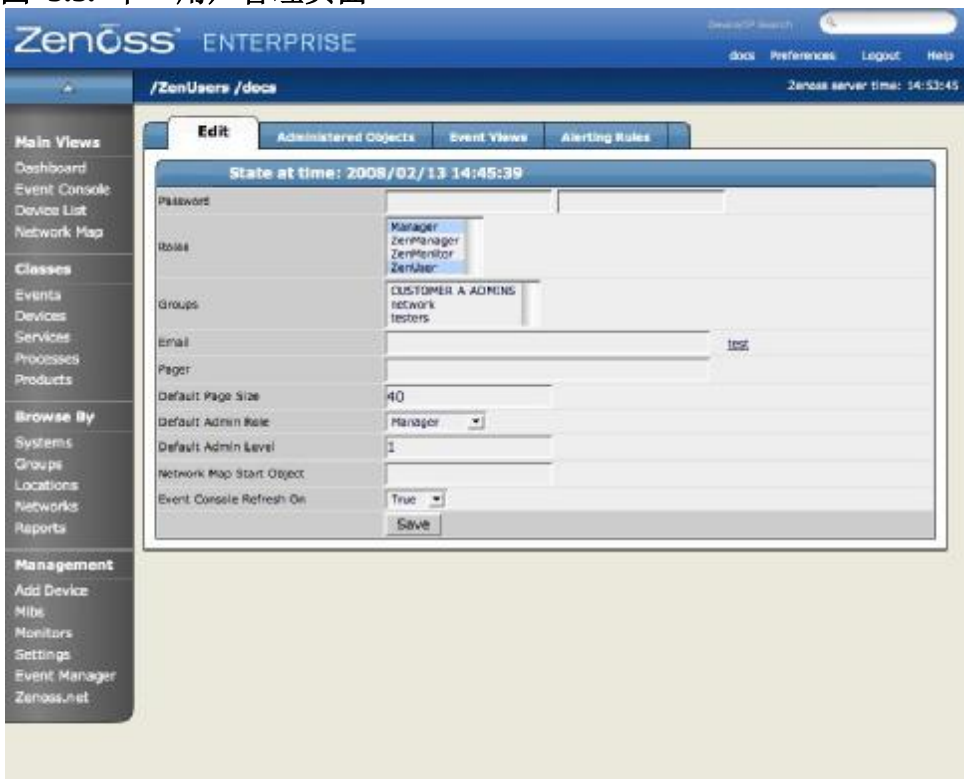
3. 编辑用户帐户

1. 从Zenoss仪表板左侧的菜单中选择Settings(设置)。

2. 点击Users Tab（用户页），系统将显示用户管理界面。

3. 点击需要编辑的帐号的用户名，系统将显示单一用户管理界面，如下图所示。

图 5.3. 单一用户管理页面



4. 修改用户详细信息，点击Save(保存)按钮后保存退出。.

3.1. 设定用户密码

从单一用户管理界面中，可以为用户指定口令。在第一个文本框中输入用户口令，在第二个文本框中再次输入口令进行确认。

3.2. 编辑用户联系信息

在单一用户管理界面中，管理员可以重新输入或者编辑用户的电子邮件地址或者寻呼机号码。

3.3. 向用户分派角色和权限

在单一用户管理界面中，管理员可以为用户指定具体的角色。如果用户被设定成manager（经理）角色，那么用户就可以更改自己或者其它任何帐号（包括经理角色帐号）的设置。

表5.1.

角色	权限
Manager	经理角色拥有所有的权利，该角色可以在系统中查看并变编辑对象。经理角色可以看到左侧导航菜单中的管理区，因此可以添加设备、管理用户、查看事件以及事件历史。
ZenUser	ZenUser角色可以查阅系统中的所有信息，ZenUser角色的用户仅可以编辑自己的帐号信息（但无法更改自己的角色）。ZenUser角色的用户无法向系统中添加设备，而且看不到左侧导航菜单中管理部分的菜单。

3.4. 设定默认用户级别

默认用户级别为一个托管对象设定默认的用户管理级别。该默认级别能够进行修改，但用户应该选择那些通过最多被添加的角色来设定成默认用户级别，可供选择的选项有：

- Administrator
- Engineer
- Analyst
- Tester

这些用户管理级别目前仅仅是组织用户的一种手段，在Zenoss未来的版本中，用户管理级别将会有更丰富的属性设置功能，但在目前仅仅用于对用户进行分类。

3.5.设定仪表板超时和刷新时间

用户可以在系统中单独为每个用户进行仪表板的设定，比如仪表板的超时时间和刷新时间等。在仪表板的超时和刷新时间文本框中，输入超时的秒数，之后超时信息将会显示在仪表板中。

3.6. 指定默认的仪表板分类

默认的仪表板分类是指，当某个特定用户登录时，显示在仪表板设备小结（Device Summary）上的仪表板分类，用户可以自定义在仪表板设备小结上的分类，比如：设备、系统、分组或者位置。

3.7. 创建对象关联

用户可以将Zenoss系统中的任何对象与一个特定用户帐号进行关联(用于监视目的)，以下是创建这种关联的步骤：

1. 在单一用户管理页中，点击Administered Objects（托管对象）页，之后系统显示托管对象。

图 5.4. 托管对象初始视图

2. 从托管设备表菜单中选择一个对象类型，可供选择的类型有设备、系统、分组、位置。之后系统将弹出一个对话框，用户可以在该对话框中指定想要管理的组件。

3.点击 OK 按钮，设备将出现在该用户的托管设备列表中。

图 5.5. 托管对象 – 添加对象

4. 至此，系统用户就可以改变与该对象关联的用户的角色了。.

默认的角色在单一用户管理页中进行设定。

用户可以直接从设备入手将该设备设定为托管对象。比如，用户可以首先找到希望托管的设备，打开该设备的页菜单，选择“More”（更多）选项，之后选择administration（管理）选项。在管理员的表菜单中，选择添加管理员，同时选择一个用户帐号并将其添加为该设备的管理员，之后该设备将进入该用户的托管设备列表，而在设备的管理员列表中将出现用户的名字。

4. 用户组

在系统中，用户可以创建Zenoss用户组，并以此来聚集业务规则并将这些业务规则应用至一组用户，以下是创建用户组的步骤：

1. 从左侧的导航菜单中选择Settings（设置）。

2. 点击用户页。

图 5.6. 显示用户组的用户页

3. 从用户组的表菜单中选择添加新组，系统弹出添加新组对话框，如下图所示：

图 5.7.添加新组对话框

4. 在组名称文本框中输入新的用户组名称。

5. 点击 OK 按钮后，新用户组名称将出现在用户组列表中。

6. 点击刚创建的用户组名称，可以看到Edit Tab（编辑页）。

7. 在用户组的表菜单中，选择Add User（添加用户），系统将显示添加用户至组对话框。

图 5.8. 添加用户组对话框

8. 在下拉列表中，选择要加入某个用户组的用户。

9. 点击 OK 按钮后，选择的用户将出现在该组的用户列表中。同时，还可以为该用户组指定托管对象和告警规则，告警规则将对该组中的所有用户适用，同时用户原有的告警规则和托管对象还将保留。

第6章. 邮件与寻呼设定

1. 关于邮件与寻呼设定

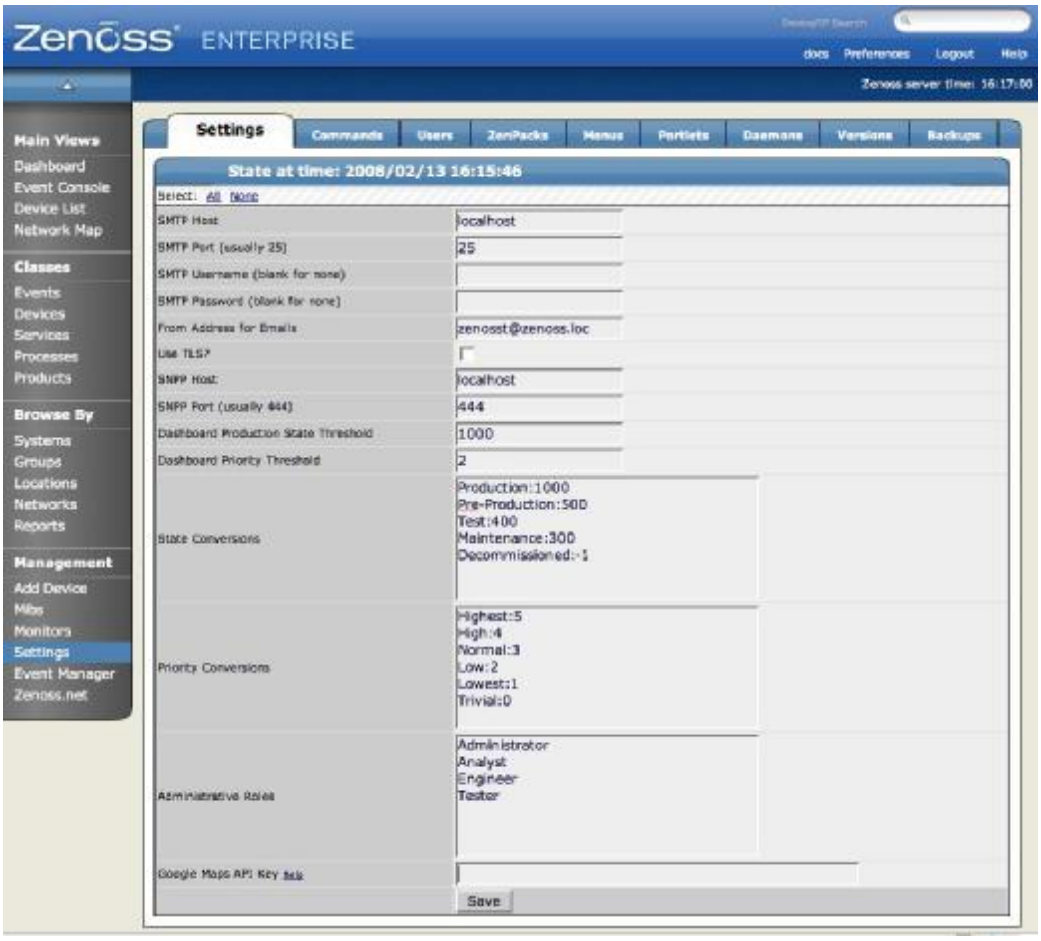
Zenoss警告可以通过email (SMTP) 或寻呼 (SNPP)发出。许多操作系统都内建有SMTP服务器（比如Sendmail或者Postfix）。如果用户的操作系统中没有邮件服务器，那么就必须安装一个邮件服务器，或者在Zenoss的设定中指定一个SMTP服务器。许多寻呼机都可以通过邮件接收到信息，但Zenoss还是提供了通过SNPP发送寻呼的选项。

2. 设定 SMTP与SNPP信息

在设置Zenoss的SMTP和SNPP之前，必须以一个Zenoss的管理员（manager account）登录，之后在左侧的导航菜单中点击设置（settings）。

1. 以管理员身份登录后，在左侧的导航菜单中点击设置(settings)，之后系统将显示设置页。

图 6.1. Zenoss 设置-设置标签页



2. 在必要的情况下修改如下SMTP配置信息:

表6.1.

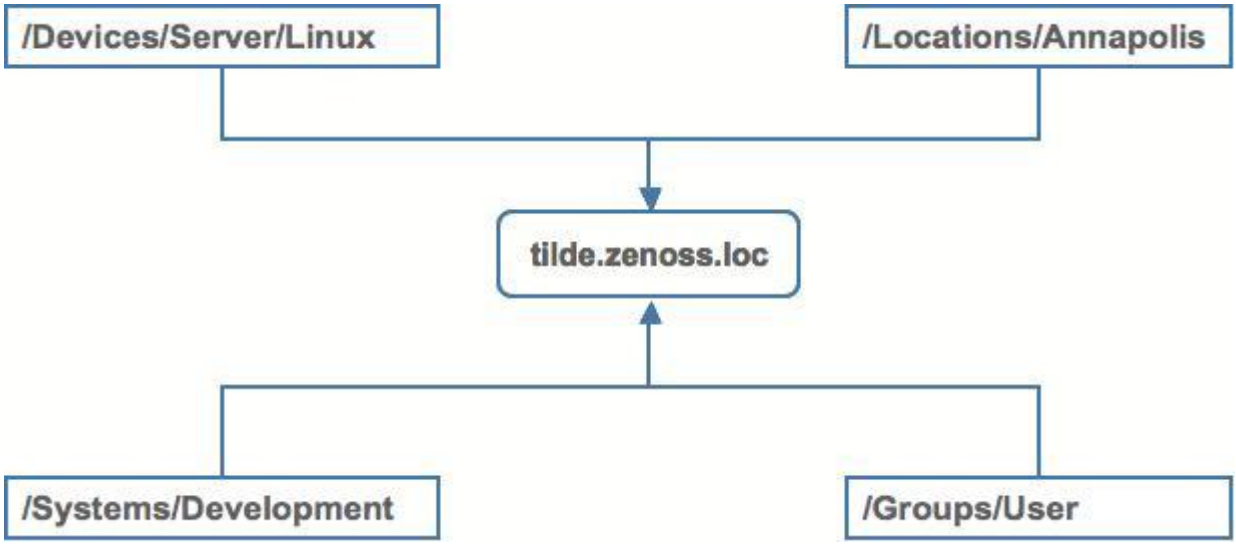
设置域	说明
SMTP Host	SMTP主机
SMTP Port	SMTP端口，通常为25
SMTP Username	如果没有则留空
SMTP Password	如果没有则留空
From Address for Emails	如果想让Zenoss告警以指定的邮件帐号发出，则在该项中设定邮件地址。
Use TLS?	如果邮件告警需要传输层安全保证，则选中该选项。
SNPP Host	发送寻呼的主机
SNPP Port	寻呼服务端口，通常为444

第七章.分类与路径导航

1. 关于分类与路径导航

Zenoss允许对系统中的任何对象（设备、子系统、zProperties、模板等）进行分组 ，分组可以以用户希望的任何方式进行创建。同时，系统中已经定义了设备分类，而设备分类本身就是一种分组，这样以来，每个设备都可以属于多个分组，比如系统、位置、设备分类。在下面这个例子中，您可以看到tilde.zenoss.loc这个设备属于5个不同的分类。每个分类的zProperties以及监视设置对该设备同时生效。

图 7.1. 设备分组



Zenoss使用分层分类的方法来对系统中的设备进行分类和组织。

• Class – 系统中最重要的分类，是所有模板和zproperties 进行继承的基础。

• Systems

• Groups

• Locations

2. 分类

Zenoss中最重要的设备组织形式就是分类（Class）。系统中有设备分类、事件分类、服务分类、产品分类。而模板和zProperties都是从分类继承而来。每次继承后，这些属性都可以在继承后的层次中进行覆写（overwritten）。分类层次结构包含了所有已定义的和标准的分类及其子类。

下面的这个例子中使用了设备分类及其子类，同样的概念也适用于事件分类、服务分类和产品分类。当用户向Zenoss系统中添加一个新的设备时，至少要提供一个设备分类信息（在提供了网络名和IP地址之后）。用户可以在设备分类层次结构中的任何一层对模板和zProperties进行设定。

如果要查看设备分类中的所有设备：
1. 从左侧的导航菜单中选择设备（Device）。系统将显示设备标签页，请注意标签页上面的导航路径已经变为'/Devices',用户可凭借此信息了解目前自己在设备费分类层次结构中所处的位置。

图 7.2.设备分类标签页

设备分类标签页中显示了该层设备的“事件彩虹”及其下层设备分类的事件总结列表。

2.1. 在顶层分类中设定zProperties

如果想要在设备分类层次上设定zProperties：：

1. 首先导航至设备的分类标签页，然后点击zProperties标签页，如下图所示：

图 7.3. 设备分类的 zProperties 标签页

2. 用户可以在该标签页中定义所有通常的设备zProperties，并且这些设置将在本类的所有设备上生效，除非其子类的zProperties设置覆写了本类的zProperties设置。

2.2. 在分类层次上定义并使用模板

如果想要在设备分类层次上定义模板：：

1.导航至需要定义模板的设备分类，点击模板标签页，出现模板设置界面，如下图所示：

图 7.4. 设备分类模板标签页

2. 在该界面中，用户可以定义任何需要的模板，这些模板将对该设备分类中的所有设备生效，除非其子类模板设置对其进行了覆写。

2.3.创建新的设备分类

如果要新建设备分类：：

首先导航至设备分类标签页，从SubClasses表菜单中，选择添加新分类(Add New Organizer)选项,然后输入新分类的名称。如果想要向新建的设备分类中添加设备,可以先导航至设备列表,从表菜单中选择设备分类，然后在该设备分类中添加设备。

3. 系统

是指按照系统对设备或者网络进行分组。

3.1. 添加、移动与嵌套系统

要创建新的系统或者子系统：

1. 从左侧的导航菜单中选择系统（Systems），之后将显示子系统状态界面，如下图所示：

图 7.5. 子系统状态菜单

2. 打开子系统表菜单，选择添加新分类（Add New Organizer）选项，之后系统将弹出添加新分类对话框，如下图所示：

图7.6.添加新分类对话框

3. 在ID域中输入新分类的名称。

4. 点击 OK按钮，新建的子系统将出现在列表中。.

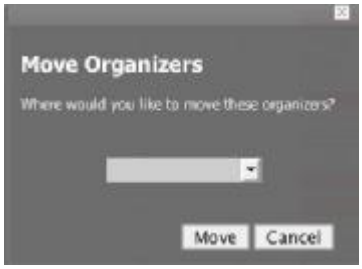
3.1.1. 移动子系统

如果要将子系统移至其它分组或者子组：

1. 在系统列表中选择想要移动的系统（在前面的checkbox上打勾），然后打开子系统表参带，将显示子系统选项。

2. 选择移动（Move）选项，系统将弹出移动分类（Move Organizer）对话框：

图 7.7. 移动分类对话框



3. 在下拉菜单中，选择移动的目标分组。

4. 点击移动（Move）按钮，该分组将移动至所选分组。

5. 位置

使用位置（Location）分组与使用系统(systems)和分组（gourps）进行逻辑分组的方法是类似的，使用位置进行分组的意思是，按照设备的物理位置进行分组。这个物理位置可以是一个城市，一个州，也可以是特定的某个机架或者槽位。使用位置进行分组是完全按照客户需要进行的。需要指出的是，位置信息虽然不出现在仪表盘上，但按照位置信息进行分组对监视网络和设备也是同样重要的。

5.1. Google Maps –网络健康的地理视图

5.1.1. 概览

Zenoss 可以通过设定一个Google Maps认为合理的位置属性信息来使用Google Maps，用户输入的位置信息在Google Maps上将表示为一个点。点的颜色代表了当前位置上有多高重要等级的事件发生，而且在Google Maps上还可用连线来显示节点之间的网络连接，同时还可以不同的颜色显示网络连接的状态。

如果要访问一个网络在Google Maps上的位置，可以从左侧导航菜单中点击位置(Locations)，然后选择该网络所在的位置，之后点击Map标签页，系统将显示地图。

5.1.2. API Key

用户在使用系统的Google Maps功能之前，必须在Google Maps上注册一个API key，免费的Google MapsAPI keys必须通过Google Maps获取，而Google Maps的企业客户无需这样做。如要获得一个免费的Google Maps API key:

1. 在浏览器地址栏中输入http://www.google.com/apis/maps/signup.html

2. 然后输入访问用户Zenoss系统的URL链接，包括端口，比如"http://local-host:8080".

如果用户未通过上面的链接来访问自己的Zenoss系统（比如，通过IP地址而不是主机名来访问）将无法使用系统的Google Maps 特征。

3. 同意条款并点击OK按钮后，用户可以获得一个API key，将该API key拷贝纸粘贴板。

5.1.3. 给位置设定一个地址

1. 从左侧的导航菜单中选择位置（Locations）。

2. 在Summary 表中选择编辑（Edit），之后系统将弹出一个编辑对话框。

3. 在新的地址栏中，输入一个可被Google Maps完全解析的地址（如果您无法确定，请参考 http://maps.google.com and see if it maps)

4. 点击保存按钮，这样，用户就可以为“位置”创建一个地址，该地址将在Google Maps上显示。同时，用户必须在该位置上添加设备，这样在Google Maps上才能显示出来那个位置的点。

5.1.4. 网络连接

如果有位于同一个网络内的两个设备位于地图上的不同位置，那么系统将自动在这两个设备之间画一条线以代表两台设备之间的网络连接。如果是同样的上面的两个位置之间有多个不同的网络连接，那么系统将只能画一条线，线的颜色将代表这个网络连接的最严重级别事件的颜色，该网络连接的状态是由以下因素决定的：

- 两台设备之间任何一端的ping down事件；或者
- 网络连接任何一端设备上的端口上产生了事件。

5.1.4.1. zDrawMapLinks 属性

我们知道，计算网络之间的联络是一个非常复杂的过程，如果网络中存在大量的设备，且这些设备已经被指定了位置，在地图上绘制这些链路将会是一个非常耗时的过程。为了节省时间，您可以告诉Zenoss以使其不绘制某些特定网络的链路（比如说一个包含了很多设备的局域网）。

1. 导航至 Network并点击"zProperties" 标签页

2. 将 "zDrawMapLinks" 属性设置成"False."

3. 点击保存（"Save."）

与所有的zProperties一样，该设置将被与该网络有集成关系的所有子网继承。如果您的网络链路不需要在地图上进行绘制，那么将zDrawMapLinks属性的值设置成 False将会是个好主意，我们建议，在您知道网络生成树的情况下，将zDrawMapLinks的属性设置为True.。

5.1.5. Google Maps 例子

下面的这个例子将向您展示如何为设备创建并显示google map ，以及发送一个测试事件以检查网络连接是如何被系统中的变化影响的。

1. 导航至/Network, 点击 "zProperties" 并将 "zDrawMapLinks" 设置为 False.保存。.

2. 导航至 /Locations 并创建两个子位置（sub-Locations）,这两个子位置称为"New York" 和 "Los Angeles".

3. 到两个位置的状态标签页，将每个位置的地址（Address）属性分别设置为"New York, NY" 和"Los Angeles, CA" 。

4. 将Zenoss系统中的某个设备的位置设置为/Locations/New York. ，将另一个设备的位置设置为/Locations/Los Angeles。

5. 导航至 /Locations 并点击"Map" 标签页，这时地图上将以两个点的方式显示New York 和 Los Angeles，但此时在两点之间并没有连线。

6. 现在，导航至与两个设备连接的网络，点击"zProperties"标签页并将"zDrawMapLinks" 属性设置为True，然后保存。.

7. 重新回到位置(Location)的 "Map"标签页，可以看到New York与Los Angeles之间已经被连上了一条绿线。.

8. 向/Locations/New York 的设备发送一个级别为"Critical."的事件 （参见第十章第四节），不指定事件的组件。现在刷新/Locations "Map" 标签页，可以看到New York前面那个小圆点的颜色已经变成了红色，而New York与Los Angeles之间的连线仍旧是绿色。

9. 现在，导航到 New York设备的 "OS"标签页， 确定组件的ID，该组件与网络连接，而且Los Angeles 的设备也连接在该网络上。然后发送一个测试事件，这次要在事件中指定组件。之后刷新/Locations的"Map" 标签页，我们就可以看到，两地之间的链接变成了红色。

5.2. 添加、移动与嵌套位置

可以通过如下步骤创建新的位置或者子位置：

1. 从左侧的导航菜单上选择位置（Locations），之后可以看到子位置（Sub-locations）状态标签页。.

2. 打开子位置表菜单，显示子位置选项.

3. 选择添加功能，之后系统弹出添加对话框。

4. 在ID域中输入子位置的名称。.

5. 点击 OK按钮后，新的子位置将被添加至系统并出现在列表中。.

5.2.1. 移动子位置

如果要将子位置移动到另外一个位置或者子位置，必须采取以下步骤：

1.从位置列表中选中一个希望移动的位置，然后从子位置表菜单中选择移动分类（Move Organizer）选项，系统将弹出移动分类对话框。

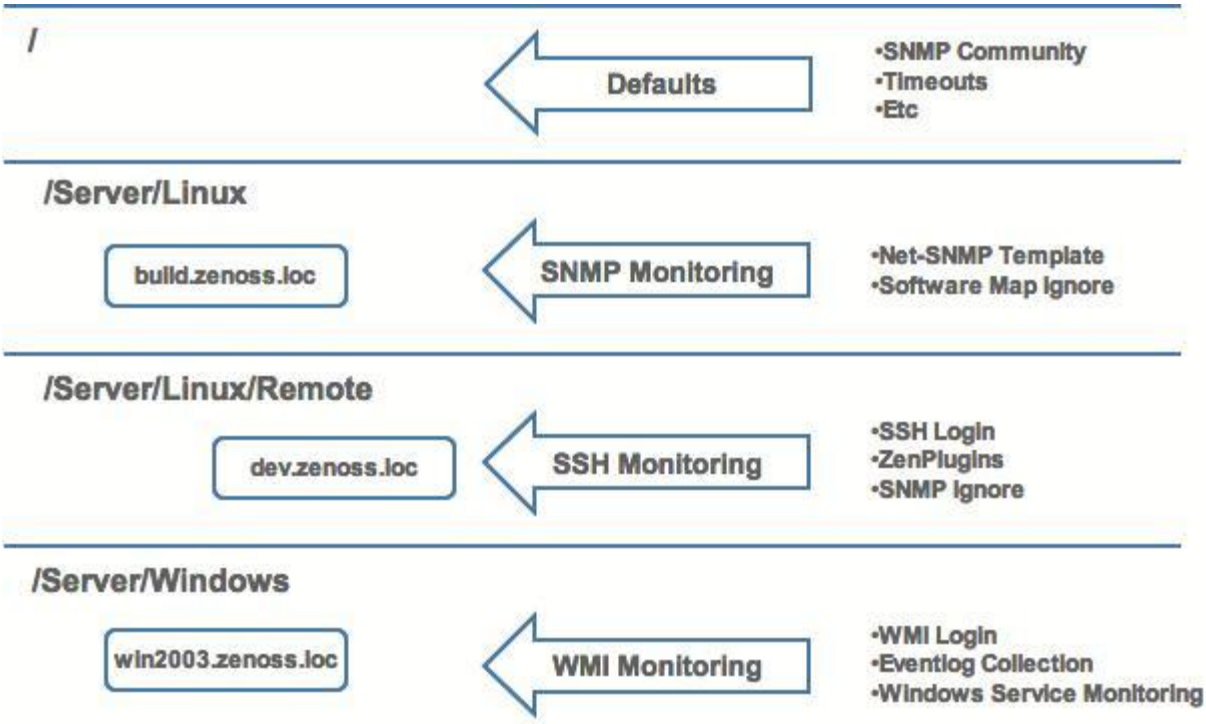
2.从弹出菜单中，选择你希望移动到的位置。

3. 点击移动（Move）按钮，就可将位置移动至希望的位置，之后系统将显示目的位置的属性页。

6.继承(Inheritance)

继承是指，属性在设备层次结构中的不同位置应用到了设备上。下面这个例子显示了zProperties 在整个设备树上从哪里以及如何进行设置。

图 7.10. Zenoss设备分类树及其继承体系



在这个例子中，我们可以看到默认属性可以在分类树层次结构的最高层（/）进行设定，随着树层次结构的深入，任何子层中都可以对根层次的 zProperties属性进行覆写，接下来的两条线，我们可以看到设备树如何进一步定义Linux Servers的属性，如果您希望建立并使用SNMP来监视所有的Linux设备，您可以在/Server/Linux 层改变属性设置。现在如果您想要改变如何采

集远程Linux Servers信息的方式，您可以在/Server/Linux 分组中创建/Server/Linux/Remote分组，并对该分组中要使用SSH进行远程监视的的主机进行设置。同时在/Server分组中，您还可以为Windows主机创建一个子组，并对其进行设置以使这些主机可以通过WMI进行监视。在设备分类树体系中，永远是下层属性的优先级高于上层属性的优先级。用户还可以将设备同时放在多个分组下面，这与文件系统的目录树非常相似。

7. Multiple Mix-In 继承与性能（RRD）模板绑定

性能配置存储在称之为RRD模板的对象中，我们通常就称其模板。模板中也包含了其它一些对象，这些对象定义了从哪里以及如何获取性能数据、门限以及数据的图表。模板可以在设备层次体系中的任何一个位置进行定义，也可以在任何一个单独的设备上进行定义。

决定将哪些模板应用到什么对象上，即称之为绑定。进行绑定有两个步骤：首先要知道正确的模板的名字，其次要找到模板的位置。用于组件的模板的名字有文件系统（FileSystems）、接口（Interfaces）等等。举例来说，文件系统使用的模板的名称就是文件系统（FileSystems）。至于设备，则比较复杂，因为用户可以修改将要使用的模板的名称，而且还可以指定不止一个名称。模板名称列表存储在一个叫zDeviceTemplates的zProperty中，这个zProperty可以直接在任何一个设备或者设备分类的zProperties页面中进行编辑。同样的，这个zProperty还可以通过绑定模板菜单项以及任何设备或者设备分类的模板页对话框进行编辑。

模板绑定的第二个步骤是，根据给定的名字找到正确的模板。至于zProperties, 由于模板可以直接被定义到一个设备上，或是可以从高层设备分类中继承，因此在搜索设备分类层次体系的过程中，第一个找到的并具有正确名称的模板会被首先使用。任何其它拥有相似名称的，且位于当前模板所在层次之上的模板会被忽略。

7.1.模板绑定例1

首先，您可以在/Devices/Server/Linux/下新加一个设备并称之为Example1Server，此时您尚未编辑该设备的zDeviceTemplates，所以zDeviceTemplates将从root Device Class继承，在本例中，将在"/Devices"中继承。Zenoss将检查Example1Server上是否有叫做Device的模板，如果没有，zenoss再检查/Devices/Server/Linux，该分类下有一个叫做Device的模板，因此该模板将应用到Example1Server。同时在/Devices下也有一个叫做Device的模板，但由于/Devices/Server/Linux下的Device模板对其进行了覆写，因此/Devices下的那个叫做Device的模板不会应用到Example1Server上。

7.2.模板绑定例2

在本例中，您希望对运行有特定web应用的主机执行特定的监视任务，但这些主机分属不同的设备分类。此时，您可以在/Devices下新建一个叫做WebApplication的模板，该模板拥有适当的数据源、门限以及性能图表。之后，您就可以将"WebApplication"挂接到设备分类或者任何一个运行web应用的设备的zDeviceTemplates zProperty上。如果您喜欢，你可以在模板页中使用绑定模板菜单项，同时也可以直接编辑zDeviceTemplates。

第8章. zProperties

1. 关于zProperties

在上一章中，我们提到了zProperties以及他们在系统中是如何进行继承的。本章将对这些zProperties的含义进行详细的描述，同时还将介绍如何对它们进行设置。Zenoss使用zProperties对各种不同的项目进行指定，而这些项目则控制着系统中的不同信息。zProperties是一个强有力的工具，通过zProperties可以使许多能够重复进行的任务自动执行，同时。对于那些不得不基于单个设备执行且有很多设备的情况下，也可以使用zProperties使任务自动化执行。

在系统中，用户可以一次为所有的项目同时配置zProperties，也可以为一个设备单独配置zProperties，或者是为在设备层次目录树更下一层的任何设备配置zProperties。同样的，zProperties是可以打包的（ZenPack），用户对zProperties进行的设置可以被加入到任何一个ZenPack中。zProperties主要针对系统中的以下一些对象存在：

- 事件 （Events）
- 设备（Devices）
- 服务（Services）
- 网络（Networks）

2. 事件的zProperties

如果要访问事件的zProperties，用户可以从左侧的导航菜单中选择事件（Event），然后点击zProperties标签页。在事件分类层次结构中，用户在任何一层都可以对任何一种事件分类进行zProperties的设置，所要做的仅仅是点击一下该事件分类的zProperties标签页即可。

表 8.1.

属性名称	属性类型	描述
zEventAction	string	事件将要存储到的位置，可能的值有：status\ history \ drop。默认值是status，意味着事件是一个“活动”的事件。History则直接将事件发送到历史表中，而Drop则意味着系统将丢弃该事件。
zEventClearClasses	lines	A list of classes that a clear event should clear in addition to its own class.
zEventSeverity	Int	允许用户覆写事件的级别的值，如果是-1，则被忽略，可能的值在0-5之间。

3. 设备的zProperties

如果要访问设备的zProperties，可以从左侧的导航菜单中选择设备（device），然后点击zProperties标签页。还可以在设备目录树的层次结构中任选一个设备分类的zProperties标签页进行设置，如果要设置单个设备的zProperties，可以先导航到设备页，然后打开页菜单，选择更多选项（More option），然后再选择zProperties标签页进行设置。

表 8.2.

属性名称	属性类型	描述
zCollectorClientTimeout	Int	允许用户设置采集客户端超时时间，单位为秒。
zCollectorDecoding	String	将接收到的字符转为Unicode编码。
zCollectorLogChanges	boolean	是否将变化记录到日志的开关。
zCollectorPlugins	Lines	一个到该设备采集器插件的链接。
zCommandCommandTimeout	Float	等待一个命令完成的时间。
zCommandCycleTime	int	为某设备或者某分类（organizer）执行zcommnads命令时花费的时间。
zCommandExistanceTest	String	***
zCommandLoginTimeout	Float	等待登录提示符的时间。
zCommandLoginTries	Int	尝试登录的次数。
zCommandPassword	int	用命令行登录时要使用的口令。
zCommandPath	String	ZenCommand 插件安装在本地主机上的默认路径，或者是安装在远程主机上的默认路径（zenoss可能会使用SSH来运行命令）
zCommandPort	Int	使用命令方式采集时要连接的端口。
zCommandProtocol	string	使用命令方式采集时所使用的协议，可能的值有ssh/ telnet。
zCommandSearchPath	Lines	搜索zCommand的路径。
zCommandUsername	String	使用命令方式采集时要使用的用户名。
zDeviceTemplates	lines	与该设备关联的模板，由名称关联。.
zFileSystemMapIgnoreNames	String	要忽略的文件系统名称的正则表达式。
zFileSystemMapIgnoreTypes	Lines	不使用
zIcon	lines	表示设备的图标,无论该设备出现在何处(网络拓扑图、设备状态页）都用该图标显示。绝大多数设备，比如windows主机、linux主机以及路由器都有默认设置的图标。
zIfDescription	boolean	在接口列表中是否显示接口描述。
zInterfaceMapIgnoreNames	string	一个正则表达式，用于过滤那些不应当被发现的接口。
zInterfaceMapIgnoreTypes	String	一个正则表达式，用于过滤那些不应当被发现的接口图。
zIpServiceMapMaxPort	Int	要扫描的最高端口，默认为1024.
zKeyPath	lines	到用于访问一个设备的key的路径。
zLinks	Text	输入与设备相关的链接的地方。
zLocalInterfaceNames	string	一个正则表达式，该正则表达式使用接口名称来决定接口上的IP地址是否应当被合并到Zenoss的网络图中。举例来说，环回地址”loopback”接口应当排除。

zLocalIpAddresses	Int	应当从网络图中排除的IP地址IP。举例来说，127.x地址是最有可能被排除的，因为该段地址经常被用于集群内的主机之间互联。
zMaxOIDPerRequest	Int	zenoss的SNMP采集进程才查询信息时发送的OID的最大数量。部分设备的缓存较小，无法处理该信息，因此应当适当把该数字降低。
zPingInterfaceDescription	String	一个类别查询字符串，该字符串通过描述来找到需要被ping的接口。
zPingInterfaceName	String	一个类别查询字符串，该字符串通过名称来找到需要被ping的接口。
zPingMonitorIgnore	boolean	是否ping设备。
zProdStateThreshold	int	生产状态门限值，在该门限值处，Zenoss将开始监视设备，默认值为500，相当于预生产（Pre-Productions）状态。
zPythonClass	String	不使用
zRouteMapCollectOnlyIndirect	Boolean	仅仅采集那些与设备直连的路由器的信息。
zRouteMapCollectOnlyLocal	Boolean	仅采集本地路由器数据（这些路由器通常是手工配置路由而不是通过路由协议学习路由的）。
zSnmpAuthPassword	string	一个用于验证的共享的密钥，该密钥至少需要8个字符长度。
zSnmpAuthType	String	使用 "MD5"或者r "SHA"签名来认证SNMP 请求。
zSnmpCommunities	lines	ZenModeler在采集SNMP信息时将会尝试使用的SNMP community字符串列表。
zSnmpCommunity	string	采集SNMP信息时要使用的Community，如果采集到的信息与ZenModeler采集到的不同，那么snmp采集到的信息将被设置到已被建模的设备上。
zSnmpMonitorIgnore	boolean	是否忽略设备上的SNMP监视。
zSnmpPort	Int	SNMP代理监听的端口。
zSnmpPrivPassword	String	一个共享的密钥，用于加密的SNMP请求，该密钥要求至少有8个字符的长度。
zSnmpPrivType	String	"DES"或者 "AES" 加密算法。
zSnmpSecurityName	string	发出SNMPv3请求时使用的安全用户名。
zSnmpTimeout	Float	SNMP请求超时，单位以秒计算。
zSnmpTries	Int	采集snmp数据尝试的次数。
zSnmpVer	string	使用的SNMP版本，可用值有v1, v2.
zStatusConnectTimeout		Zenstatus进程测试连接超时。
zSysedgeDiskMapIgnoreNames		目前未使用
zTelnetEnable	Boolean	登录至Cisco设备时，enable命令将使得在命

		令行采集数据期间的访问可用。
zTelnetEnableRegex	String	匹配enable提示符的正则表达式列表。
zTelnetLoginRegex	String	匹配登录提示符的正则表达式。
zTelnetPasswordRegex	String	匹配口令提示符的正则表达式。
zTelnetPromptTimeout	Float	等待命令提示符返回的时间。
zTelnetSuccessRegexList	Lines	匹配命令提示符的正则表达式列表。
zTelnetTermLength	Boolean	在一台Cisco设备上，将终端（term）的长度设置为0。
zWinEventlog	boolean	是否发送日志
zWinEventlogMinSeverity	int	设置从windows eventlog采集的日志的最低级别。该数字越高，则事件级别越低，1为等级最高的事件，5为等级最低的事件。
zWinPassword	String	登录远程windows主机所要使用的口令。
zWinServices		
zWinUser	String	登录远程windows主机要使用的用户名。
zWmiMonitorIgnore	Boolean	使用该选项打开或者关闭所有的WMI监视。
zXmlRpcMonitorIgnore	boolean	使用该选项打开或者关闭所有的XML/RPC监视。

4. 服务的zProperties

如果要访问服务的 zProperties，用户可以从左侧导航菜单中选择服务（Services），然后点击 zProperties标签页。同样的，还可以在服务层次目录树中的任何一层中，点击zProperties标签页来访问该层的zProperties。.

Table 8.3.

属性名称	属性类型	描述
zFailSeverity	Int	服务失败时发送事件的级别。
zHideFieldsFromList	Lines	服务实例列表中要隐藏的列。
zMonitor	boolean	是否要监视一个服务

5. 网络的 zProperties

要访问网络的zProperties，用户可以从左侧的导航菜单中选择网络（Networks），然后点击zProperties标签页。同样的，用户也可以在网络页中的任何子网中访问zProperties标签页。

Table 8.4.

属性名称	属性类型	描述
zAutoDiscover	boolean	Zendisc是否要在本网络中执行自动发现。
zDefaultNetworkTree	Lines	在创建网络时要使用的网络掩码的列表。默认为24，如果一个网络使用的子网掩码小于24，则在网络树顶层将使用24作为掩码。

zPingFailThresh	int	在Zendisc移除设备之前要发送的ping（无需返回）的数量。

6. 制造商的 zProperties

Table 8.5.

属性名称	属性类型	描述
zDeviceClass	String	未来使用
zDeviceGroup	String	未来使用
zSystem	string	未来使用

第9章. 设备库与配置

1. 什么是Zenoss中的设备库与配置

设备库与配置是指系统中的设备数据库以及系统中有关设备信息的集合。设备库与配置通常与“建模”相关，Zenoss有三种设备建模手段，分别是自动发现、逐一建模以及通过载入XML文件方式进行建模。

2. Zenoss如何对设备建模

Zenoss可以通过使用SNMP, SSH, 或Telnet 协议进行建模。每种建模技术在模型中的信息丰富度各有不同。SNMP通常提供最为完善的信息，而SSH/Telnet 手段通常在SNMP 代理无法报告设备的某些特定信息时作为补充手段使用。

3. ZenModeler进程

Zenoss通过使用“ZenModeler” 进程进行建模。ZenModeler进程不断对系统中的设备进行遍历并试图自动发现每个设备的子组件，设备的子组件包括网络接口、文件系统、进程、IP服务等等。默认情况下，系统每6小时进行一次重新建模。对大型的企业环境来讲，该频率可能过于频繁。通常情况下，可以通过Cronjob来设定重新建模并使其每天仅进行一次重新建模工作。

4. 添加一个设备

Zenoss能够对加入系统的设备进行建模和监视，以下我们将详细描述如何添加一个单独的设备。

1.在左侧导航菜单中选择Add Device（添加设备），如下图9.1所示：

2.在设备名称区域内输入设备的网络(DNS)名称或者IP地址，同时在该界面中用户还可以指定其它与设备有关的可选信息。

3. 在设备分类下拉列表中选择设备分类，在本例中我们将设备归类为Windows server,所以我们选择/Server/Windows 作为该设备的设备分类路径。

4. 选择一个发现协议，用户可以选择snmp或者none，我们将在下面对章节中描述Zenoss如何使用这些方法对设备进行建模。.

5. 设备名称、设备分类路径以及发现协议是添加设备时必须选择的，Zenoss会尝试着去填余下的信息，这些余下的信息也可以晚点时间添加。用户手工填写的信息可能会与Zenoss自动发现的信息冲突。

注：如果不是在/Network,分类中添加 Cisco 路由器，那么应当将 zIfDescription 属性设置为 True.，这将会给用户带来更多的有关 Cisco 路由器的信息。默认情况下， /Network 分类的 zIfDescription 属性已经被设置为 True。

6. 将页面滚动至底部，然后点击添加设备（Add Device）按钮。此时系统将显示一个状态页，在该页中将显示 Zenoss 的有关收集设备信息的相关操作日志。

7. 设备添加成功后，在页面底部将会显示有一个链接，点击该链接，用户将被系统导航至刚刚添加的设备页面，用户此时将首先看到设备的状态标签页，如下图 9.2 所示：

图 9.1. 添加设备页

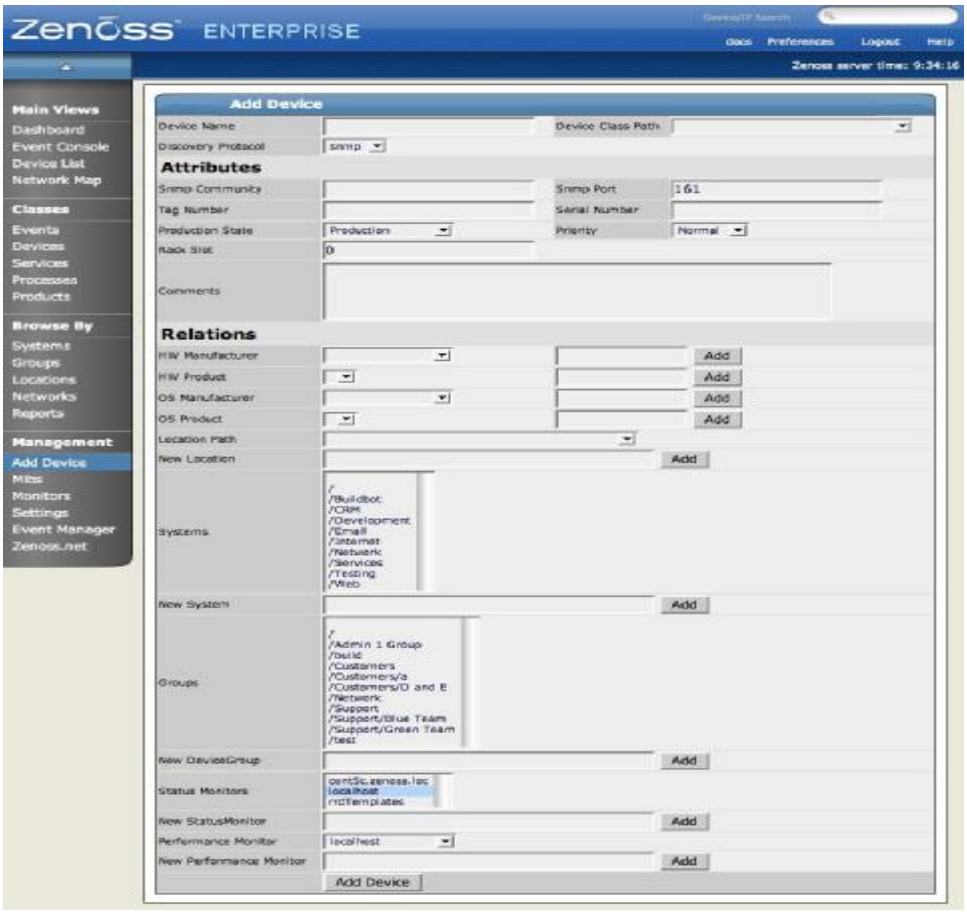


Figure 9.2. 主设备页

5. 在设备分类层次树环境下添加单个设备

用户可以在设备分类层次树环境下添加单个设备，这意味着，设备将被添加至用户选择的设备分类中。否则，默认情况下设备将被添加至/Discovered 分类中。要在设备分类层次树环境下添加单个设备：

1．在设备树中首先找到一个位置，该位置就是用户想要将该设备添加到的位置。

2．打开页菜单，选择管理（Manage）选项，之后选择添加设备（Add Device）选项，之后系统将弹出添加设备对话框。

图 9.3. 在设备分类树层次环境下添加设备对话框

3．该对话框内容的填写与前文提到的添加设备时代操作一致，唯一不同的是，在这种情况下，设备的分类已经被用户预先指定。

4．点击 OK 按钮，设备将被加入到用户选中的设备分类，之后用户将看到设备页面。.

6. 设备的自动发现

Zenoss 能够通过 SNMP-walk 来遍历整个网络和路由表，然后对每个设备进行单独建模，并将这些自动发现的涉恶被一次性全部加入 Zenoss 的数据库。该进程称为 ZenDisc 进程。如果用户需要执行自动发现过程的话，那么安装 Zenoss 的主机上就必须运行有 SNMP 的代理。以下是将给定网络或者子网中所有设备加入 Zenoss 系统的方法：

1.从左侧导航菜单中选择网络（Networks），系统将显示网络概览（Networks Overview）页，参见图9.4。

2．选中一个网络，该网络中的设备是您希望添加至系统中的设备，同时您也可以使用子网的表菜单来向子网列表中增加一个子网。

3．一旦您选择了网络后，您可以打开子网的表菜单并选择发现设备（Discover Devices）。

4.之后，系统将显示设备状态页，在该页面上将显示所有的设备采集信息。

图 9.4.网络概览页

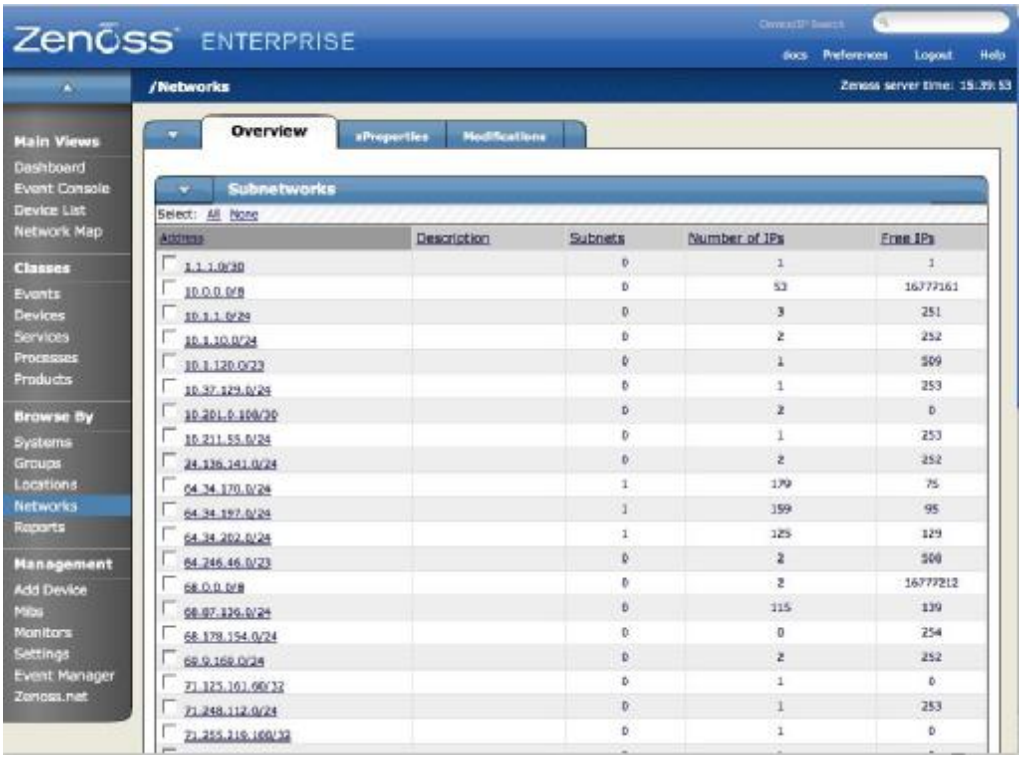


图 9.5. 设备的自动发现

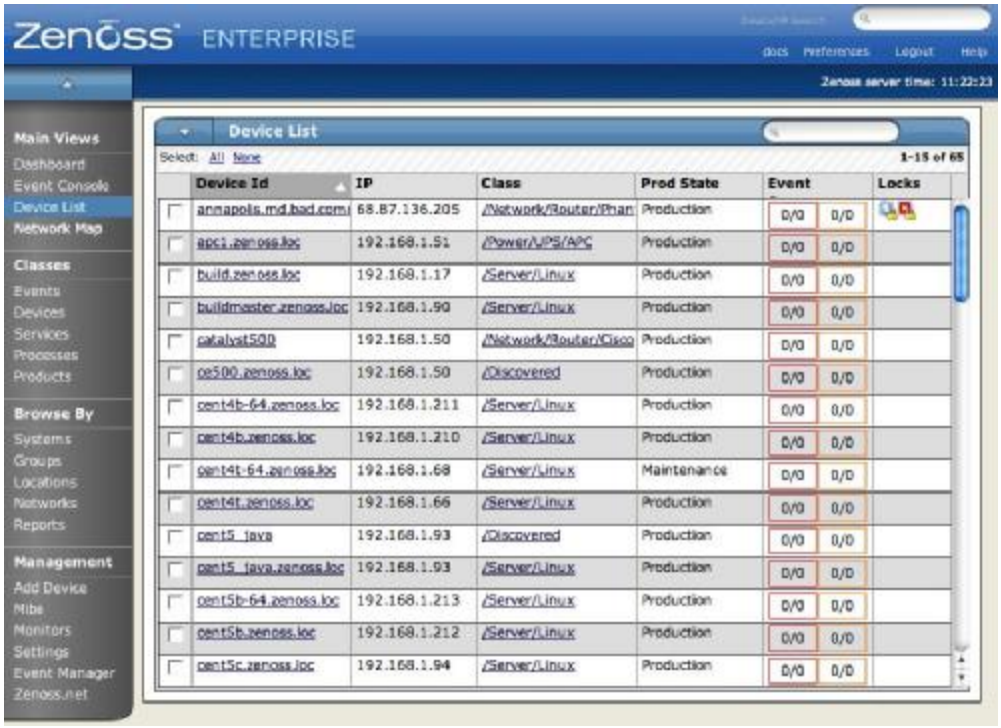
该命令首先对监视主机进行建模，之后将遍历所有其能够发现的路由器的路由表，只要设备的SNMP可用或者设备的zAutoDiscover property属性被设置为true，设备的自动发现就可以继续进行。

通过这种方式发现的路由器将被自动放在/Network/Router.这个分类下面。通过自动发现而找到的设备会被系统放置在/Discovered路径下，之后这些自动发现的设备应该被移至一个更确切的目录，服务器通常以OS进行分类，所以windows主机应当放在/Server/Windows.路径下，使用设备的编辑(Edit)标签页，我们可以添加设备的更过信息，比如设备属于哪个系统，设备的位置等等。

7. 设备列表

设备列表显示了系统中的所有设备，用户可以在设备列表中搜索所有的设备，同时也可以执行部分管理任务以对单个或者多个设备同时生效。要访问设备列表，在左侧导航菜单中选择设备列表(Device List)，之后系统将显示设备列表。

图9.6. 设备列表



7.1. 在设备列表中管理多个设备

用户可以在设备列表中一次选中多个设备后，使用设备列表的页菜单来管理多个设备，用户可以选择的管理项有：

- Move to class – 将多个设备同时移至新的设备分类
- Set Groups – 将多个设备同时指派给一个分组.

- Set Systems – 将多个设备同时指派到一个系统
- Set Location – 将多个设备同时设定到一个位置
- Set Monitor – 同时指定多台设备归哪个监视器（主机）进行监视
- Delete devices – 将设备从系统中删除
- Lock devices – 为设备提供配置锁定

8. 单个设备的页标签

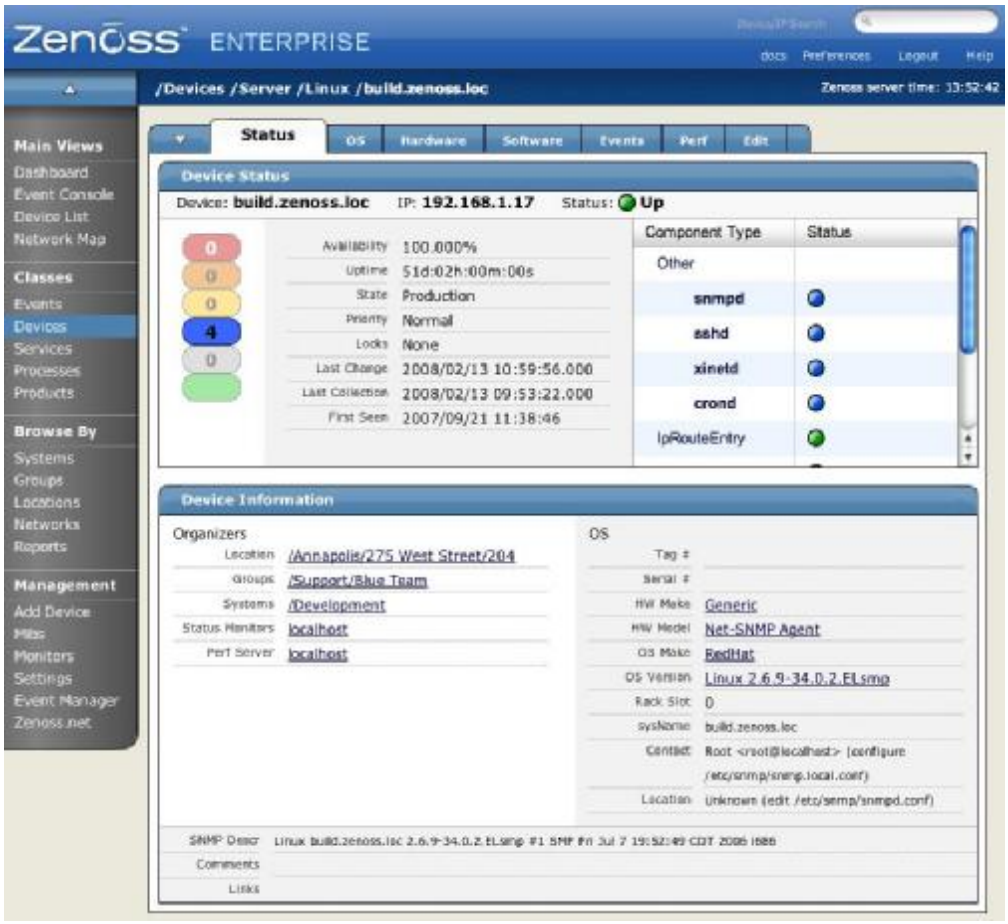
一旦Zenoss完成了对设备的发现和建模，Zenoss将会为设备指定特定的属性，同时将这些属性按照分类分成不同的信息标签页。

8.1. 设备状态页标签

用户点击一个设备时，默认先显示设备的状态页标签。

如图所示，设备状态页标签上有一个设备状态表，用户可以一眼就能看到重要的设备状态信息。在设备状态表达左侧，系统按照不同颜色显示不同级别的事件数目，不同的颜色代表不同的重要级别。用户可以点击“事件彩虹”来查看设备上的事件。

图 9.7. 单个设备的状态页标签



在设备状态表达右侧显示的是设备可用性、运行时间、最后一次采集信息和修改配置信息的时间。在设备状态表的右侧显示的是组件状态列表。列表中的每个条目是一种设备组件，这些组件可能是IpService, WinService, IpRouteEntry, IpInterface, CPU, FileSystem或者其它。每个设备组件的状态由系统对其采集的数据决定。如果IpService状态是绿色的，那么说明该设备上所有被监视的IpServices的功能是正常的。如果一个受监视的IpServices与一个事件关联，那么该组件将以与该事件关联的级别的颜色显示出来，如果一个事件未与系统已知的设备组件进行关联，那么该事件将被归入其它设备组件类型。如果用户点击组件类型，用户将被导航至OS标签页，用户可以在该标签页上管理设备组件。

8.1.1. 例：设备状态标签页

1.点击设备的 "OS" 标签页，如果没有IpInterface eth0，可以在表菜单中点击e "Add..."来添加一个IpInterface，一旦您添加了这个组件，您将会希望这个组件处于系统的监视之下：

2. 在IpInterface的编辑标签页上，将Monitor设置为"True"并点击 "Save"按钮。.

3.现在重新回到设备状态标签页，此时IpInterface类型将会显示在状态标签页中。

4. 现在，可以通过发送一个与该组件关联的事件来更新组件状态表，我们可以在左侧的导航菜单中，点击事件（Events）.

5.从页菜单中选择"Add Event" 菜单项，输入设备的名称，在组件域中输入 "eth0"，将级别设为"Critical"定点击 "OK"按钮。

6. 现在，再次发送一个事件。仍旧使用"Add Event"菜单项，在组件域中输入"Not a known component" (或者其它字符串，且该字符串不是该设备上的任何一个设备名称) ,将重要等级设置为"Error" 并点击"OK"按钮。.

7. 重新刷新设备状态页标签。您将会看到IpInterface类型分组中有了一个"eth0"组件，其显示为红色，这说明该组件有一个Critical 级别的事件发生。

8. 如果点击状态球，此时用户将被导航至该设备的事件页标签，用户将会看到一个新的类型分组，Other，该组件状态的颜色应该是橙色，因为事前我们将一个Error级别的事件与该组件进行了关联。

用户还可以看到显示设备详细信息的设备信息表。设备信息表的左侧有Location, Device Groups, Systems 以及status monitors和performance monitor选项。用户可以点击一个选项或者一个监视器来查看该分类的状态。设备信息表的右侧显示设备的硬件和操作系统信息。

8.2. OS (操作系统) 页标签

OS页标签中包括操作系统的逻辑组件：

•接口

• IP服务

• Windows服务

•文件系统

• 路由

• OS进程

图 9.8. 单个设备的 OS 页标签

8.2.1. 文件系统监视

文件系统监视仅在系统拥有良好的HOST-RESOURCES mib的情况下才可使用。文件系统监视显示已用的文件块和文件块总量、可用字节数、已用字节数以及使用百分比。用户可以设定门限值，比如，如果文件系统利用率达到90%时系统将产生一个告警事件。

8.3. 硬件页标签

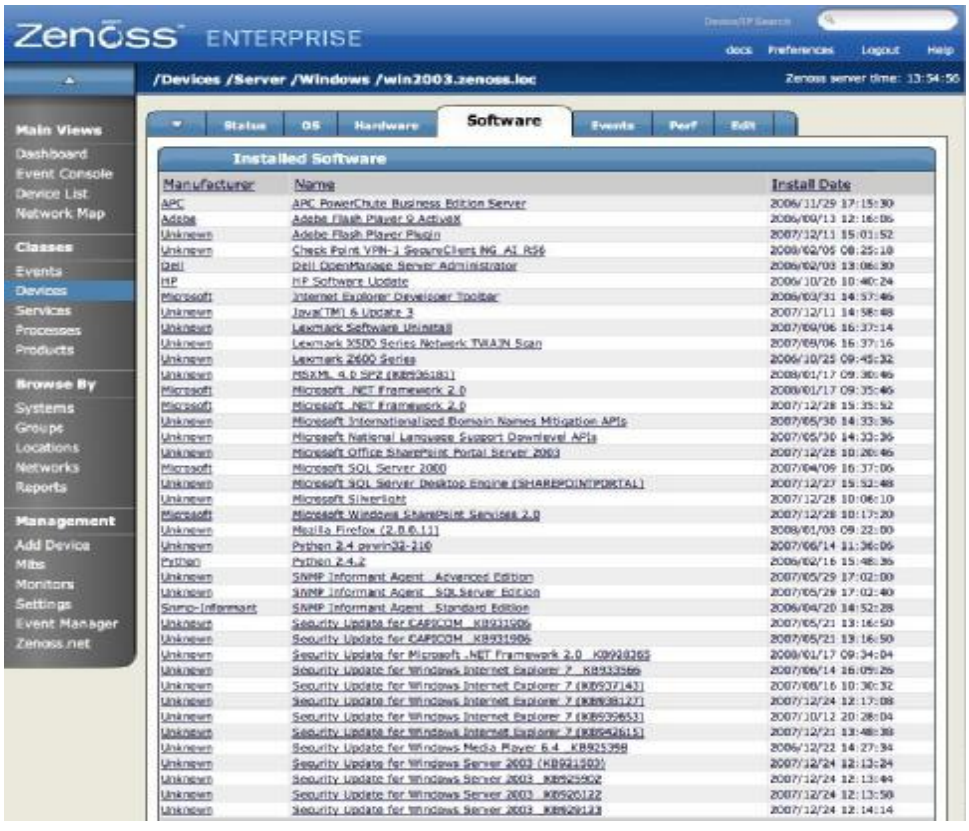
硬件标签页显示设备的可用/已用内存、可用交换分区以及CPU的相关信息，如下图9.9所示：

图 9.9. 单个设备的硬件页标签

8.4. 软件页标签

软件标签页中列出系统中安装的所有软件，这些软件可按照制造商、软件名称以及安装日期进行排序。

图 9.10. 单个设备的软件页标签



8.5.事件页标签

事件标签页与其它事件视图非常类似，用户可以使用组件（Component）、事件分类（EventClass）以及计数（Count）对事件进行分类，但不仅限于此。用户可以通过事件级别、状态或者一个正则表达式对事件进行过滤，同时还可以在这里对事件进行移动、映射或者确认。

图 9.11. 单个设备的事件页标签

8.6. 历史页标签

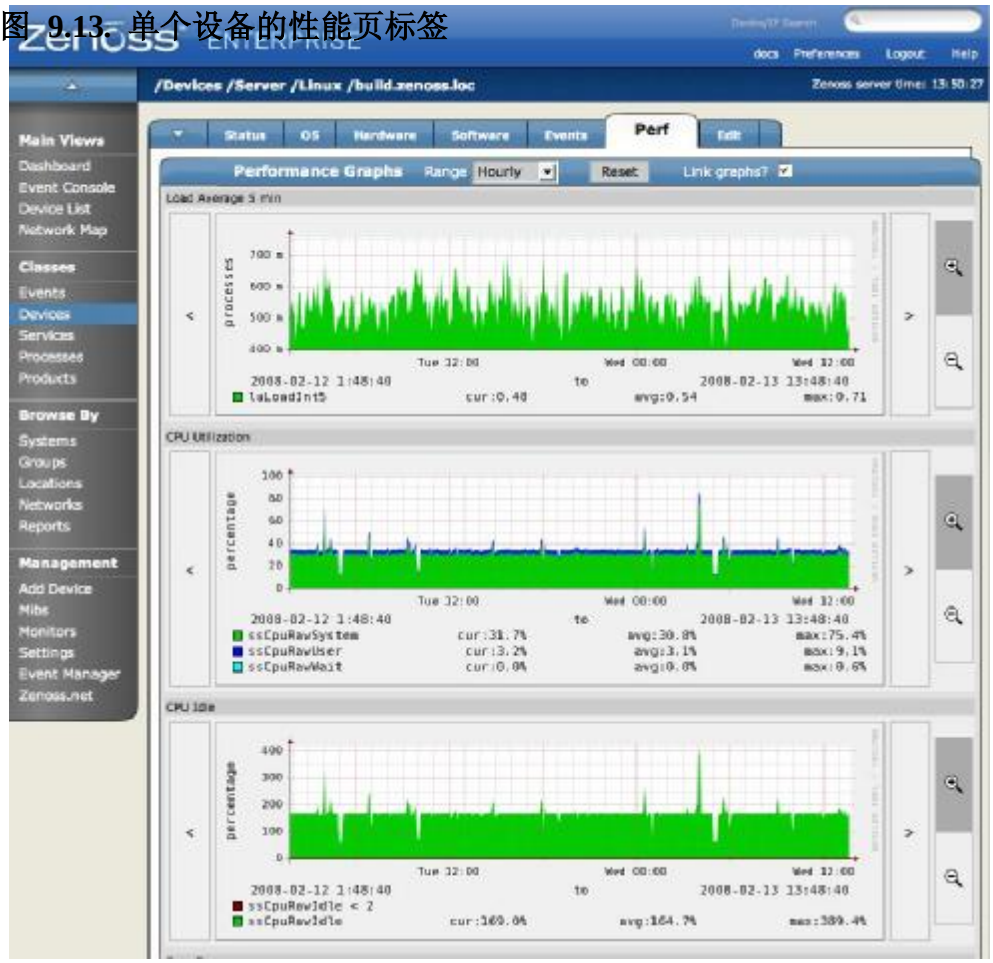
历史标签页显示那些生命周期已经结束的事件或者已经被存档的事件。

图 9.12. 单个设备的历史页标签

8.7. 性能页标签

性能标签页显示当前被选中的设备的性能图表，该性能图表显示设备的5分钟平均负载、CPU利用率、CPU空闲率、可用内存、可用交换分区，这些性能图表可被设置成每小时、每天、每周、每月或者每年更新一次。

图 9.13. 单个设备的性能页标签



8.7.1. 浏览性能图表

用户可以使用图表左右两侧的控件来浏览图表，用户可以使用左右箭头和放大图标来浏览性能历史，放大缩小性能图表。

8.7.1.1. 取消图表联系

默认情况下，用户在图表上点击左右箭头或者放大缩小图标时，性能图表是一起变化的。如果希望操作仅对一个图表有效，用户可以去掉Link Graph后面的对勾。

8.7.1.2. 重新设定视图

点击重设（ Reset）按钮，系统将返回默认的图表（初始视图）。用户可以在视图上通过点击左右箭头来向前或者向浏览数据。每点一次箭头，将移动一个时间单位。

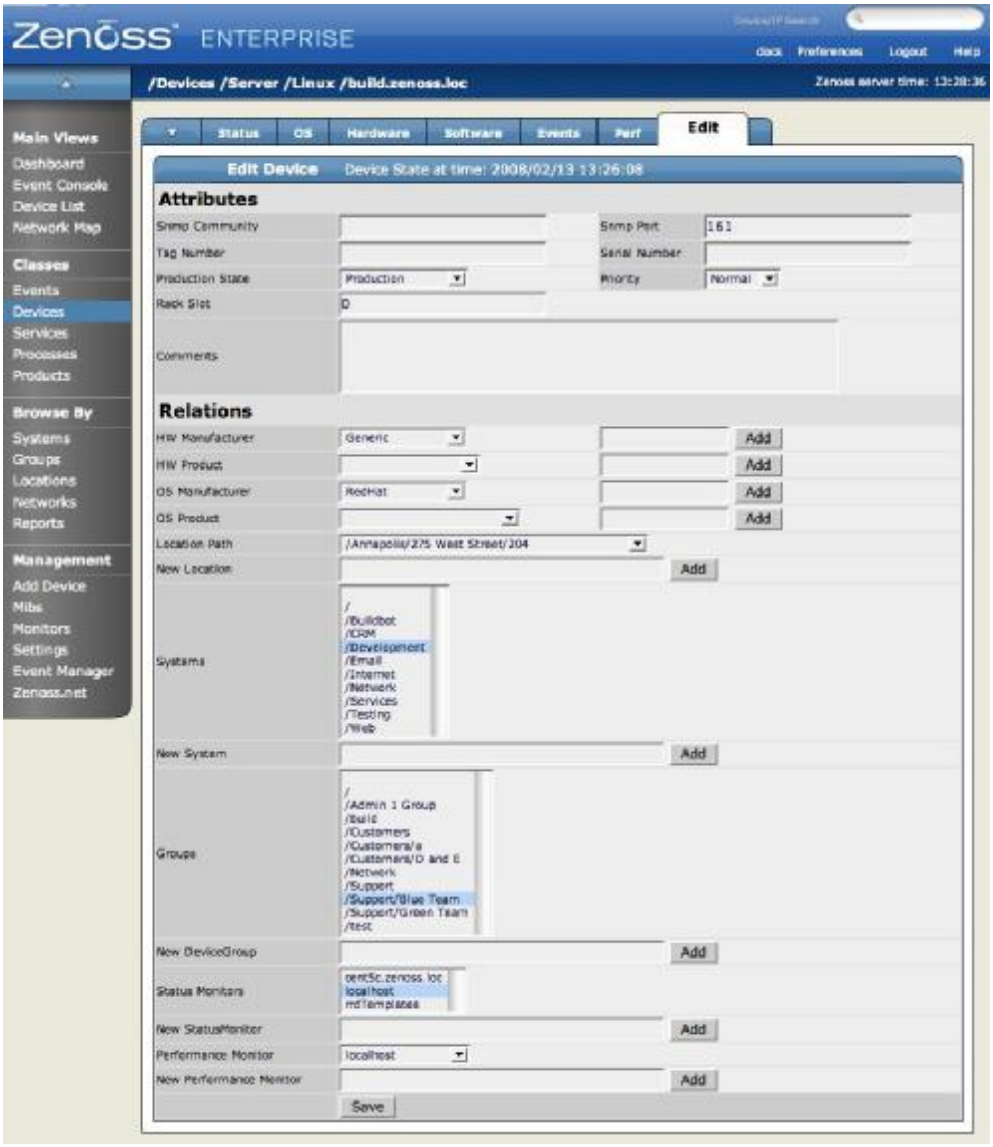
8.7.1.3. 数据缩放

用户可以首先点击放大 (+)图标，然后点击相应图表来放大图表中的数据，同样可以先点击缩小(-)图标，然后点击图表来缩小数据。

8.8. 编辑页标签

用户可以使用该标签页来编辑设备信息。

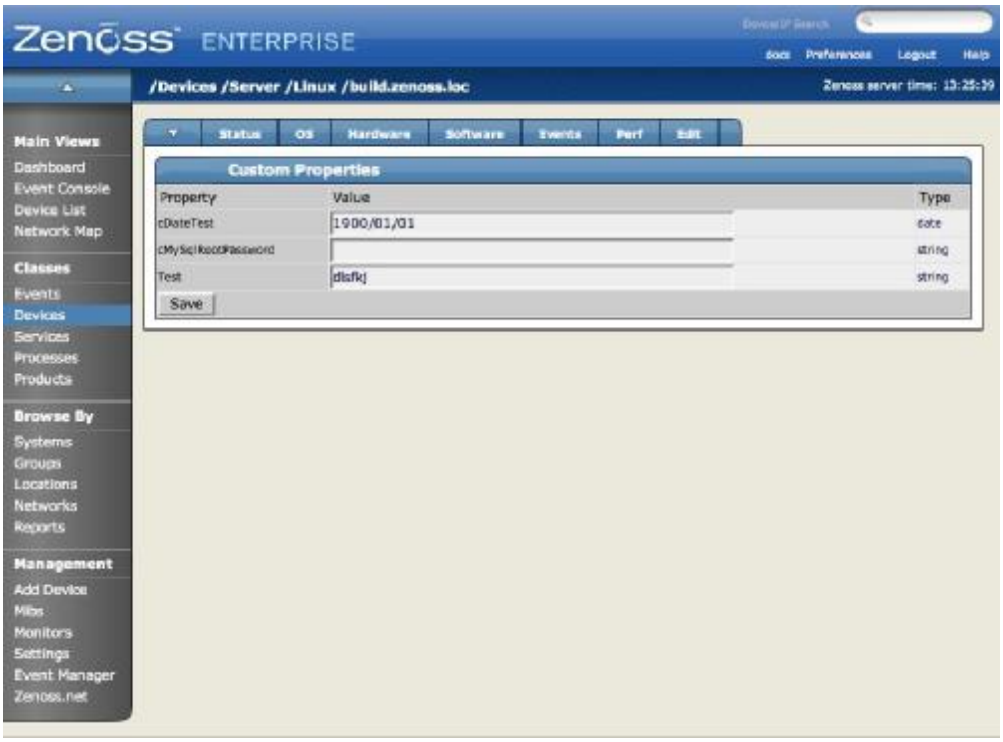
图 9.14. 单个设备的编辑页标签



8.9. 定制标签页Custom Tab

该标签页允许用户设定自定义模式中某些属性的值，可通过设备页菜单，选择More后选择Custom选项进入定制标签页。

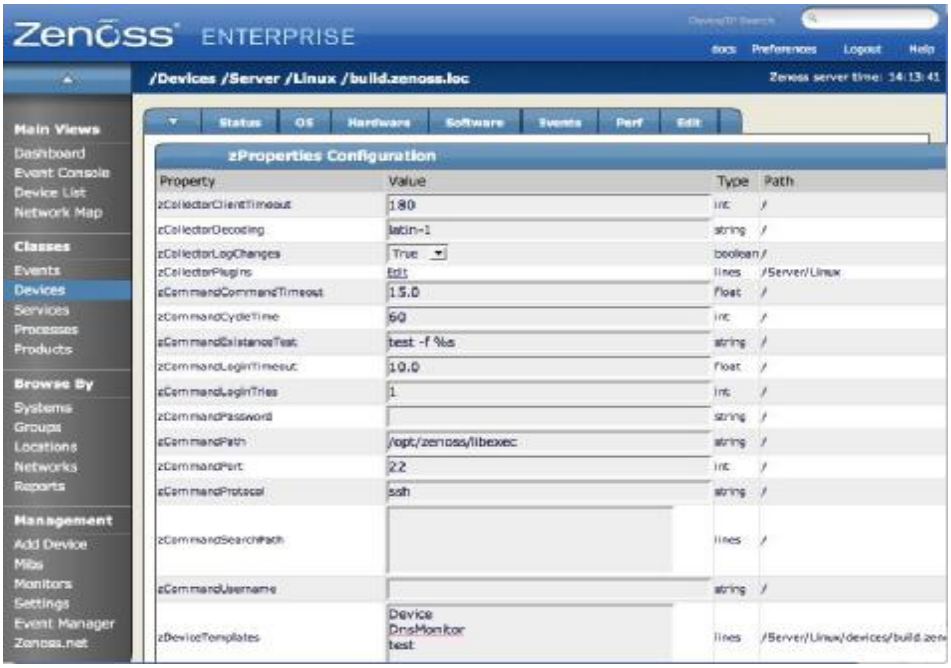
图 9.15. 单个设备的定制标签页



8.10. zProperties 页标签

在设备的zProperties标签页中，用户可以为所有的设备配置zProperties,也可以为单独的设备配置zProperties，同时还可以为当前设备所处层次之下任何一层的任何设备配置zProperties。如果要为所有的设备配置zProperties,可以从设备概览标签页(Device Overview tab）中点击thezProperties 标签页，点击设备名称,然后点击该设备的zProperties标签页。要访问zProperties标签页,需要打开设备的表菜单,选择更多(More)选项,之后选择zProperties。

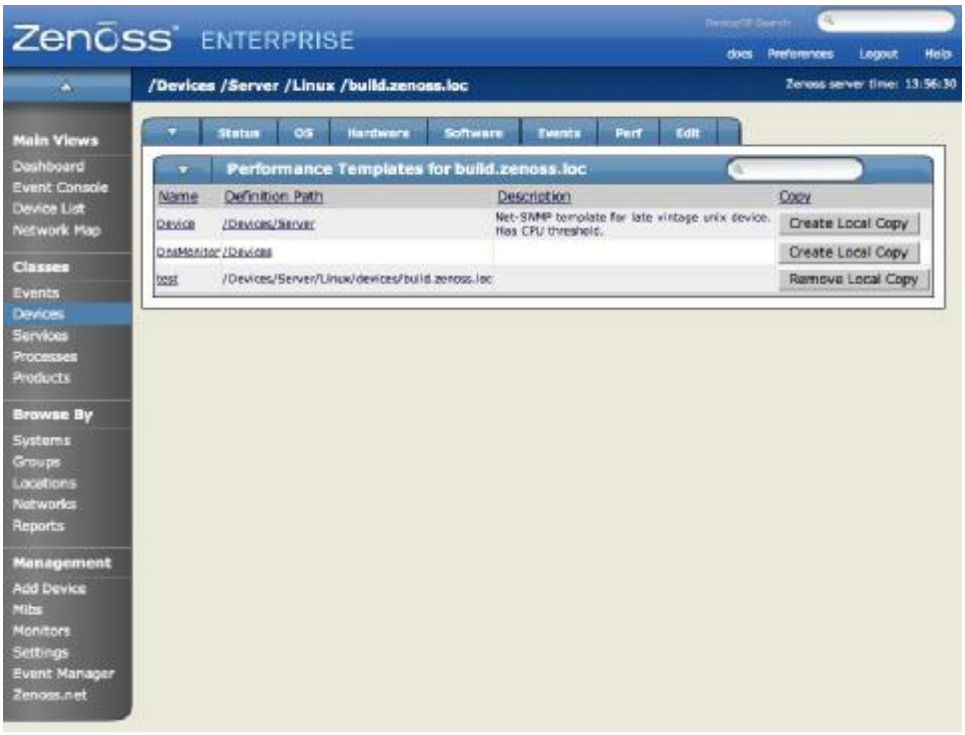
Figure 9.16. 单个设备的zProperties 页标签



8.11. 模板页标签

模板标签页显示所有绑定到设备或一组设备的性能模板，用户可以以打开设备的表菜单，选择更多（More）选项，然后选择模板，这样就可以访问模板页标签。

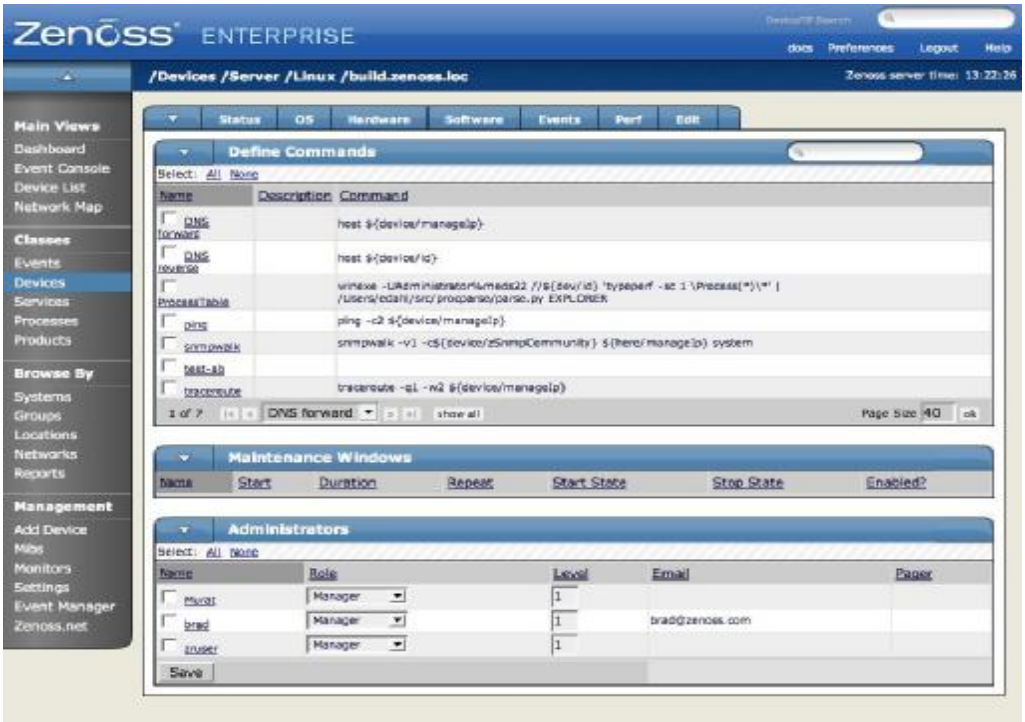
Figure 9.17. 单个设备的模板页标签



8.12. 管理页标签

打开设备的页菜单，选择更多（More）选项，然后选管理（Administration），可以访问设备的管理页标签。

图 9.17. 单个设备的管理页标签



8.13. 采集器插件（Collector Plugins）页标签

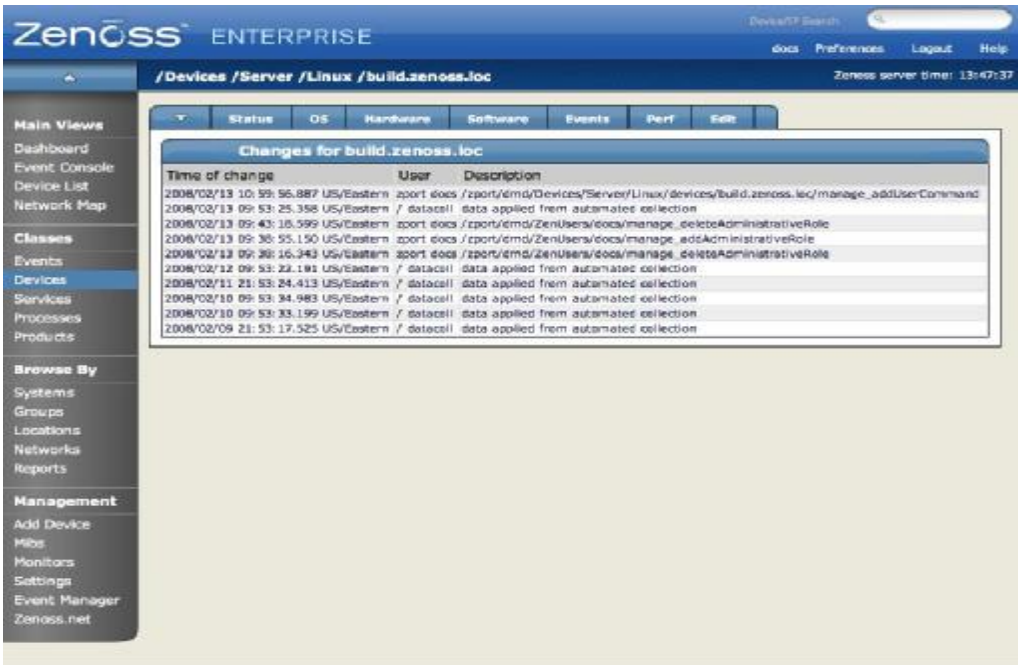
采集器插件页标签显示当前设备所有可用的插件，用户可以打开表菜单，选择更多（More）选项，然后选择采集器插件来访问采集器插件页标签。

图 9.19. 单独设备的采集器插件页标签

8.14. 修改(Modifications)页标签

修改页标签记录用户通过Zope接口进行的变更，用户可以打开设备的页菜单，选择更多（More）选项，然后选择Modifications来访问修改页标签。

图 9.20. 单独设备的修改页标签



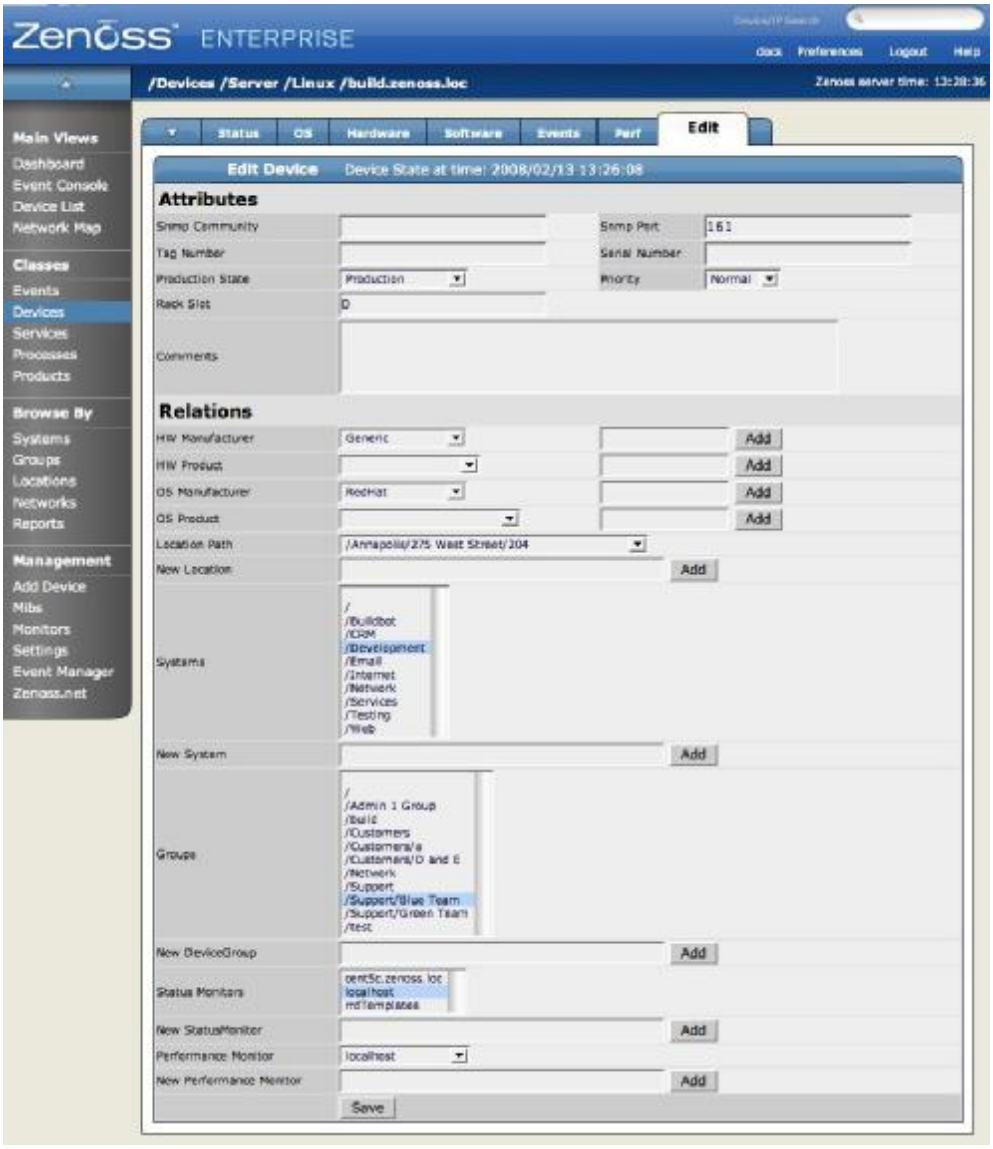
9. 通过名称或者IP地址搜索设备

第一个也是最重要的搜索设备的方法是，在顶部的”Device/IP Search”框中输入设备的名称或者IP地址，然而这种方法并非总是有效，因为用户很有可能忘掉设备的名称或者IP地址。这种情况下，就可以使用左侧导航菜单中的”Browse by”功能。”Browse by”允许用户按照系统、分组、位置以及报告来浏览设备。按系统浏览允许用户按照特定的设备类型，比如文件服务器、打印机、基础设施来浏览设备。按分组浏览允许用户按照自己事前定义的设备分组进行浏览，按位置浏览与按分组浏览相同，只不过是按照位置对设备进行了组织。

10. 编辑设备配置

如果用户希望编辑设备的配置，首先要通过搜索或者使用“Browse By”定位到该设备，之后再设备页面中选择编辑（Edit）标签页，如下图所示，用户在该标签页中可以对设备的配置信息进行任意修改。

图 9.21. 设备的编辑页标签



11. 管理设备

如果希望管理设备的各种属性，可以使用设备页菜单中的管理菜单项，操作顺序是More... Manage... Run Commands... 在下一章中，我们将对设备管理的内容进行详细描述。值得注意的是，用户可以通过继承来设定多个管理活动。比如，用户可以首先对导航至分类、系统、分组或者位置，然后重复上述操作步骤，就可以一次性设定多个管理活动。

11.1. 设备重新建模

如果要让Zenoss获取设备信息并重新采集设备的配置信息，需要进行以下步骤：

1．导航至设备

2．从设备的页菜单中选择管理（Manage）,然后选择设备建模（Model Device），此后设备将被重新建模，系统将显示重新建模的状态页。

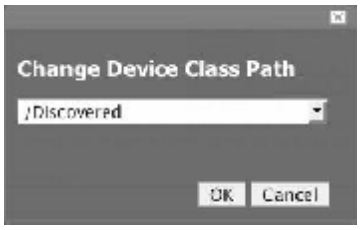
11.2. 改变设备分类

如果要改变一个设备的分类，必须采取以下步骤：

1．导航至设备.

2．从设备的页菜单中选择管理（Manage）,然后选择改变分类（Change Class），系统弹出改变设备分类对话框，如下图所示。

图9.22. 改变设备分类对话框



3．从下拉菜单中为设备选择新的分类。

4．点击OK.按钮，当前设备的分类即被改变。

11.3. 重新设定设备管理IP

如果要重新设定设备的管理IP，可以采取以下步骤：

1．导航至设备。

2．从设备的页菜单中，选择管理（Manage），然后选择重设IP（Reset IP）,系统将弹出重设IP对话框。

3．输入设备新的IP (或者留空以便由DNS决定IP)。

4．点击 OK按钮后设备的IP将被重新设定。

图9.23. 重设IP 对话框

11.4. 设备重命名

在Zenoss中如果对设备重命名，必须采取如下步骤：

1．导航至设备。

2. 从设备的页菜单中选择管理（Manage），然后选择重命名设备（Rename Device） and then 。

图 9.24. 设备重命名对话框

3．在ID域中输入设备的新名称。

4．点击OK按钮后设备将在Zenoss中被重新命名。

11.5. 锁定设备配置

用户可以通过几种手段来锁定设备的配置，以防设备重新建模时设备的配置信息被覆写。有两个层次的设备锁定，即用户可以通过锁定来防止设备配置被删除或者更新，同时用户还可以解锁设备配置信息，操作步骤如下：

1．导航至设备。

2．从设备的页菜单中，选择管理（Manage），然后选择锁定（Lock），之后系统弹出锁定配置确认对话框。

图 9.25. 配置锁定编辑对话框

3．如果用户希望在操作被锁定所阻塞而发送一个事件时，则选中前面的checkbox控件。

4.如果希望改变锁定的配置，则在上图中的四个按钮中任选一个，即可改变配置的锁定选项。

11.6. 重新设定设备的Community

要修改设备的community字符串，需要采取以下步骤：

1．导航至设备。

2．从设备的页菜单中，选择管理（Manage），然后选择重新设定Community(Reset Community)。

11.7. 将配置变更推回至Zenoss 系统

用户可以将设备配置的任何变更推回至Zenoss系统以及相关的采集器，在此过程中无须等待设备的重新建模。要将配置变更推回至Zenoss，需要采取以下步骤：

1.导航至设备。

2．从设备的页菜单中，选择管理（Manage），然后选择推回变更（Push Changes）。此时一个状态页将显示在屏幕的右上方，该状态页确认变更已经被推回至系统。

11.8.清除心跳

要清除与一个特定设备关联的心跳，需采取以下步骤：

1．首先导航至设备。

2．从设备的页菜单中，选择管理（Manage），之后选择清除心跳（Clear Heartbeats.，该设备的心跳将被移至事件历史。之后系统显示ZenEventManager的Edit标签页。在这里进行用户想要的修改然后点击保存按钮。

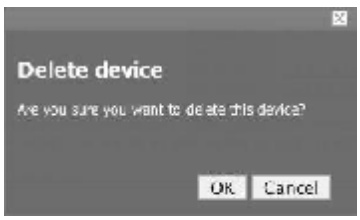
11.9. 从系统中删除设备

要从Zenoss中删除设备：

1．导航至设备。

2．从设备的页菜单中，选择管理（Manage），然后选择删除设备（Delete Device），之后系统将弹出删除设备确认对话框。

图 9.26. 删除设备确认对话框



3．点击OK按钮后设备将从Zenoss系统中删除。

12. 使用SNMP对设备进行建模

以下将介绍几种使用SNMP对设备建模的方法.

12.1. 测试并检查设备是否正在运行 SNMP

首先使用以下命令测试设备上是否正在运行SNMP:

```
$ snmpwalk -v1 -c communityString gate system
```

如果该命令没有超时，则说明设备上已经安装SNMP并且运行正常。

12.2. 使用SNMP对远程Windows主机建模

默认情况下Windows操作系统并不安装SNMP。如果要安装SNMP，要采取以下步骤：
开始 ->控制面板->添加或删除程序->添加/删除Windows组件。
选中管理和监视工具并安装它们。接下来，需要到系统服务中对其进行配置。步骤如下：
控制面板->管理工具->服务，启动SNMP服务以及SNMP Trap服务。设置SNMP Community字符串。
如果希望监视处理器和内存，则需要在windows主机上安装SNMP-Informant，用户可以到 http://www.snmp-informant.com 站点上下载该软件。如果希望采集Windows Event logs或者使用syslog产生的windows主机日志，用户可以从http://syslogserver.com/syslogagent.html 上下载并安装syslogAgent。

12.3. 使用SNMP对远程Linux主机建模

如果要监视Linux主机，则Linux主机上必须安装SNMP。目前比较主流的Linux下的SNMP工具是net-snmp。该工具需要下载、安装和配置，此后才能使用SNMP对Linux主机进行监视。

12.4. 使用SNMP对Cisco设备建模

Cisco 设备上都装有SNMP. 然而，你必须像网络的其他设备一样为每台思科设备配置相同的snmp口令。

13. 使用 SSH/命令进行建模

在某些情况下，有些系统并不提供Zenoss安全访问所需的SNMP代理，在有些生产系统中，某些设备通常只提供SSH的访问方式。如果无法使用SNMP进行设备建模，那么就必须使用命令插件来对设备建模。注意，这里的建模命令插件与Zenoss其它地方中用到的性能命令插件是不同的，不要混淆。

每个zenoss内置的建模命令插件都通过其运行的平台进行区分。如果想要查看需要建模的系统的平台是什么，可以在该设备的shell下输入命令 “uname”。Zenoss的建模命令插件目前兼容Linux和Darwin操作系统，而且还可以在上述两种操作系统之外进行扩展，但创建新的建模插件的过程不在本手册的讨论范围。

如果使用命令插件对设备建模，首先使用协议“none”将设备加入Zenoss系统，然后选择希望使用的插件，步骤如下：

1．到添加设备页面。
2．将发现协议设置为“None”。
3．设备被添加到系统后，导航到该设备页并查看其zProperties标签页。
4．如有必要，分别设置设备的username和password 的zCommandUsername 和zCommandPassword属性(或使用RSA/DSA 密钥来建立无口令验证)
5．点击“Edit”。
6．点击右边的添加域以获得一个完成的命令插件列表。
7.确保zenoss.cmd.uname位于左手边的第一个位置。
8．点击你希望移除的插件旁边的X，并把其它插件拖至左侧。
9．重新对设备建模。

13.1. 利用/Server/Cmd设备分类对设备进行SSH监视

/Server/Cmd设备分类是一个用于举例的分类，该分类用于说明使用SSH对设备进行建模和监视。CollectorPlugins 已经如上文所述进行了修改,并且通过SSH采集数据的设备、文件系统、以太网接口已经建好。用户可以把该设备分类当作是自己配置的一个参考，或者是，如果有一个设备需要建模或者通过SSH命令进行监视，用户可以把这个设备直接放入该分类，这样用户就可以使用系统预定义的模板和zProperties。同时，用户需要为每台设备分别设置SSH登录信息，也就是zCommand-Username 和zCommandPassword 两个属性。

14.使用端口扫描（PortScan）对设备建模

用户可以通过端口扫描的方式来对基于IP的设备进行建模，步骤如下：

1．到设备的zProperties标签页。
2．将zTransportPreference 的值更改为“portscan”。
3．对设备重新建模。

14.1. 利用/Server/Scan Device分类对设备进行端口扫描监视

/Server/Scandevice是一个用于举例的设备分类，该分类用于说明如何使用端口扫描的方式对设备进行监视。用户可以把该设备分类当作是自己配置的一个参考，或者是，如果有一个设备需要通过端口扫描进行监视，用户可以把这个设备直接放入该分类，这样用户就可以使用系统预定义的模板和zProperties。

15.建模插件（Modeling Plugins）

Zenoss使用plug-in maps将真实世界的信息与标准的Zenoss模型进行映射。插件的输入信息可能会来自SNMP、SSH或者Telnet。通过插件名称与zProperty和zCollectorPlugins匹配，用户可以选择在某个设备上运行哪一个插件。

- DeviceMap – 采集设备的基本信息，比如操作系统类型以及硬件型号。
- InterfaceMap – 采集设备上的接口列表。
- RouteMap – 从设备上采集路由表。
- IpServicesMap – 采集设备上运行的IP服务信息。
- FileSystemMap – 采集设备上的文件系统信息。

15.1. 获取设备的建模插件列表

建模插件受一个正则表达式的控制，该正则表达式匹配这些插件的名称。用户在设备页中打开页菜单，选择更多选项（More），然后选择采集插件（Collector Plugins），就可获取设备上的插件列表。采集插件标签页显示设备上所有可用的插件。

16.建模过程调试

Modeler进程可在单台设备上从命令行方式运行。如果使用插件来调试一个问题，命令行方式将会非常方便。通过将命令 “ --collect”传递给modeler进程，用户可以控制使用哪一个collection maps are used。下面这个命令将仅运行叫做build.zenoss.loc 的插件，该插件是一台设备上的接口插件。

```
$ zenmodeler run -v 10 --collect=IpInterface -d build.zenoss.loc
```

如果该命令返回有堆栈的跟踪信息，可以将这些信息、产生这些信息的命令以及Zenoss实例的版本发送给Zenoss的支持团队。

17. 使用zLinks向设备状态页添加自定义链接

用户可以在Zenoss中添加一个与设备、设备分类以及设备层次关联自定义链接。用户可以使用zLinks属性来添加自定义链接。这些链接通常都是到其它控制页面的快捷方式。添加自定义链接的步骤如下：

1. 从设备列表中选择一个想要添加自定义链接的设备。同样的，也可以在设备分类这一层次上创建一个zlink,并将其应用到该设备分类中的所有设备,而在未来被添加至该分类下的设备也将会拥有该zlink。

2.从设备的页菜单中，选择更多（More）选项，然后选择zProperties。

3. 找到zLinks属性。

4. 在zLinks的文本框中键入链接。 用户可以使用任何URL表达式，这其中包括http/telnet/file/mailto等等。用户也可以输入其它类型的http标记，比如图片链接（比如来自其它页面的性能图片）。

5. 点击保存按钮后，zLink将被保存并将出现在设备的状态标签页中。

18. 使用 XML 列表加载或卸载设备

Zenoss允许用户将本系统中的设备列表导出到一个XML文件中，然后再使用该XML文件将设备导入到另外一个Zenoss实例。如果使用命令行方式，则参考下面这个命令：

```
zendevicedump -o mydevicelist.xml
```

该命令将当前Zenoss系统中设备的名称、分类、分组、所属系统等等全部导出到一个称为mydevicelist.xml的文件。如果要想将这些设备导入到另外一个Zenoss实例，运行下面的命令：

```
zendeviceload -i mydevicelist.xml
```

Zenoss 将会尝试发现该XML文件中的每个设备。

第10章.事件监视

1. Zenoss的事件监视

Zenoss的事件管理系统能够从syslog, Windows event log, SNMP traps, 以及XML-RPC采集事件。之后,Zenoss可以基于采集的原始事件进行处理,也可以将事件整合到Zenoss模型中。特别需要指出的是，在处理过程中，每个事件都要经过一系列规则的判断，以决定事件的类别、上下联系以及重要级别，该过程可以使事件可以提供更多的信息。

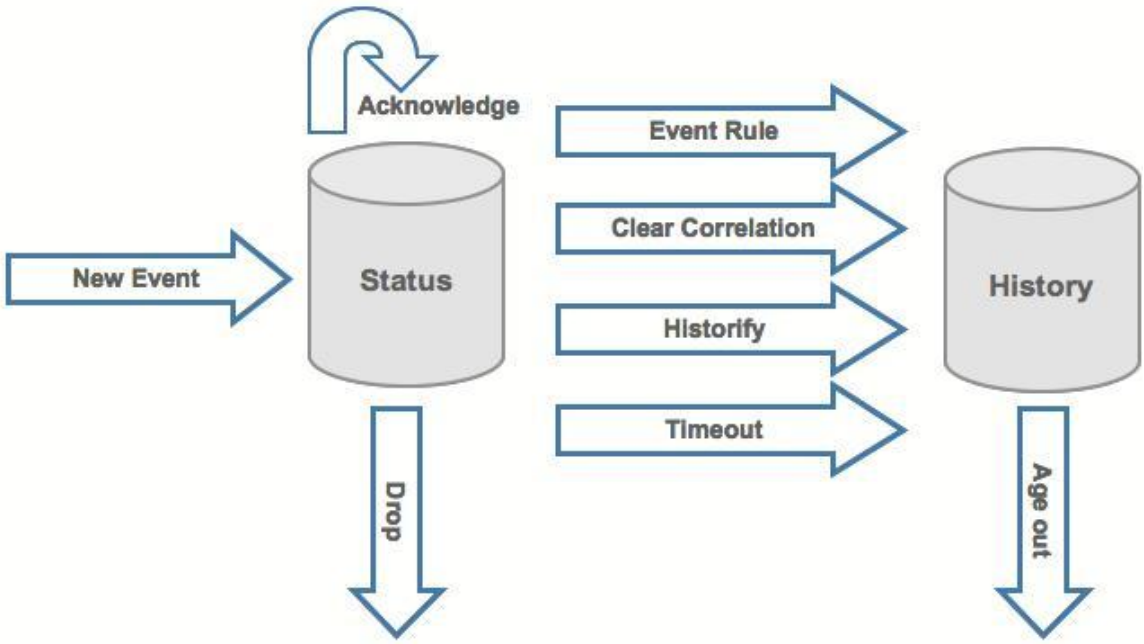
2. 事件概念

以下介绍与Zenoss及其事件监视系统有关的一些主要概念。

2.1. 事件生命周期

Zenoss 事件生命周期是一个非常直白的过程。该生命周期的第一步就是事件的产生。时间的默认状态被设置为”New”。在事件分类规则的作用下，事件可以被Acknowledged（确认），Suppressed（压制）或则“dropped”（丢弃），此后，事件可以通过四种方式被存档进入事件历史(Event History)数据库。事件可以被手工加入历史数据库，可以由于自动清除关联（坏的事件发生，同样对象的好的事件也同时发生，此时系统将坏的事件移入事件历史）而被加入历史数据库，事件规则以及非活动超时都可将事件加入事件历史数据库。

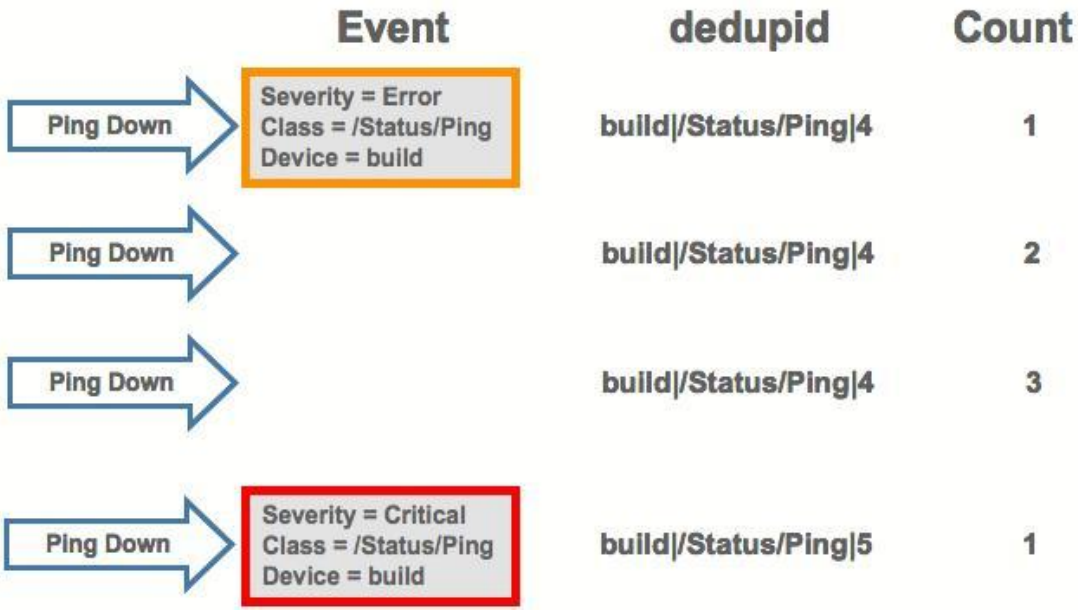
图 10.1. 事件生命周期



2.2. 事件压缩（De-duplication）

如果一个单独事件由于某种原因被提交了多次，可能是数百次，也可能是上千次，那么该事件的计数器就会大幅增加。事件的匹配是通过事件分类进行的，如果事件匹配某个分类，那么重复的事件提交将不再产生新的事件实例，而仅仅是增加事件的计数器，这样不至于在事件产生与事件被确认这段时间内产生过多的告警。

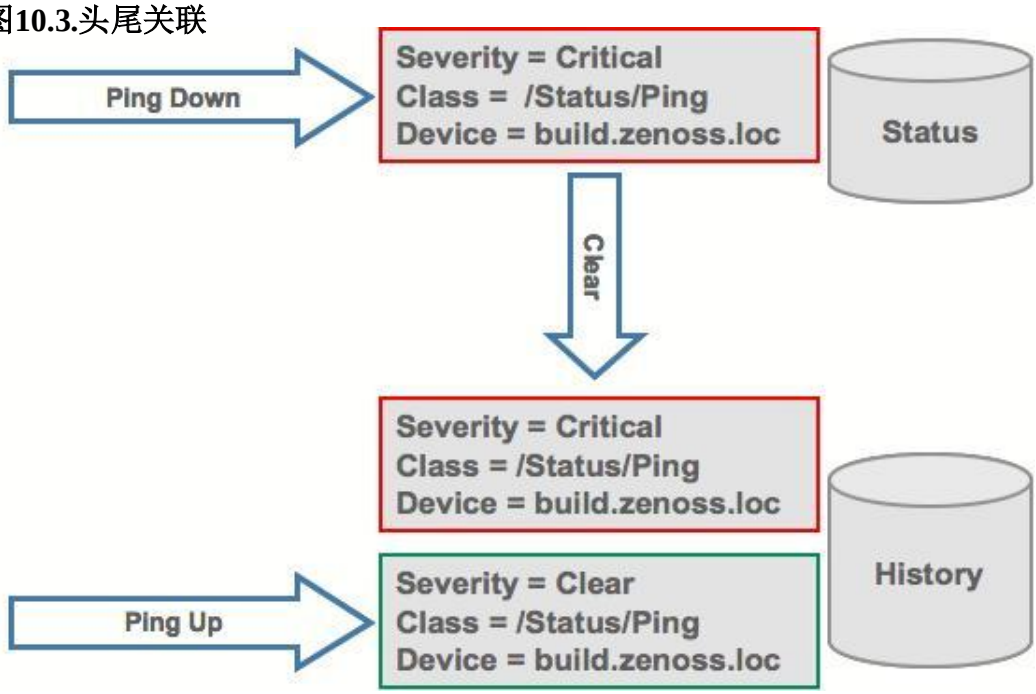
图 10.2. 事件压缩



2.3. 头尾关联

一个事件从事件的开始进入事件生命周期，从结束离开事件生命周期。如果一个与负面事件相关联的正面事件被系统接收后，则该相关的负面事件将被清除。系统通过匹配事件中的几个关键数据点来实现该功能，如下图所示：

图10.3.头尾关联



3. Zenoss 事件控制台（Event Console）

Zenoss 的事件控制台是整个系统中浏览事件的中心，事件控制台是所有进入系统的事件的仓库，用户可以在左侧导航菜单中点击 Event Console 来访问事件控制台。

图10.4. Zenoss 事件控制台

在浏览事件时，Zenoss提供集中进行排序和过滤的手段。通过点击“device”，“component”，“eventClass”，“summary”，“firstTime”，“lastTime” 和 “count”，用户可以按照这些属性对事件进行排序。同时还可以通过使用“Filter” 域，按照状态和事件等级对事件进行过滤，这里的“Filter” 是指一个正则表达式，也就是一个字符串。用户可以在事件列表中对事件进行确认、映射或者将事件移至某个特定的位置。

3.1. 查看事件明细

在事件列表中，在任何一个事件后点击后面的放大镜图标，用户就能看到事件明细。事件明细标签页如下图所示：

图 10.5. 事件明细页标签

Event: 0a84182935501326cfffc1ce

Fields	Details	Log
Field	Value	
dedupid	marc-irlandezs-computer.local||/Status/Ping||5|marc-irlandezs-computer.local ip 192.168.1.103 is down	
evid	0a84182935501326cfffc1ce	
device	marc-irlandezs-computer.local	
component		
eventClass	/Status/Ping	
eventKey		
summary	marc-irlandezs-computer.local ip 192.168.1.103 is down	
message	marc-irlandezs-computer.local ip 192.168.1.103 is down	
severity	5	
eventState	0	
eventClassKey		
eventGroup	Ping	
stateChange	2007/05/16 16:12:05.000	
firstTime	2007/05/15 17:47:24.000	
lastTime	2007/05/16 16:12:05.000	
count	1338	
prodState	1000	
suppid		
manager	tilde.zenoss.loc	
agent	ZenPing	
DeviceClass	/Discovered	
Location		
Systems	|	
DeviceGroups	|	
ipAddress	192.168.1.103	
facility	unknown	
priority	-1	
ntevid	0	
ownerid		
clearid		
DevicePriority	3	
eventClassMapping		

事件明细标签页详细记录了事件的每一个细节。

4. 生成并发送一个测试事件

用户可以使用Zenoss 手工地向事件数据库中添加一个事件，手工添加事件的方法如下：

1. 从左侧导航菜单中选择事件（Events），系统显示Events/Classes标签页。.

2. 打开事件的页菜单，选择添加事件（Add Event），系统弹出添加事件对话框，如下图所示：

图 10.6. 添加事件对话框

3. 事件对话框要求天下以下一些内容:
- Message
- Device
- Component
- Severity
- Event Class Key
- Event Class

4. 点击 OK 按钮，刚添加的事件将出现在事件控制台顶部。

4.1. 此命令行发送事件

Zenoss允许用户通过命令行工具zensendevent向系统发送事件，而脚本仅依赖于python，所以用脚本写成的命令可以方便地进行跨平台移植，zensendevent 的使用方法如下：

```
zensendevent [options] summary words
```

options:

- -d, --device

本地的完全限定性域名（The local fully-qualified domain name）

- -p, --component

发出事件的组件。

- -s, --severity

事件的等级：Critical, Error, Warn, Info, Debug, Clear. Default: Warn

- -c, --class

事件的分类

- --port

xmlrpc 服务器端口，默认为: 8081

- --server

xmlrpc 服务器，默认为：localhost

- --auth

xmlrpc server auth. Default: admin:zenoss

例子:事件监视

```
zensendevent -c /App/Fail -p sky -s Critical Onos\! very bad message\!
```

5. 事件分类

系统接收到事件后的第一步工作就是先给事件指定一个分类,在 Zenoss 系统中拥有完整的事件分类层次体系。事件通过事件分类实例被映射到事件分类，系统通过一个称为 EventClassKey 的非唯一键找到事件分类实例。

5.1. 新建事件分类

系统通过以下步骤来事件事件分类：

1. 从左侧的导航菜单中点击事件（Events）.
2. 点击分类（Classes）标签页.
3. 打开子分类（SubClasses）表菜单并选择添加分类（Add Organizer）选项，系统弹出添加分类对话框，如下图所示：

图 10.7.添加分类对话框

4. 在 ID 域中输入新的事件分类名称。.
5. 点击 OK 按钮，新的事件分类将出现在子分类列表中。

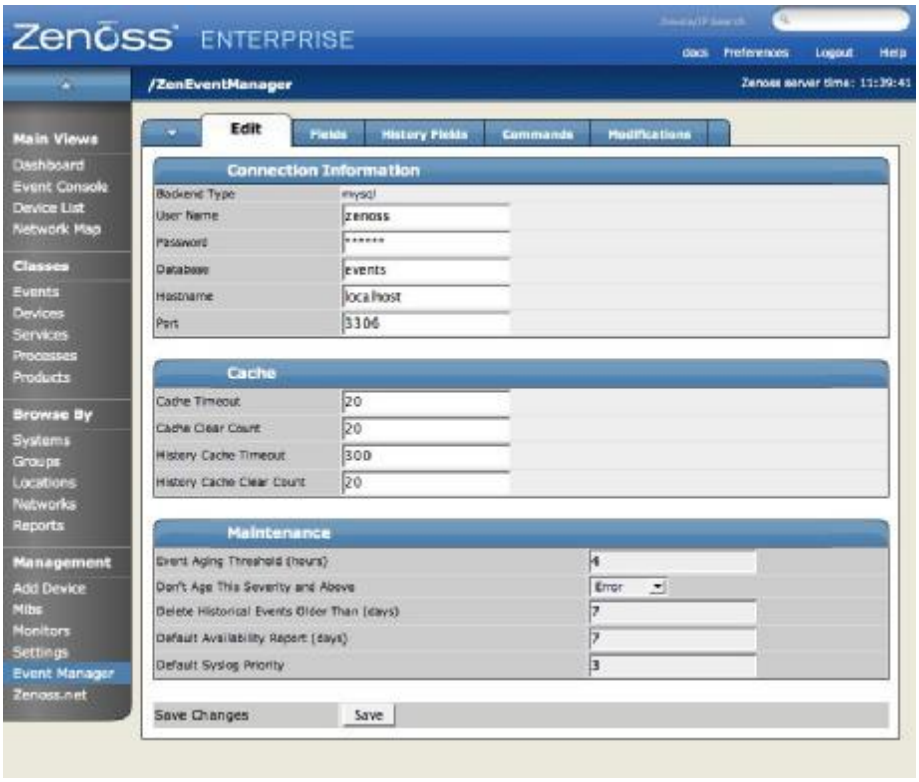
6. 事件经理（Event Manager）设置

用户可以为事件经理(Event Manager)指定特定的设置。用户可以改变的设置是到 MySQL 事件数据库的连接、事件缓存的超时时间以及计数器。

6.1. 访问事件经理设置

从左侧的导航菜单中，选择事件经理后，系统显示事件经理设置编辑标签页，如下图所示：

图 10.8. 事件经理设置编辑标签页



6.2. 更改事件数据库连接信息

导航至事件经理，在连接信息区域内修改连接信息，在此区域内可用的连接信息包括：

- Backened Type －无法编辑
- User Name －MySQL数据库的用户名
- Password － 上面用户的口令
- Database － 要使用哪个数据库
- Hostname － 主机的IP地址
- Port － 访问事件数据库时要使用的端口

6.3. 修改事件经理的缓存设置

导航至事件经理，在缓存区域内进行必要的修改，其中可供操作的选项有：

- Cache Timeout －设置事件经理的缓存超时时间，值越低，事件控制台的响应时间越短。
- Clear Cache Count － 清除缓存计数器（影响事件计数器）
- History Cache Timeout － 设置历史缓存的超时时间。数字越低，事件历史的响应时间越短。
- History Cache Clear Count － 清除事件历史的计数器。

6.4. 修改事件经理的维护设置

导航至事件经理，在维护区域内可以对事件的经理的维护设置进行修改，可选项包括：

- Event Aging Threshold (hours) （事件老化门限（小时））
- Don't Age This Severity and Above
- Default Availability Report (days) 默认可用性报告（天）
- Default Syslog Priority（默认Syslog优先级）

7. 事件确认

确认事件是一种手段，通过这种手段让系统知道，某人已经看到了这个事件，该事件正在进行处理，或者已经处理完毕。事件的确认状态将被显示在仪表盘上。要对事件进行确认，必须采取以下步骤：

1. 导航到事件控制台。
2. 在事件控制台中，选择您想要确认的事件。
3. 打开事件的页菜单，选择确认事件（Acknowledge Events）选项。此时可以看到，刚确认的事件的颜色已经从深红色变成了浅红色。
4. 回到仪表盘，我们可以看到，对事件进行确认的帐号的名称已经出现在“Acked By”字段中。

8. 将事件移入历史记录

在Zenoss系统中，我们不能直接删除事件，而是将事件首先移入事件历史记录（event history），我们将该过程称之为事件的“历史化”。更多有关事件被送入事件历史的自动化操作，我们将在事件生命周期部分进行详细的描述。要将事件移入事件历史，必须采取以下步骤：

1. 导航到事件控制台。
2. 从事件控制台中选择你想要移入事件历史的事件。
3. 到事件列表的底部点击“History”按钮。
4. 或者打开事件控制台的页菜单，选择将事件移入历史（Move Events to History）选项，系统将显示确认对话
5. 点击OK.按钮，事件将从事件控制台中消失。

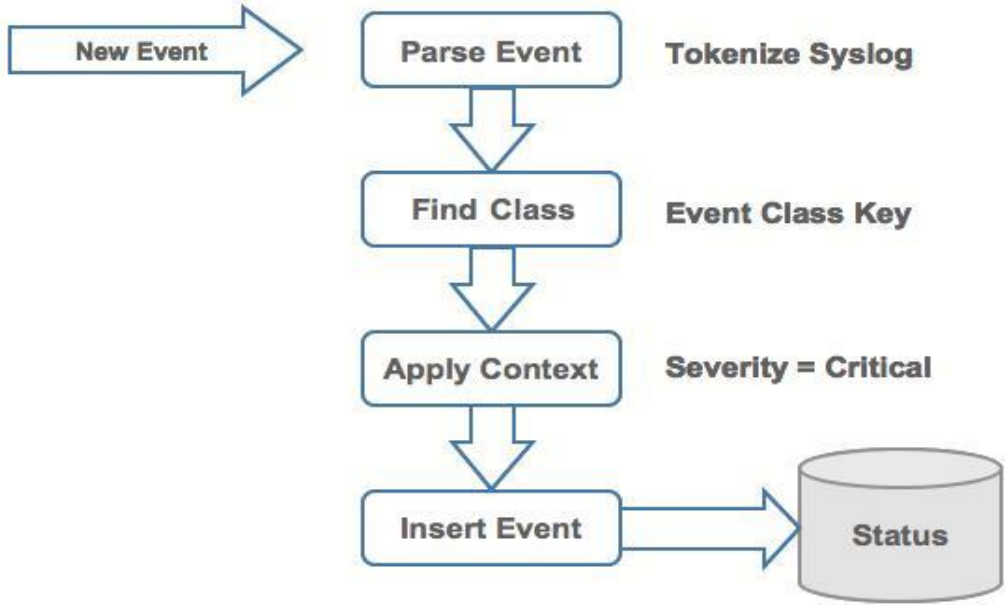
9. 清除事件历史

默认情况下，事件历史被设置为0。这意味着事件历史中的事件将永远不会老化或者被删除。您可以修改这个设置，方法是，从左侧导航菜单中选择事件经理（Event Manager）。在维护区（Maintenance area）的"Delete Historical Events Older Than (days)"域中，设置事件在从事件历史中被彻底清除之前要等待的天数。

10. 事件分类映射

事件分类映射是一种机制，通过这种机制，事件被整合进入 Zenoss 系统。下图显示了一个事件分类映射的例子：

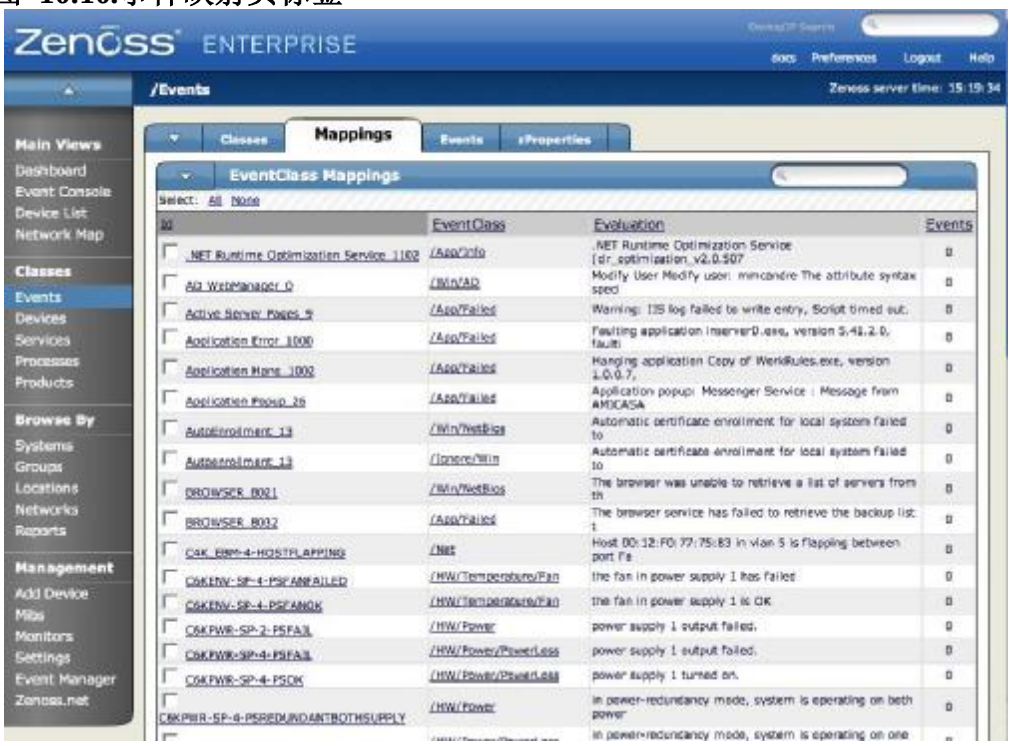
图 10.9. 事件分类映射



在这个例子中，事件经过分析后进入系统并被分配了一个相应的事件分类。之后，随着事件分类键的发现以及该键与事件的关联，该事件分类的上下文（事件等级等）将会被应用到未来进入到系统的事件。之后，事件的状态将会根据其分类进行更新。要创建或者更新事件分类映射，要采取以下步骤：

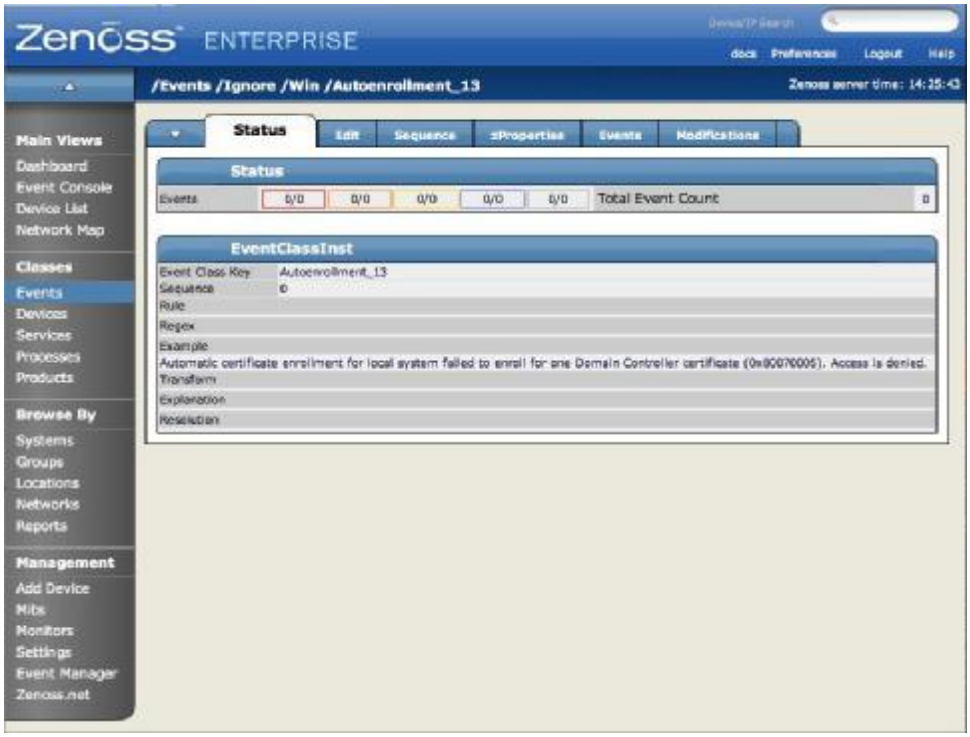
1. 从左侧的导航菜单中，选择事件（Event）
2. 点击映射（Mappings）标签页，如下图所示：

图 10.10.事件映射页标签



3. 点击您想要编辑的事件分类映射的名称，如下图所示：

图 10.11. 单独事件映射页标签



4. 点击编辑（Edit）标签页，如下图所示：

图 10.12.编辑事件映射页标签

5. 使用编辑标签页中的域进行事件映射，该标签页中的域是如下所述进行定义的：

Name - 事件分类映射的名称。

Event Class Key - 事件分类键是最初用来将到来的事件映射到事件分类的手段。对 syslog 事件，我们通常将 EventClassKey 称为标签（Tag）或者是 syslog 事件的标识符。通常情况下，EventClassKey 会映射到产生该事件的进程的名字。

由于事件分类键是非唯一的，因此，如果想要进行进一步的匹配，就必须找到一个唯一的实例。这种进一步的匹配需要通过机制(mechanisms)、正则表达式匹配或者 Python 表达式评估(Python expression evaluation）进行。因为系统中会有很多实例需要根据这些规则进行评估，因此评估的顺序将会是非常重要的。

Sequence - 如果有多个匹配，则可以使用 Sequence 域来定义顺序的优先级。“Sequence”标签页允许一个特定时间分类键的所有实例进行排序。

Rule - 输入一个 python 表达式来匹配事件。 This expression will be evaluated with the variable name “evt” bound to the current event. A detailed list of fields can be found in the Event Database Dictionary Appendix of the Admin Guide. 规则使用这些域的例子如下所示：

```
evt.priority>4
```

Regex - 用户可以在该域中输入正则表达式来匹配事件。在对事件进行正则表达式匹配时，可以使用提取指令(Extraction directives)来,这些指令遵循Python指令格式(?P<keyName>\S+)。

example - 当创建一个正则表达式时,原本的事件文本将会被添加到example域中,保存之后,该正则表达式将针对这些文本进行评估。对正则表达式来说，这是个很好的调试工具。

Transform - 一个或者多个python声明。这使得用户可以通过操作EVT变量来修改事件。本章中使TALES表达式, 有关如何使用这些表达式的详细信息，参见本文档的TALES 表达式章节。

Explanation - 输入一个文本来描述这些事件分配映射的匹配。与Resolution 域一起使用。

Resolution - 使用该域来输入用于清除事件的resolution（解析）指令。

如果EventClassKey的搜寻过程没有结果，系统将会使用“defaultmapping”键再次进行搜寻。通过正则表达式，系统可以使用默认映射去匹配绝大范围的事件。

6. 在完成所有的修改之后,点击保存按钮。

11. 使用事件 zProperties 来应用事件和设备上下文

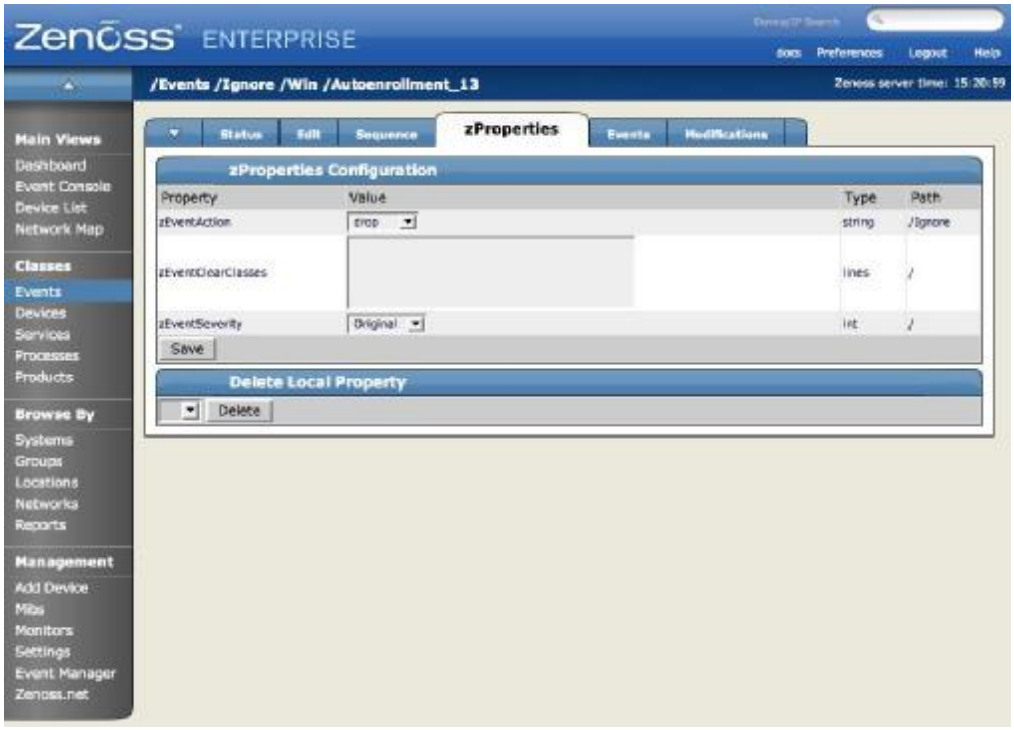
一旦事件被映射到了一个分类，将会有两件事情发生： (1)其事件上下文将被应用。(2)其设备上下文将被应用。事件分类上下文应用是通过查找zProperty列表zEventProperties进行的。任何在zEventProperties找到的属性名称都将以设置其它zProperties的同样的方式进行，但一种情况例外，那就是在查找属性时，'zEvent_'前缀被添加到了属性名的前边。zEventProperties的值改变时，Zenoss将会为列表中的每个属性创建一个placeholder。

一个常见的将会改变的事件属性是'Severity'，而且事实上，该属性默认被添加到zEventProperties。如果要改变一个已分类事件的事件等级，需要在事件分类路径中改变zEvent_severity的值。

在事件上下文被应用后，设备上下文将被应用。在该过程中，productionState,location, DeviceClass, DeviceGroups, Systems 属性都将被附加到该事件上。一旦这些属性与事件关联之后，Zenoss将会尝试着去更新zEventProperties(如上文所述)，但会使用设备分类路径而不是事件分类路径。这将允许某一个设备或者某类设备覆写（override）任何给定事件的默认值。要编辑事件的zProperties,必须采取以下步骤：

1. 从任何事件分类映射的主窗口中，选择您希望改变的事件分类的zProperties标签页，如下图所示：

图 10.13. 事件映射zProperties



2.使用如下属性来定义事件的 zproperties：

zEventAction - 该属性告诉系统当有一个该分类的匹配时应该做什么。可选项有 Drop,Status,以及 History。

zEventClearClasses - allows you to enter matches that send clear events when they appear on this particular device.

zEventSeverity -给出该事件分类的默认级别。

12. 通过用户界面映射事件

事件进入系统后，某些情况下，Zenoss 在之前并未见过这种事件，因此也不知道如何对这些事件进行分类。此时，作为系统管理员就必须通过用户界面对事件进行分类并创建事件映射，这样以后，系统每次都将按照同样的方式处理此类事件。用户可以通过如下步骤来通过用户界面对事件进行映射：

1.导航至事件控制台。

2. 选择想要映射的事件。

3.打开事件控制台的页菜单，选择映射事件到分类（Map Events to Class）选项。系统弹出事件映射对话框，如下图所示：

图 10.14. 映射事件对话框

4. 选择您想要该事件映射的事件分类。

5. 点击OK按钮，该事件的分类以及具有同样参数的事件将执行该分类映射。

13. 使用映射对事件进行关联

您可以使用Zenoss创建多个事件分类映射，这可以使系统对两个事件进行关联，同时，如果两个事件出现的话，系统就可以创建一个动作。举例来说，我们可以在一个失败的开始（一个事件）和失败的结束（一个事件）之间创建关联，这样一来，清除事件就可以被自动移入事件历史。

1. 导航到一个您希望进行关联的时间分类，点击映射标签页，创建一个新的映射。

2. 再次点击事件分类标签页，然后点击刚刚创建的映射，之后点击该映射的编辑标签页。

3. 将事件分类键重新设回到您想要关联的事件分类。

4. 对事件映射域(Event Mapping fields)进行必要的修改，比如在Regex域中添加文本“totally broken”。

5. 使用一个与第一次映射相关的名字进行二次映射。

6. 将事件分类键设置成您刚刚为上面的事件映射设置的同样的事件分类键。

7. 在事件映射的各个域中进行必要的修改,比如在Regex域中添加字符串“everything is ok”。

8. 到zProperties标签页并设置要清除的事件等级zEventSeverity属性(万一您希望在哪里第二个事件产生一个清除关联)。当这个映射匹配时，事件将会变成一个清除事件，系统将会清除所有同类的活动事件。

9. 上述工作完成后，到“Sequence” 标签页，我们可以看到同样事件分类键的三个映射都列在这里

10. 该顺序用来控制映射如何被应用，第一个映射将被用于给定事件分类键的所有事件，并会组织我们新的规则激活。

11. 将该映射的顺序号改为3，把该映射放在顺序的最后，然后点击保存按钮。

14. 事件命令

事件命令允许用户定义shell命令，这些shell命令在事件被zenoss系统接收并满足事件命令定义的条件时执行。事件命令通过zenactions进程执行。在下文中，我们将会创建一个事件命令，该事件命令使用我们在上一章节中创建过的映射。我们创建的这个事件命令将会从我们的事件中把文本写入到一个日志文件中。

14.1. 创建一个事件命令的例子

下面这个例子将给出创建一个事件命令的框架。.

1. 点击事件经理（Event Manager）并到命令（Commands）页。
2. 创建一个新的事件命令，称之为写文件（writeFile）。
3. 按照以下设置编辑该命令：

```
Set Enabled = True
Default Command timeout = 60
Delay = 0
```

4. 对命令进行如下设置：：

```
echo "${evt/evid} ${dev/id} ${dev/manageIp} ${evt/message}" >> /tmp/cmdoutput
```

5. 设置清除（Clear）命令

```
echo "${evt/evid} ${evt/clearid} ${dev/id} ${evt/message}"" >> /tmp/cmdoutput
```

6. 在 Where 区内，添加一个消息过滤条件"testing event commands"。
7. 点击保存（Save）按钮。

14.2. 测试事件命令

Add Event 页面是一个用来测试事件命令的非常好的工具。我们可以采取以下几个步骤来创建一个事件，该事件满足您刚刚创建的事件命令的条件：

1. 查看命令的输出文件：

```
tail -f /tmp/cmdoutput
```

2. 使用添加事件功能来人工创建一个 down 事件，并对其进行如下定义：

```
Message = My Application is totally broken
Device = <系统中的任何设备>
Event Class Key =
Severity = Critical
```

3. 查看输出文件，在下一个zenactions周期后，您将会看到该down事件。
4. 现在，我们可以通过发送一个up事件来清除该down事件，事件定义如下所示：.

```
Message = Now everything is ok
Device = <上文刚使用过的设备>
Event Class Key =
Severity = Info
```

15. SNMP 陷阱与事件转换

15.1. SNMP 陷阱映射

如果一个 SNMP 陷阱到来并被划分为一个"/Unknown" 事件类型，这主要是因为 Zenoss 并不知道您想要怎么处理该事件。如果想要映射这个事件，我们可以到事件控制台中，选中该事件，

从页菜单中，选择映射事件到分类（Map Events to Class）选项。此时系统将显示事件映射对话框。在对话框中，每个分类将对事件进行不同的操作：改变事件的等级、将事件移入事件历史等等。现在，我们选择"/App"并按下映射（Map）按钮。

如果要对该映射进行编辑，我们可以从左侧的导航菜单中，选择事件（Events），然后点击映射(Mappings)标签页。搜寻并找到您的事件映射，点击事件映射的名称，然后点击编辑(Edit)标签页。这将会把您带到一个事件“映射”的编辑屏幕。这些“映射”是一些规则，这些规则用于将该事件映射到"/App"分类。这些规则通过一个特定的 OID 映射陷阱，因此正是您所全部需要的。

在“映射”的转换章节中，您可以添加一些代码来修改小结。举例来说，您可能希望将小结字符串设置成"spam Filter Detects Virus"，那么您需要将下述内容添加到转换编辑区：

```
evt.summary = "spam Filter Detects Virus"
```

SNMP 陷阱的头部有一些标准的信息，这些信息的绝大部分都是没用的，但有一个顺序（sequence）属性/值需要注意。您已经表明，您希望使用 OID".1.3.6.1.4.1.9789.1500.2.5"的值作为事件的小结，如果您在系统中加载了 MIB，您就可以这样做：

```
evt.summary = evt.spamFilterDetectsVirus
```

但是... 我们有 OID 而且数据仍在那里。因此我们可以直接使用 OID，如下所示：

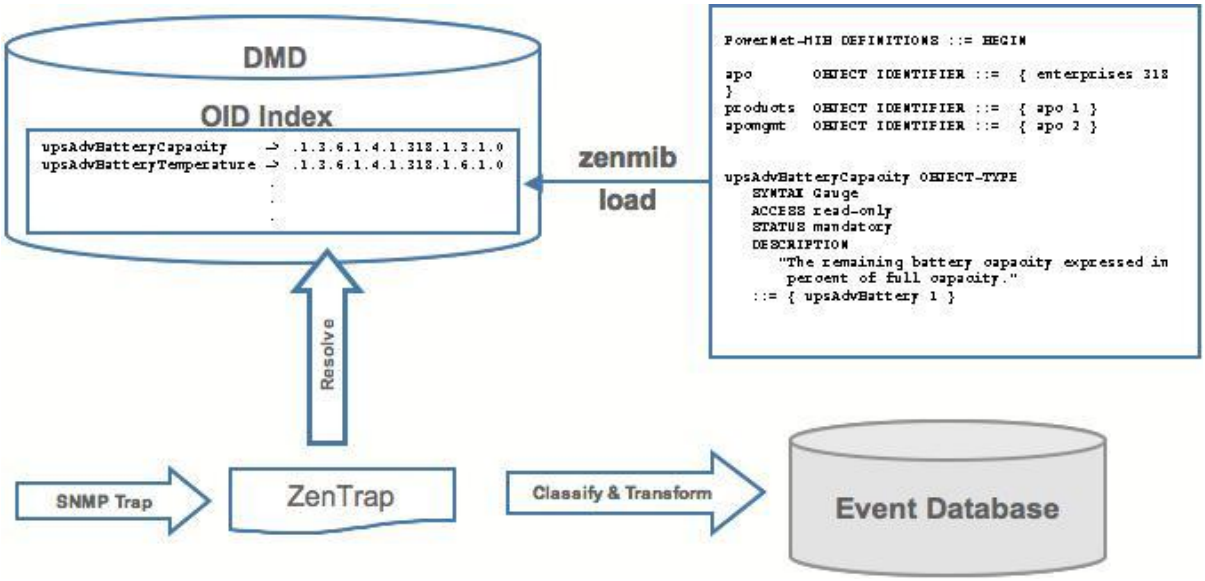
```
evt.summary = getattr(evt, ".1.3.6.1.4.9789.1500.2.5", "Unexpected missing OID")
```

事件的 "device" 对象就也变得可用了：

```
evt.summary = getattr(evt, ".1.3.6.1.4.9789.1500.2.5", "Unexpected missing OID") + " from device " + device
```

Zenoss 使用 MIBs 来翻译 SNMP 陷阱（SNMP 陷阱包含原始的 OID 值）。向 Zenoss 中加载 MIB 能够使 Zenoss 将数字化的 OID，比如.1.3.6.1.2.1.1.6 转换为描述性文字，比如“sysLocation”。同时还是的事件映射变得更容易操作。

图 10.15. SNMP 陷阱转换



以下是一个小的展示用途的 MIB。

```
NOTIFICATION-TEST-MIB DEFINITIONS ::= BEGIN
IMPORTS
ucdavis FROM UCD-SNMP-MIB
NOTIFICATION-TYPE FROM SNMPv2-SMI
;
demonotifs OBJECT IDENTIFIER ::= { ucdavis 991 }
demo-notif NOTIFICATION-TYPE
OBJECTS { sysLocation }
STATUS current
DESCRIPTION "Just a test notification"
::= { demonotifs 17 }
END
```

15.2. 发送测试陷阱

下面我们介绍如何发送一个SNMP陷阱。

1. 从命令行中输入如下命令：

```
$ snmptrap -v 2c -c public localhost '' 1.3.6.1.4.1.2021.991.1.3.6.1.2.1.1.6 s "Device in Annapolis"
```

2. 将该用于展示目的的MIB保存到一个文件。

3. 发送陷阱。

4.打开事件控制台，可以看到刚刚发送的陷阱。

5.在事件控制台的右侧，点击刚刚发送事件的放大镜图标。

6. 点击事件明细（Details）标签页，您将会看到以下内容：

```
.1.3.6.1.2.1.1.6 Device in Annapolis
```

7. 将该事件移入事件历史。.

您现在可以向Zenoss中加载一些MIB库了，这样我们就能够将这些OID翻译成更容易理解的格式。

1. 将示例用途的MIB拷贝到$ZENHOME/share/mibs/site。

2. 运行zenmib 将其加载到Zenoss系统中。

3. MIB载入后，仍缺少一些其它的定义。这些定义恰巧在系统的某处，此时您可以将其拷贝入Zenoss。

4. 然后再次运行zenmib将其载入Zenoss系统中:

```
$ zenmib run -v 10
```

5. 再次发送一个陷阱。

现在，检查一下时间，确认计数器为1.如果计数器是2，就将该事件移入事件历史并再次发

送陷阱。查看事件的事件明细标签页，此时您将看到如下内容：

```
sysLocation Device in Annapolis
```

您还可以看到事件小结已经从

```
snmp trap 1.3.6.1.4.1.2021.13.991 from localhost
```

变成了：

```
snmp trap ucdExperimental from localhost
```

尽管这是个改进，但我们仍希望“sysLocation”的值“Device in Annapolis”直接显示在事件小结中，这样用户就不必再到事件明细中去查看了。

15.3. 使用事件映射进行事件转换

如果要在事件到来之时进行修改，我们就必须要通过用户界面创建一个事件映射，也就是说，要创建一个新的事件分类。

到事件控制台中，从现有的事件中以该分类创建一个事件映射。到编辑标签页。我们可以过滤那些映射针对的事件。举例来说，您可以过滤包含有文本ucdExperimental的消息，只要在Regex域中输入字符串“ucdExperimental”即可。

在事件转换过程中，您可以使用更为详细的数据来更新事件。Zenoss允许使用Python脚本来对事件进行更新。在下面这个例子中，我们可以使用设备的sysLocation属性充当事件的小结：

```
evt.summary = evt.sysLocation
```

保存该事件映射。将所有的事件移入事件历史。重新发送陷阱。 陷阱小结现在将会在您刚刚指定的位置中读取设备的名称。如果您对转换有任何问题，就检查一下zentrap.log文件，看一下有什么样的错误发生。

15.4. 基于事件分类的事件转换

当事件进入到Zenoss系统后，事件的值（比如事件的等级）可能会被改变。举例来说，事件小结可能会变得更丰富，而事件的等级可能会根据事件小结内容中的部分文本发生改变。

每个事件分类都允许在事件到来的时候执行一小段Python脚本。举例来说，一个用户可能需要文件系统满的告警门限事件是critical等级，而下面这段python脚本代码就实现了这个功能：

```
if evt.component == '/data' and evt.severity != 0:evt.severity == 5
```

就像事件分类键的事件映射一样，我们可以在使用脚本转换事件分类的过程中使用"evt"和"device"对象。

16. 定制事件视图

在Zenoss系统中，用户可以创建并编辑自定义事件视图，这样，用户就可以保存自己已经过滤的事件列表的视图，这种自定义视图是以单个用户为基础的。定制事件视图的步骤如下：

1. 点击仪表盘右上侧的 Preferences 链接。

2. 点击事件视图标签页，如下图10.16所示：

3. 打开事件视图的表菜单，选择添加事件视图（Add Event View）选项，系统将弹出添加事件视图对话框，如下图10.17所示：

图 10.16.定制事件视图-初始视图

图 10.17. 添加定制事件视图

4. 在 ID文本框中，输入客户自定义视图的名称。

5. 点击 OK按钮，该定制视图将出现在列表中，注意，该事件视图中有一个定制告警彩虹。

6. 点击您刚创建的新视图的链接,注意列表的尺寸和条目的数量。点击编辑标签页,如下图10.18所示。

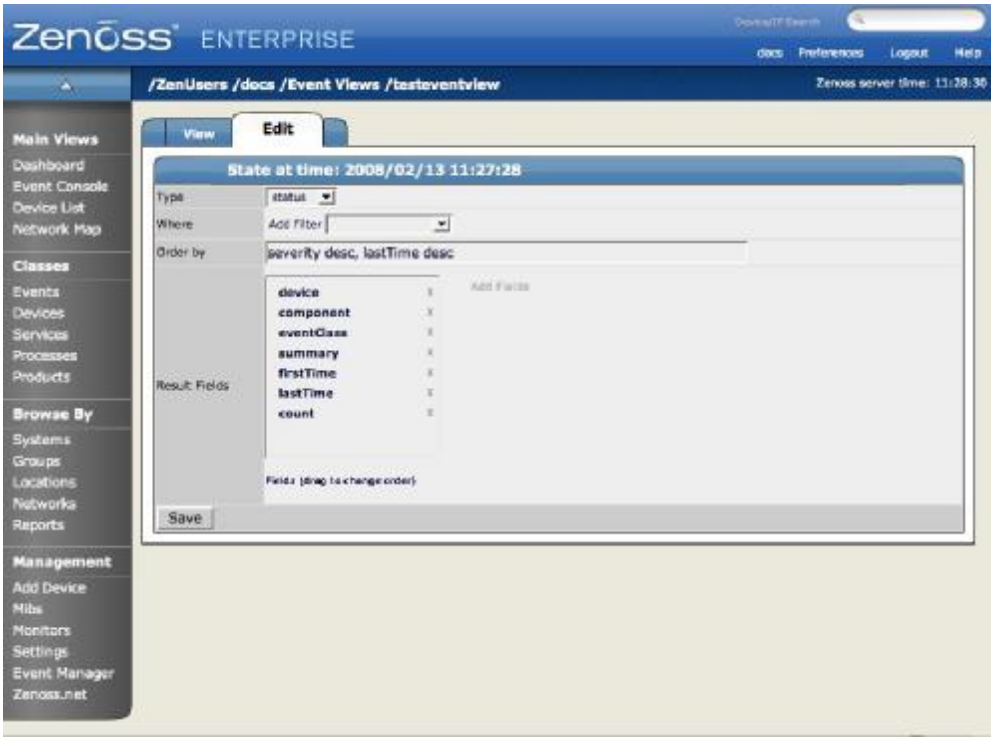
8. 为该事件视图添加条件：

Type - 如果您想要显示活动的事件或者事件历史，您可以在这里进行选择。
Where - 使用该选项来添加过滤条件 (类似于告警规则的where语法)
Order by - 该选项决定视图中事件的排序顺序。
Result Fields：定义在视图中显示哪些字段。可以点击字段后面的X按钮来将该字段从结果视图中移除。

9. 点击保存（Save）按钮。

10.点击视图（View）标签页查看定制视图的结果。

图 10.18.定制事件视图 – 编辑页标签



17. 使用ZenMail和ZenPop将邮件消息转换为Zenoss事件

ZenMail 和 ZenPop 使网络管理员能够将邮件消息转换为Zenoss中的事件。.该功能在嵌入式系统（WAPs, NAS设备, RAID 控制器）为事件进行邮件中继时将会非常有用。

17.1. ZenMail

ZenMail起SMTP服务器的作用。您可以配置您的嵌入式系统，使其使用Zenoss服务器的IP地址作为邮件中继服务器。相应的，ZenPop将允许您从一个POP服务器读取事件邮件。

ZenMail 支持以下配置指令：

${ZENHOME}/bin/zenmail (无参数)：默认选项。绑定到端口25来监听即将到来的邮件消息。忽略邮件中的TO选项，并使用From地址作为设备的IP地址。

${ZENHOME}/bin/zenmail -监听端口: 适用于一个SMTP服务器已经运行在Zenoss系统中，且用户不希望干预现有的邮件传递系统的情况下。如果是没有参数，则使用From地址作为设备的IP地址。

17.2. ZenPop

ZenPOP 支持以下一些配置指令：

--usessl:向POP服务器发送STARTTLS命令，并试图使用SSL传输邮件消息。如果从Google读取邮件，该选项是必须的。

--nodelete: 在读取所有的消息后不发送 DELE 命令。该指令通常用于最初的测试，这样您就不必再次向POP帐号重新发送测试消息了。要注意的是，一些邮件系统（比如Google）在发送DELE后并不真正地删除邮件。

--pophost: POP服务器的主机名或IP地址。

--popport: POP服务器的TCP监听端口。默认为110.在POP提供者监听另外一个端口时（Google监听995）使用。

--popuser: 邮件帐号的用户名。

--poppass: 邮件帐户的口令。

--cycletime: 轮询的周期。ZenPOP在每次读取所有邮件之前睡眠的时间。

17.3. Zenoss如何将不同的消息元素翻译成事件

17.3.1. Zenoss如何使用 FROM 域

如果FROM域是一个IP地址，Zenoss将会把事件与具有相同IP地址的设备关联起来。如果FROM域是一个FQDN名称，Zenoss将会将其解析成一个IP地址，并将事件与具有该IP地址的设备关联起来。主机名的解析使用A记录或者MX记录。

17.3.2. How Zenoss uses the TO Field

Zenoss完全忽略邮件消息中的TO域，ZenMail承认发送至任何用户和域名的邮件名称的组合。ZenPOP也完全忽略邮件中的TO域，而仅仅关注邮件的FROM域。

17.3.3. How Zenoss uses the SUBJECT Field

ZenMail和ZenPOP使用邮件的SUBJECT域作为事件的小结。

17.3.4. How Zenoss uses the Message Body

ZenMail和ZenPOP使用第一个MIME附件作为事件的明细。第二个消息体（通常是HTML编码的消息）及其它附件（通常是文件）会被忽略 。

第11章. 可用性监视

1. 使用ZenPing监视拓扑

Zenoss 的可用性监视功能能够对IT基础设施进行主动测试。Zenoss的可用性监视手段包括Zenping、 Zenoss的三层拓扑监视进程和Zenstatus,, 这是一个 TCP状态测试进程。.

ZenPing 是自动进行配置。您可以使用About page、Status tab 来停止或者启动Zenping。ZenPing对ICMP的状态进行高性能异步测试。该进程的最重要之处在于，Zenoss已经为您的路由系统建立了一个完整的模型。如果Zenoss的路由模型有缺陷的话，ZenPing的拓扑坚实能力将不可用。如果Zenoss的路由模型有缺陷，那么我们可以在zenping.log文件中找到症结所在。

Zenmodeler进程负责发现zenoss网络中到每台设备的路由。Zenoss将会试着不使用Internet路由表，而是依靠Zenmodeler进程来发现设备之间的关系并创建自己的网络拓扑图。

通常来讲，如果有任何一个已知路由中断，那么该路由中断将仅产生一个ping事件。除此之外的任何额外的路由中断将仅会在系统中的该路由设备上打上标签。Zenmodeler进程可以在zenoss系统中运行，也可以在远程主机上运行。如果路由器不共享他们的路由表或者接口信息，该监视模型将会中断。

1.1. 控制Ping周期

1. 从左侧的导航菜单中，选择Monitors（监视器），将Status monitor设置为localhost 并点击localhost链接。

2. 注意目前正被该Monitor 执行Ping操作的设备列表。

3. 在编辑（Edit）标签页中，将周期间隔（Cycle Interval）修改为想要的周期间隔时间。

4. 在下一个配置周期中，ping monitor将会按照刚刚设定的周期来执行Ping操作。

1.2. 使用预定义的 /Ping 设备分类

/Ping设备分类是系统中设备配置的一个例子，该例子中的设备将仅进行可用性监视。Zenoss也不会为位于该分类下的设备采集性能数据。在您进行自己的配置时，您可以将该分类作为您配置的一个参考，或者您刚好有一个设备仅需要进行可用性监视，您可以把该设备放在该分类下。

2. 监视TCP服务

用户可以使用服务（Service）菜单来管理和监视运行在网络中的服务。要访问服务页，用户可以从左侧的导航菜单中选择服务(Service)，之后系统将显示服务概览(Services Overview)页，该页面显示了两个服务文件夹，也就是两类服务，同时还显示了已经被添加到系统监视器下的服务。

2.1. Zenstatus

ZenStatus用于监视TCP服务。 方法是，到导航菜单条下的”Service”根下找到该服务，并打开其监视。服务监视可以在服务分类的基础上全部打开，但服务分类下的具体的服务实例可以对该设置进行覆写（override）。举例来说，默认情况下SMTP服务将被监视，但该服务并非所有主机上的重点服务，因此我们可以在某个设备上把该服务的监视给去掉。同样的，如果一个服务被配置成仅在localhost上进行监听，那么zenoss也不会监视该TCP服务。

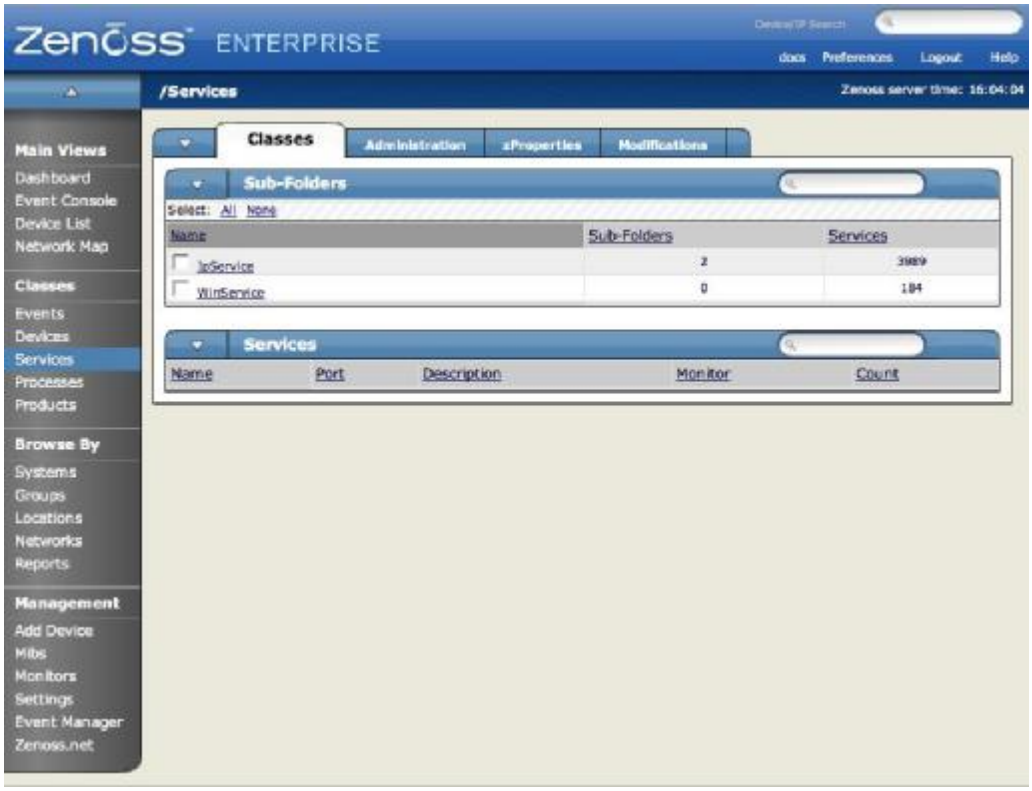
2.2. 向监视器（Monitor）添加一个服务

要向监视器添加一个服务，必须采取如下步骤：

1. 从服务概览（Services Overview）页 中，找到Services text域，输入您希望监视的服务的名称。

2. 点击添加（Add）按钮，该服务将出现在服务列表中。

图 11.1.服务列表 – 分类标签页



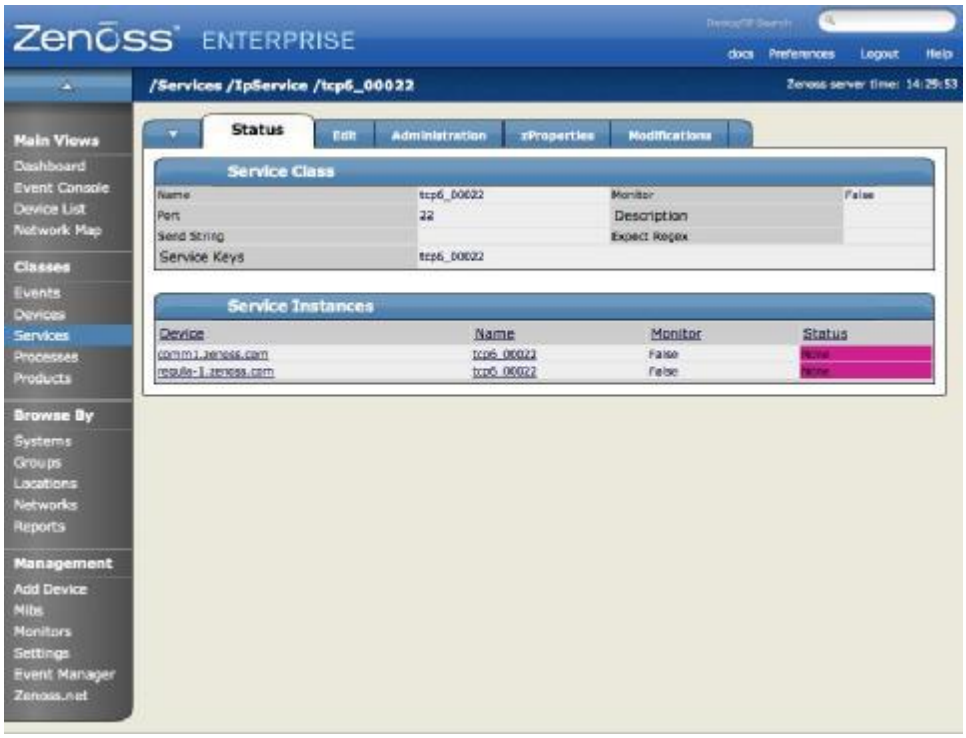
3. 将监视（monitoring）设置为True。点击编辑（ Edit）标签页，将Monitor设置为True，该服务将处于监视状态之下。

2.3. 监视状态服务状态信息

要查看一个给定服务的状态信息，可以从服务概览页的服务列表中先选中该服务，系统将显示单个服务状态标签页，如下图 11.2 所示：

该标签页显示了该服务的小结信息，您可以在该页中看到服务的名称、是否被监视、描述、任何相关的服务键（Service Keys）以及一个设备列表。要改变该标签页上的任何信息，需点击编辑（Edit）标签页。

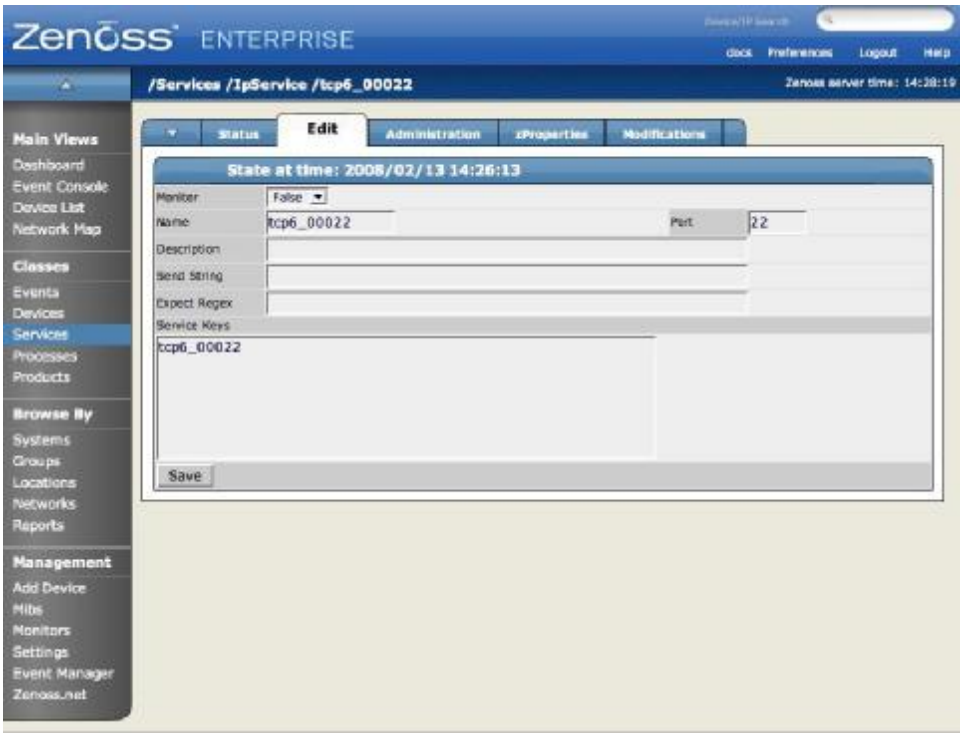
图 11.2.单个服务状态页标签



2.4. 编辑服务信息

要编辑显示在单个服务状态标签页上的信息，可以在单个服务状态标签页中点击编辑（Edit）标签，如下图所示：

图 11.3. 单个服务-编辑页标签

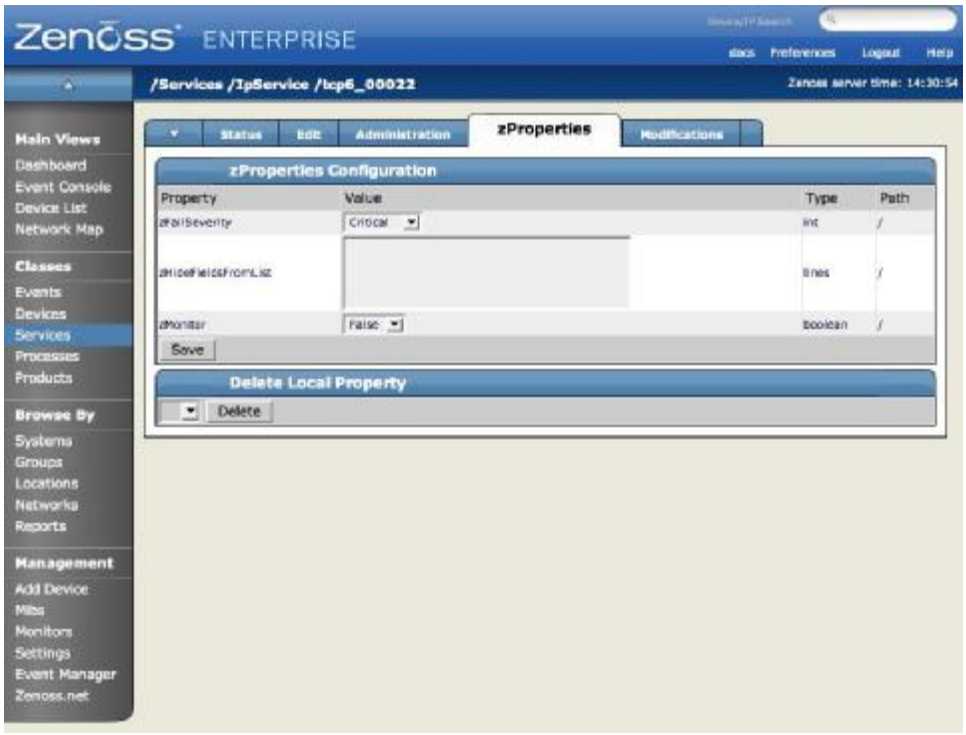


在该页中，设置该服务的Monitor 为 True 以便对该服务进行监视，您还可以添加任何有关的服务键（Service Keys）或者为该服务添加一段简短的描述。

2.5. 配置服务的 zProperties

您可以同时为所有的服务配置zProperties，也可以单独为一个服务配置zProperties ，还可以为服务层次树中的任何服务配置zProperties。如果要为所有的服务配置zProperties，可以从服务概览页中进行配置。如果要为一个服务进行单独配置，就必须在服务概览页中找到该服务，然后点击该服务的名称，之后点击该服务的zProperties标签页，如下图所示：

图 11.4. 单个服务的 zProperties标签页



您可以在该标签页中为一个服务配置以下的zProperties：

- ZFailSeverity
- ZHideFieldsFromList
- zMonitor

更过有关服务的zProperties，请参阅zProperties附录。

2.6. 使用预定义的/Server/Scan设备分类

/Server/Scan 设备分类是系统中的一个配置举例，该配置用于通过端口扫描手段来监视设备的TCP服务。在您进行自己的配置时，您可以将该设备分类作为您配置的一个参考，或者您正好有一个设备仅需要进行服务可用性监视，那么您可以直接将该设备放到该分类下。

2.7. 使用服务类来监视一个服务

本章将具体描述如何通过使用一个服务分类来监视一组设备的一个IP服务。
导航至一个设备的OS标签页，如下所示：

图 11.5.显示受监视的进程

在IP Services区内，点击您希望监视的服务的链接，系统显示该服务的小结，如下图所示：

图 11.6. 服务小结

3．在该界面中，将Monitored 标签设置为True来监视该设备上的这个服务。

4．要在整个系统范围内监视一个服务，点击服务分类链接，该页面将显示系统中所有正在运行的服务（无论该服务是否被监视）。

5. 点击编辑（Edit）标签页并将Monitored设置为True。该操作将使得系统中该服务的所有实例处于监视之下。

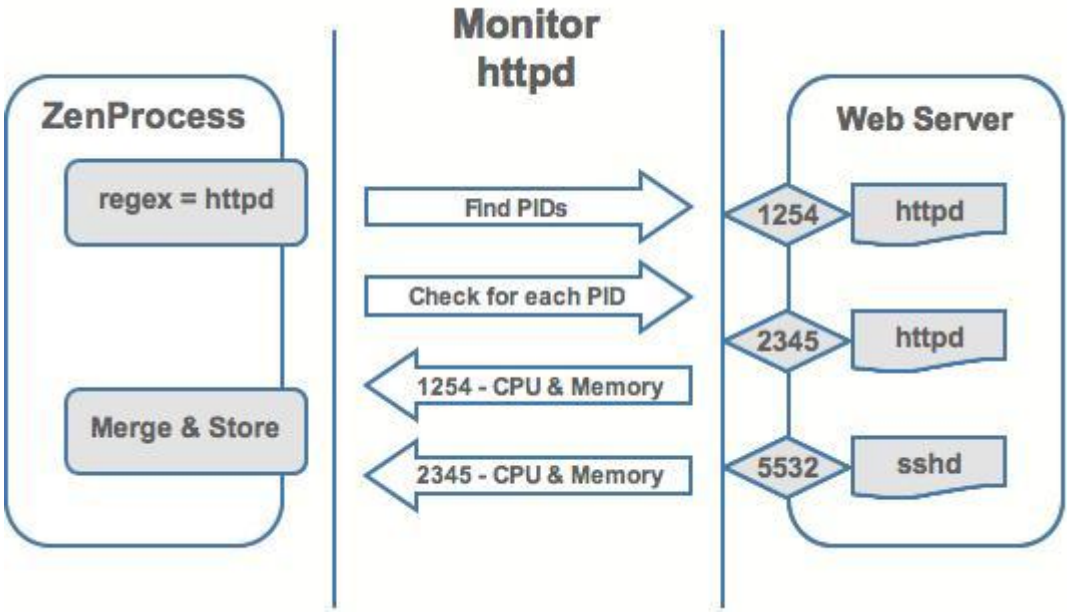
6. 点击保存（ Save）按钮。

7. 再次点击状态（Status）标签页。绝大多数服务的实例将会被设置成绿色，也就是说这些服务目前正处于监视之下并且运行正常。

3. 监视进程

Zenoss 能够监视任何运行于您网络内的进程。您可以对zenoss进行配置并使其对整个系统内的进程进行监视。Zenoss进程监视的原理如下图所示：

图 11.7.进程监视



您可以看到，ZenProcess进程使用一个正则表达式进行匹配，这样做的目的是，检查一下看是否能够发现与该正则表达式匹配的PID，也就是说看一下设备上是否有与正则表达式字符串匹配的进程。

3.1. 向监视器添加一个进程

要向进程监视器添加一个进程，需要采取如下步骤：

1. 从左侧的导航菜单中，选择进程（ Processes）。系统显示进程页面，如下图所示：

图 11.8.进程页标签

2．从进程页的表菜单中，选择添加进程（Add Process），系统显示添加进程对话框，如下图所示：

图 11.9. 添加操作系统进程对话框

3．在ID域中，输入与进程进行匹配的正则表达式，然后点击OK按钮。该进程被添加且进程窗口重新出现，该窗口显示您刚刚输入的进程。

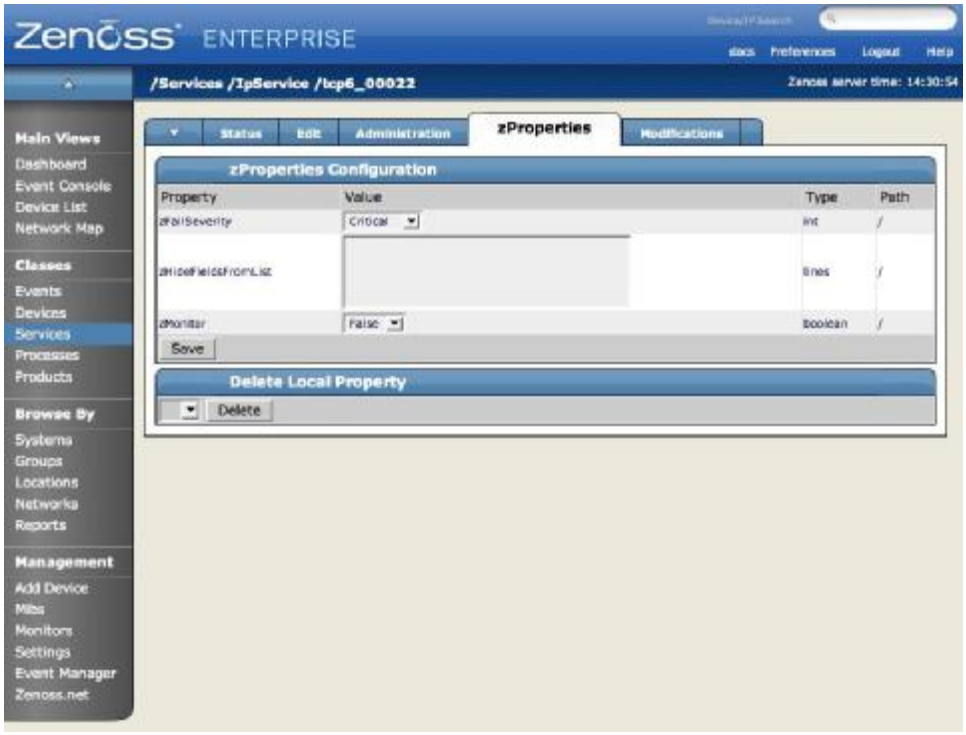
现在，您可以对该进程进行监视了。在重新建模（remodel）之后，系统将显示每台运行有该进程的设备，也就是说，zenoss现在可以对系统中所有的此类进程进行监视了。注意，重新建模需要手工执行，或者系统会每隔6小时自动执行。

在进程概览页中点击某一个特定的进程，系统将显示所有在系统中运行的该进程的实例。如果有多个实例的话，zenoss将会监视所有这些进程实例占用的CPU和内存的利用率。如果该进程仅有一个实例，系统将会以图形方式显示CPU和内存的利用率。如果要对设备上的进程进行监视，设备上的SNMP代理必须要有一个合理的HOST-RESOURCES mib。

3.2. 配置进程的 zProperties

您可以为所有的进程配置zProperties，或者为单独的一个进程配置zProperties，同时您还可以为进程层次树中的任何进程配置zProperties。要为所有的进程配置zProperties，可以从进程概览（Process Overview）页中点击zProperties标签页。如果要为单独的一个进程配置zProperties，可以在进程概览页中点击进程的名字，然后点击该进程的zProperties标签页，之后系统将显示进程的zProperties标签页，如下图所示：

图 11.10. 进程的 zProperties 标签页



您可以从该标签页中为所有进程或者是所选进程设置如下的Properties：

- ZAlertOnRestart
- ZCountProcs
- ZFailSeverity
- zMonitor

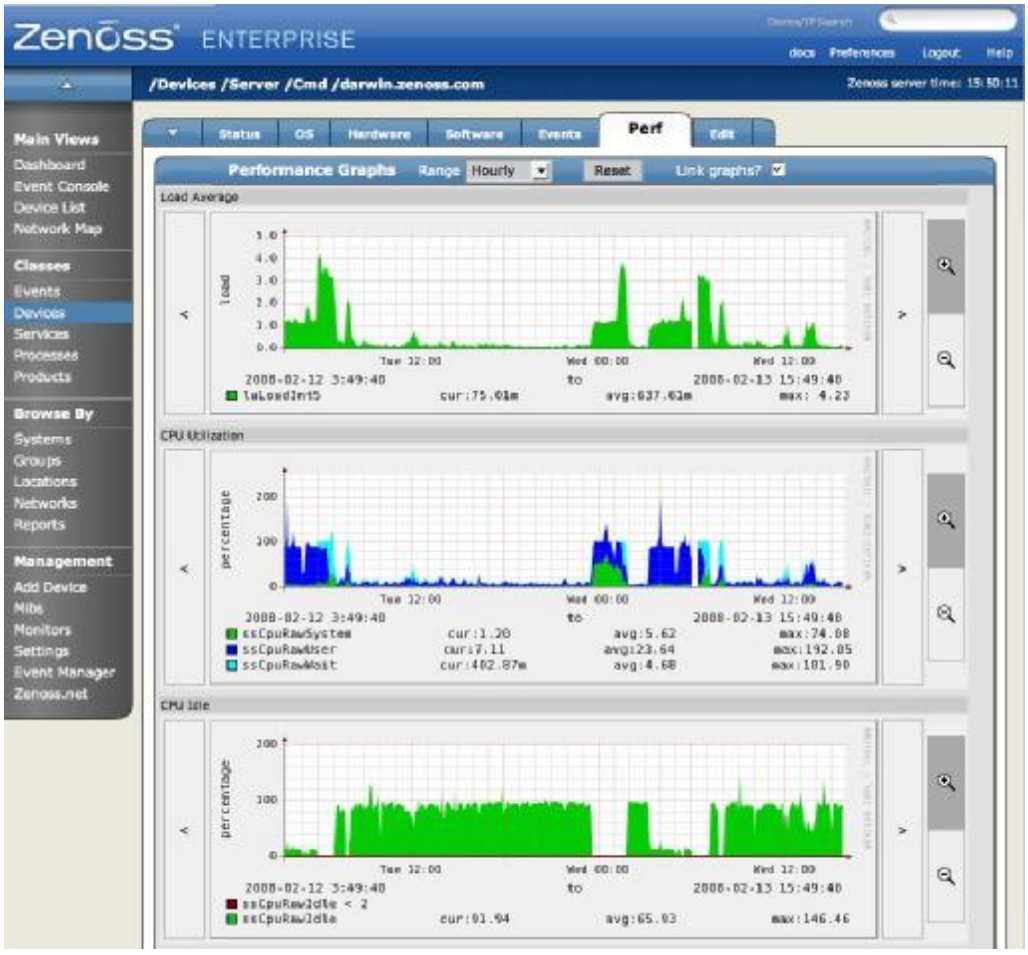
有关服务的 zProperties 的详细信息，请参阅 zProperties 附录。

第12章.性能监视

1. 性能监视

Zenoss提供集中不同的方法用于监视设备以及设备组件的性能矩阵。ZenPerf-SNMP 通过SNMP从设备上采集性能信息。ZenCommand能够通过 telnet 或者 ssh登录到设备并运行脚本来采集性能数据。而ZenPacks则可以提供更多的用于采集性能数据的手段。比如， ZenJMX 可以采集企业级Java应用的性能数据，HttpMonitor 可以检查http的可用性以及web页面的响应速度。而无论采用何种监视手段，配置信息都保存在性能模板中。下面这个图片就是性能标签页(Perf tab)的一个例子。

图12.1. 显示平均负载的性能页



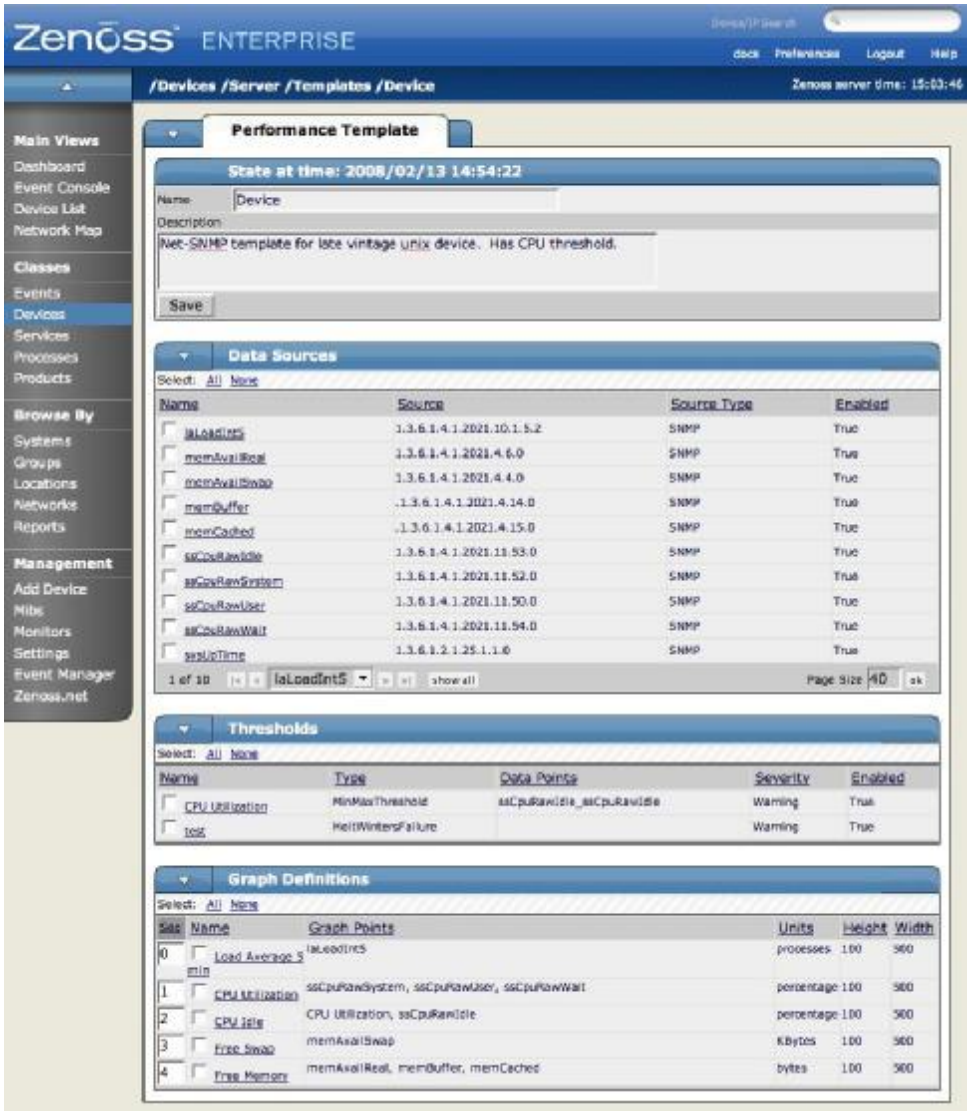
在设备页菜单中，选择更多（More），然后选择模板选项（Templates option），之后点击模板的名称，我们就能找到上图中5分钟平均负责图的性能模板。

2. 性能模板

性能模板定义了采集设备和设备组件性能数据的特定指令。性能模板包含三类子对象：数

据源、门限以及图表定义。数据源明确地指定了要采集的数据点（Data Points）以及用什么方法进行采集。门限定义了被采集数据的预期边界，以及被采集数据突破边界后在系统中产生什么样的事件。图表定义描述了如何在设备或者设备组件的性能页上展示性能图表。

图 12.2. 平均负载图的性能模板



2.1. 模板页

模板页列出一个设备或者某类设备所有可用的模板。对某一类设备来说，可以点击该分类的模板页标签来浏览该分类下所有可用的模板。对单个设备而言，您可以使用页菜单—>更多—>模板菜单项来查看该设备下所有的模板。如果模板列表下有多个同名模板，则仅显示那个可以绑定到该设备或者给设备分类的模板（其它同名模板将被该模板覆写）

3. 模板绑定

模板可以基于设备分类或者单个设备进行定义。在Zenoss进行性能数据采集之前，必须要决定使用哪个模板，这个过程就叫做模板绑定。绑定的第一步是要确定将要使用的模板的名称。对设备组件来说，通常是要确定组件的元类型（比如文件系统、CPU、硬盘等等）。对设备来说，

就要确定一个模板列表，该模板列表存储在一个称为zDeviceTemplates的zProperty中。模板绑定的第二步是找到匹配这些名字的模板。对于每个模板的名称，zenoss首先在设备中进行搜索，然后到设备分类层次树中搜索与该名字匹配的模板。Zenoss会使用第一个找到的且具有正确名称的模板，忽略其它具有相同名称的模板。

设备或者设备分类的模板页显示所有可用于绑定的模板。该页显示在该点或者比该点更高层次中定义的模板列表。用户可以使用绑定模板(Bind Templates)菜单项来执行模板绑定操作，在系统弹出的绑定模板对话框中，您可以查看或者改变当前设备或者设备分类上绑定的模板。同时,用户也可以在zProperties标签页中直接修改zDeviceTemplates属性来实现模板绑定的变更。

名称绑定	定义
Device	设备对象自身。（这些OID没有SNMP索引号）
FileSystem	当前使用主机资源MIB的文件系统对象。
Interface	接口使用他们的接口类型进行绑定，比如：ethernetCsmacd
HardDisk	用于显示I/O状态的硬盘对象。

4. 数据源

数据源指定了采集哪些数据点以及如何进行采集。系统中的每一个性能模板都有一个或者多个数据源。Zenoss提供两种内置的数据源：SNMP和COMMAND。其它类型的数据源可以通过ZenPacks提供。在创建新的数据源时，用户可以先选择添加新数据源（Add New DataSource）菜单项，然后从可用的数据源列表中选择一个。所有的数据源类型都有名称和是否可用两个字段。而其它可供选择的选项是否能够显示出来则取决于数据源的类型。

SNMP数据源定义了通过SNMP采集的数据，zenoss通过ZenPerfSNMP进程来采集SNMP数据。除了名称和是否可用两个选项之外，SNMP还要指定一个选项，那就是要采集哪个SNMP OID。注意，许多OID都必须以.0结尾才能正常工作。由于SNMP数据源精确指定了一个性能矩阵，因此，通常情况下SNMP数据源只有一个数据点，更详细信息请参阅SNMP监视的有关章节。

COMMAND数据源指定了通过shell命令采集的数据，该shell命令可以运行在zenoss server上，也可以运行在受监视的设备上，当该shell命令在受监视的设备上运行时，zenoss通过telnet或者ssh会话连接到该设备上运行shell命令。ZenCommand进程用于处理COMMAND数据源。一个COMMAND数据源可能返回一个或者多个性能矩阵且通常每个性能矩阵只有一个数据点（）。使用COMMAND数据源的shell命令必须返回符合Nagios插件输出规范的数据。更详细的信息，请参阅使用ZenCommand进行监视章节的内容。

图 12.3. 数据源

5. 数据点

一个数据源可以返回基于一个或者多个性能矩阵的数据。一个数据源可以包含一个或者多个数据点来表示一个或者多个性能指标。在浏览一个数据源时，用户可以点击添加数据点（Add DataPoint）菜单项来添加一个数据点。数据点有以下一些内容可以进行设置：

Name - 数据点名称，如果是一个SNMP数据点，则与数据源同名。对于命令数据点（COMMAND Data Points），名称与使用的shell命令同名。

Type - Zenoss使用RRDTool来存储性能数据。该设置指明了了要使用哪种类型的RRD数据源来存储该数据点的数据。该设置可用的选项有COUNTER, DERIVE, ABSOLUTE, GAUGE。

如果一个数据源声明为COUNTER，将会按照一个步长周期来保存数据值的变化率。这种情况是基于数据值总在增长的假设（当前值与上一个数据值的差大于0）。路由器上的流量计数器就是使用COUNTER的最好的例子。

DERIVE与COUNTER相同，但它允许出现负值。如果您想要查看服务器上空余磁盘空间的变化率，这是你将会用到DERIVE。

ABSOLUTE 也存储变化率，但其假设其早前的值为0。当前值和上一个值之间的差总是等于当前值。因此，它仅仅存储当前值除以步长周期（我们的例子中是300秒）的结果。

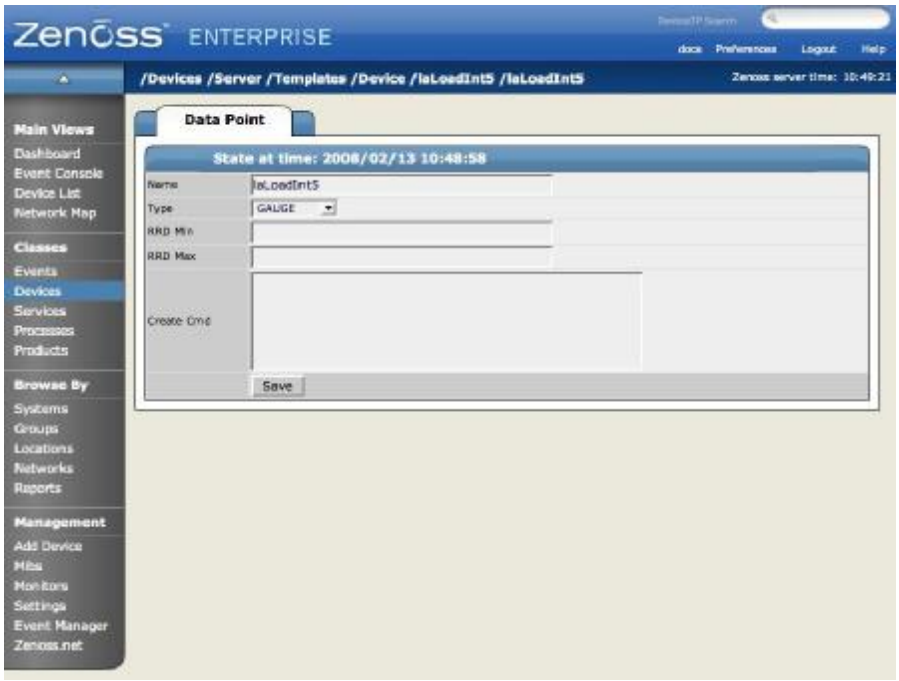
GAUGE 不保存比变化率。它存储自己实际的值，没有除或者其它计算。服务器上的内存消耗是GAUGE的一个典型例子。

RRDMin – 接收到的数据如果低于该值将被忽略。

RRDMax – 接收到的数据如果高于该值将被忽略。

Create Cmd – 是一个RRD表达式，该表达式用于为该数据点创建数据库。如果该字段被留空，Zenoss 将会自动使用一个较为合理的默认值，该默认值将适合绝大多数情况。有关创建命令的更详细信息，请参阅 http://oss.oetiker.ch/rrdtool/doc/rrdcreate.en.html

图 12.4. 数据点



6.门限（Thresholds）

门限定义了数据点的预期边界。当数据点超过门限是，zenoss将产生一个事件。Zenoss提供一个内置类型的门限，Min/Max门限。该门限对到来的数据进行检查，以确定数据是否超出最大值或者低于最小值。ZenPacks能够提供其它类型的门限。Min/Max门限可以设定的内容有：

Name – 该名称显示在性能模板页上，由于创建门限事件。

Data Points – 从该模板选择一个或者多个数据点，该门限将应用于这些数据点。

Min Value –如果该域非空，那么在任何时间，只要一个选择的数据点低于Min Value的值，系统就会产生一个事件。该域可以是一个数字，也可以是一个python表达式。当使用python表达式时，变量”here”是对要采集数据的组件或者设备的引用。举例来说，一个接口的85%门限可以

用如下方式指定：here.speed * .85 / 8（除以8是因为接口速度通常以字节/秒表示，而性能数据通常是按照比特/秒计算）。

Max Value – 如果该区域非空，那么选定的数据点在任何时间只要超过该值都将触发一个事件。与Min Value 一样，该区域除了可以设置成数字外，还可以设置成一个python表达式。

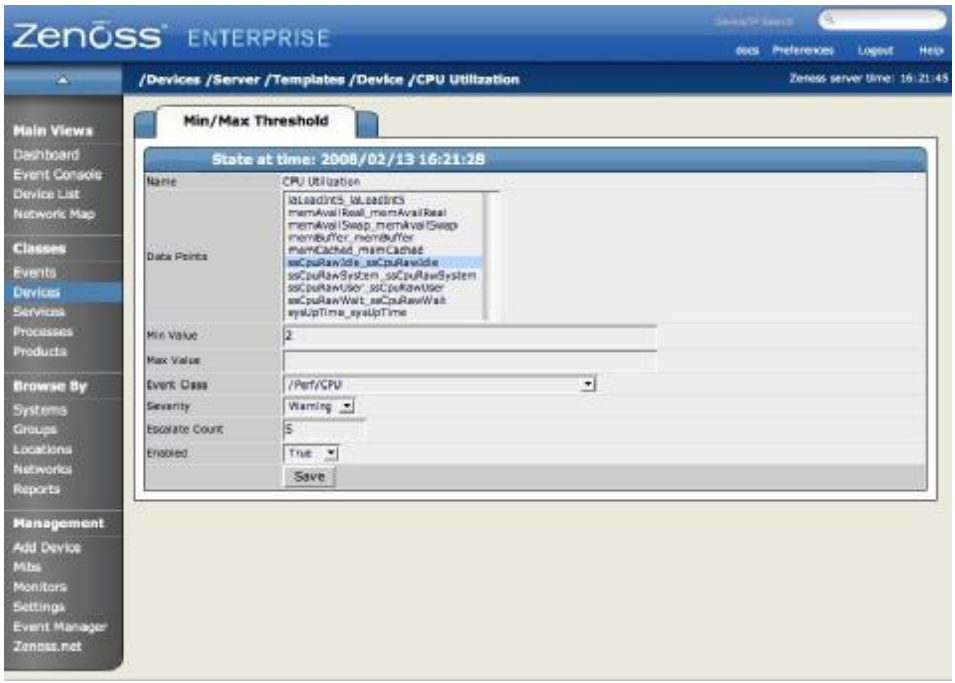
Event Class – 门限被突破后，被触发的事件的类别。

Severity – 门限被突破后，第一个被触发的事件的级别。

Escalate Count – 如果门限被连续打破多次，那么事件的等级将升级一个步长。

Enabled – 允许您启用或者禁用该门限

图 12.5.门限定义



7. 图定义（Graph Definition）

图定义包含了一些设置信息，通过这些设置，可以在设备或者组件的性能标签页上生成性能图表。这些图表可以包括来自性能模板的任何数据点或者门限。用户可以在性能模板页中，点击添加图表（Add Graph...）菜单项来创建新的性能图表。图定义中有以下一些内容可供设置：

- Name – 图的名称。
- Height – 图的高度，以像素计算。

• Width -图的宽度，以像素计算。

• Units – 图表纵轴的单位。

• Log – 如果被设置为True，纵轴刻度将会是对数形式，在数据呈指数级增长时，该设置将非常有用。

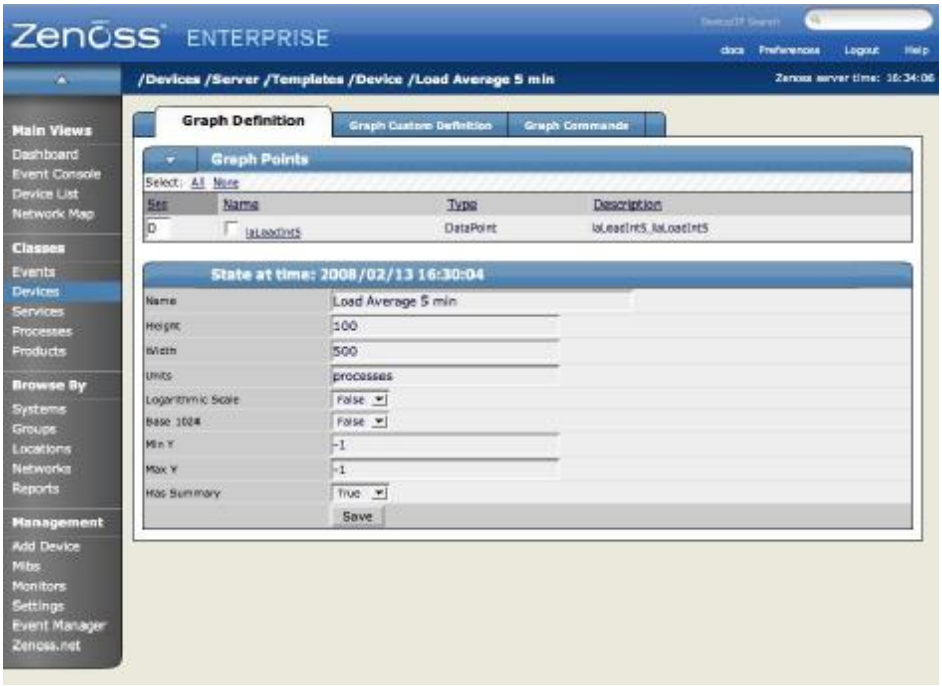
• Base 1024 – 如果您要绘制的数据以1000或者1024的倍数为基数进行计算，则该选项设置为True。

• Min Y – 图表纵轴的最小刻度。

• Max Y -图表纵轴的最大刻度。

• Has Summary – 如果被设置为True，将在图表下方显示数据点当前值、平均值和最大值。

图 12.6. 图定义



7.1.图点（Graph Points）

每一个性能图表的数据点或者门限都由一个图点来表示。对应于数据点、门限或定制图表命令，Zenoss提供几种类型的图点与之对应。只要您愿意，您可以向系统中添加任意多的图点。Sequence域决定了图点在性能图表中出现的顺序。您可以使用重新对图点排序（Re-sequence Graph Points）菜单项对Sequence进行编辑，并以此对图点重新排序。

7.1.1. 数据点图点（Data Point Graph Points）

数据点图点在性能图表上绘制出数据点的值。用户可以使用添加数据点（Add DataPoint）菜单项来添加数据点，在弹出的添加图点对话框中，用户可以选择已经在模板中定义过的一个或者多个数据点。每个选中的数据点都将单独生成一个数据点图点。而且，还有一个包含相关门限（Include related thresholds）的checkbox控件，一旦该控件处于选中状态，系统将会为任何

应用到被选中的数据点的门限创建图点。点击图点的名称，您将会看到图点的编辑页。该页面中，您可以看到不同类型图点的不同设置。数据点图点有以下一些设置内容：

• Name – 显示在图定义页的名称，默认情况下也是用在图表说明上的名称。

• Consolidation – 该选项是一个 RRD功能，该功能用于将数据点的数据按照性能图表的尺寸绘制出来。绝大多数情况下，默认值AVERAGE 都是合适的。

• RPN – 您可能会输入一个RPN表达式，该表达式用于对正被绘制的数据点的值进行告警。举例来说，如果数据以比特存储，但您希望数据以字节为单位绘制图表，你可以使用"8,/" 这个字符串。更多有关RRDTool的RPN的说明请参阅
http://oss.oetiker.ch/rrdtool/tut/rpntutori-al.en.html

• Limit – 您可以指定能够被图形化的数字的最大值。

• Line Type – 选择Line选项可以将数据绘制成线条的方式。选择Area选项将会使线条和横轴之间的区域以一定的颜色填充。如果选择None选项，系统将不再绘制图形，该情况适用于您希望使用自定义的RRD命令来控制数据点。

• Line Width – 线条的宽度，单位以像素计算。

• Stacked – 如果该选项被设置为True， 那么在早前绘制过的数据之上，系统将再次绘制当前数据点的Line或者Area。在时间轴上任何一个刻度上，性能图表所代表的值，将会是早前绘制的值与当前绘制的数据点的值之和。举例来说，如果您正在测量IP包的出入流量，您可以将两个性能图表堆叠在一起以获得一个总流量。

• Color – 您可以为线条或者区域指定一种颜色。系统可以识别部分颜色的英文名称，但我们推荐使用三位或者六位的十六进制字符串来表示颜色。

• Format – 在性能图表中显示采集到的性能数据时使用的RRD格式。RRDTool的格式信息请参阅
http://oss.oetiker.ch/rrdtool/doc/rrdgraph_graph.en.html

• Legend – 显示在性能图表上的图例名称。默认情况下是一个TALES 表达式，该表达式指定了图点的名称。该表达式中可以使用的变量有需要绘制性能图表的设备或者组件。

• Available RRD Variables – 该字段列出来已经在该图定义中定义过的RRD变量，这些RRD变量可能会用在RPN字段。

7.1.2. 门限图点（Threshold Graph Points）

门限图点用于在性能图表中绘制门限的值。我们可以使用“添加门限”（Add Threshold）菜单项来添加一个新的门限图点。在系统弹出的添加门限图点对话框中，您可以选择模板中已经定义过的一个或者多个门限。每个被选中的门限都将单独生成一个门限图点。门限图点有几个选项可供设置，分别是：名称（ Name）、颜色（ Color）和图例名称（Legend）。详细说明参见上文数据点图点的相关说明。

7.1.3. 定制图点（Custom Graph Points）

定制图点允许您向图定义中插入特定的RRD图命令。有关DEF, CDEF 及 VDEF命令的更详细内容，请参阅 http://oss.oetiker.ch/rrdtool/doc/rrdgraph_data.en.html.
有关其它RRD命令的更详细内容，请参阅
http://oss.oetiker.ch/rrdtool/doc/rrdgraph_graph.en.html

7.2. 定制图的定义（Graph Custom Definition）

定制图定义标签页能够使您精确地定义您自己的一套RRD命令来绘制性能图表。图定义标签页中指定的图点主要被用于定义数据，这些数据对于您在这里指定的命令是可用的，但除非您显示地使用命令来绘制这些数据，否则这些图点将不会被绘制成图形。可用的RRD变量列出了由图点定义的值。

7.3. 图命令标签页(GraphCommands tab)

图命令标签页显示用于画图的RRD命令。在使用自定义图点或者定制图定义时，该标签页将会是一个非常好的调试工具。

8. 改变图的顺序

您可以自定定义性能图表的许多特点，举例来说，您可以改变图表在页面中出现的顺序，操作步骤如下所示：

1. 在系统中导航到您的设备.
2. 从设备的页菜单中选择更多选项（More） >>模板（Templates）。
3. 点击创建本地副本（Create Local Copy）按钮。
4. 点击模板的名称。
6. 在页面的图表区域中，使用Seq boxes控件来改变图表在页面上出现的顺序。

9. 监视文件系统与改变门限

Zenoss同样还可以监控文件系统的利用率。如果要查看文件系统的统计数据，请采取以下

步骤：

1．导航至某个设备的OS标签页。.

2．点击 “/” 文件系统，用户将能看到该设备文件系统的统计数据。用户还可以在文件系统的本地模板中改变产生事件的门限。下面的这个例子将详细介绍如何创建一个较低的门限，以及如何基于这个门限产生新的事件。

3．点击更多（More），之后选择模板（Templates）.

4．点击创建本地副本（Create Local Copy）按钮以创建该设备模板的一个本地副本。

5．在门限区域内，在添加对话框中输入名称 “Really Low Threshold”并点击添加按钮，之后 Really Low Threshold 将出现在列表中。

6．点击 Really Low Threshold的链接。

7．选择 usedBlocks选项来使用文件块为计算单位。

8.在Max Value区域中，输入以下表达式：

```
here.totalBlocks * .05
```

通常，该设置作为文件系统最低使用率5%的一个门限设置。

第13章.通过SSH远程监视设备

1. 通过SSH远程监视设备

Zenoss允许用户通过SSH远程监视设备，但必须在设备上安装Zenoss插件，插件的安装有两种不同的方式。

1.1. 在远程主机上安装 Zenoss插件

Zenoss插件有两种打包方式：我们推荐在RedHat Linux系统中使用原生格式(RPM)，RedHat可以对RPM包进行直接管理。使用RPM包还可以使系统管理员更容易进行包的升级；另一种方式是源代码发行包，这种包的安装必须通过安装工具进行，因此开发人员更多地使用这种方式。使用源代码发行包的好处是，安装Zenoss插件时无需有root权限。

1.1.1. Zenoss 插件安装技术: RPM

Zenoss插件的RPM安装包是noarch RPM，也就是说这种RPM安装包可以安装在任何一种架构上 (i386, amd64, ia_64)。安装Zenoss插件RPM包的唯一前提是，必须安装python 环境。而绝大多数的Linux发行版已经内置了Python。

用户可以使用如下命令安装zenoss 插件RPM包:

```
$ sudo rpm -Uvh zenoss-plugins-*.rpm
```

在这里，'zenoss-plugins-*.rpm' 是指Zenoss 插件RPM 包文件。

1.1.2. Zenoss 插件安装技术: setuptools

正如我们在上文中指出的，使用安装工具安装Zenoss插件不需要root权限。Zenoss插件的工具安装支持所有参数，但绝大多数用户都使用默认安装命令，如下所示：

```
$ python setup.py build
```

```
$ sudo python setup.py install
```

上述命令将把Zenoss插件安装到用户可访问的目录，如果您无法安装系统软件（），您仍旧可以使用Zenoss插件。然而您必须进行一些额外的工作，这样Zenoss插件才能完成安装。有关使用非特权帐号进行Zenoss插件安装的更详细信息，参见如下链接：

http://dev.zenoss.org/trac/wiki/FAQ#HowdoIinstalltheZenossPlugins

另外一种下载源代码包、展开包并运行setup.py 的方法是使用setuptool的内置命令'easy_install'。如果想要自动下载并使用easy_install来安装Zenoss插件,可以使用如下命令：

```
$ sudo easy_install Zenoss-Plugins
```

这里的 'Zenoss-Plugins' 就是当前 Zenoss 插件文件的名称。

1.1.3. 测试插件安装

我们可以使用zenplugin.py命令对Zenoss插件进行操作，当不加参数运行该命令时，zenplugin.py将会提供脚本的正确用法 。

Zenoss 插件通过插件来测试平台上某些特定运行时的值。举例来说，linux2平台的CPU插件使用/proc来读值，而freebsd5的CPU插件则使用不同的技术。为了测试插件的安装，您必须确定目标操作系统上能够使用哪些插件。您可以在目标操作系统上运行以下命令：

```
$ zenplugin.py --list-plugins
```

在确定了目标操作系统上能够支持的插件列表后，您可以运行zenplugin.py命令，同时以插件的名称作为命令的参数，如下所示：

```
$ zenplugin.py cpu
```

1.1.4. 插件安装的排障

1.1.4.1. 运行zenplugin.py时提示"Command not found"

如果在运行zenplugin.py命令时收到了 "command not found"的提示，那么就要检查一下zenplugin.py的安装目录是否包含在了系统的PATH变量中。如果使用的是RPM方式进行的安装，可以使用"rpm -ql zenoss-plugins | grep zenplugin.py"进行检查。如果您使用setuptools进行的安装，就要注意以下"Installing..."安装信息，以确认zenplugin.py的完整安装目录。

1.1.4.2. 出现提示"platform 'XXX' is not implemented. no plugins exists"

该提示消息说明您的操作系统不支持该插件，如果您收到了该信息且仍旧希望该插件能够在您的操作系统下运行，您可以把下面这个命令的运行结果发送邮件给Zenoss的开发团队。

```
$ python -c 'import sys; print sys.platform'
```

1.1.5. 将Zenoss改为使用SSH远程监视设备

如果某组设备希望使用SSH远程监视，那么您就必须修改该分组的zProperties。

1. 导航到您希望远程监视的设备的设备分类路径。您可以远程监控该分类下的单个设备，也可以监视该分类下所有的所有子分类设备。点击zProperties标签页，改变该分组的zProperties的值。

Figure 13.1. 设备组的 zProperties 标签页

以下这些zProperties的值必须改变：

- zCollectorPlugins
- zCommandPassword
- zCommandPath
- zCommandUsername
- zSnmpMonitorIgnore
- zTransportPreference

以下是在我们远程设备上已经设定好的值

zProperties	值
zCollectorPlugins	snmp|portscan
zCommandPassword	远程主机的 SSH 口令
zCommandPath	到zenplugin.py的路径
zCommandUsername	远程主机SSH用户名
zSnmpMonitorIgnore	True
zTransportPreference	command

注意：得到设备的完整的模型有两步要走，首先要知道即将要运行插件的系统的操作系统类型，然后再运行如下命令：

```
$ zenmodeler run -d enter_server_name_here
```

1.1.6. 使用预定义的 /Server/Cmd 设备分类

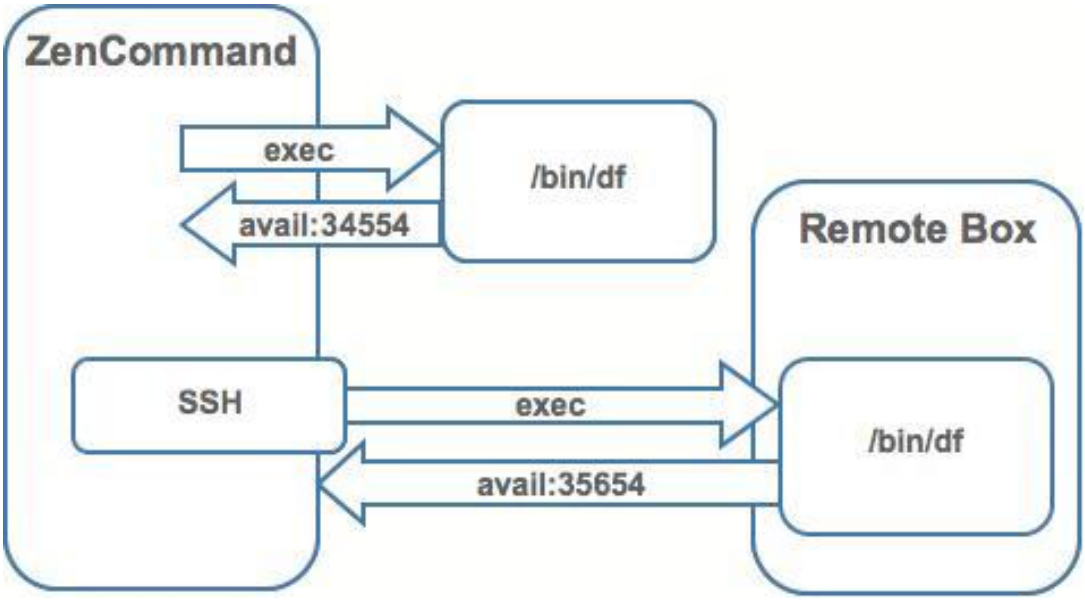
/Server/Cmd 设备分类是系统给出的一个例子，该例子说明了如何使用SSH对设备进行监视。在该设备分类中，zProperties已经像前面讲的那样进行了修改，并且设备、文件系统、以太网接口的模板已经被创建。您可以把该分类当作是您自己配置的一个参考，或者，如果您有一个设备需要进行建模并通过SSH远程监视，您就可以将该设备直接放到该目录下。但是您仍需要对设备的zCommandUsername和zCommandPassword属性进行设置，以便Zenoss能够使用SSH连接采集设备的信息。

第14章.使用ZenCommand进行监视

1. 关于ZenCommands

Zenoss可以通过ZenCommand进程运行Nagios® 或者 Cactii插件。在早期的Zenoss版本中，这个进程叫做Zenagios。ZenCommand进程可以在本地运行Nagios® 或者 Cactii插件，也可以通过原生的SSH传输远程运行Nagios® 或者 Cactii插件。当Zenoss运行这些插件时，Zenoss会跟踪每个插件的返回代码，并根据插件的输出创建相应的事件。同样的，Zenoss还可以从插件中跟踪性能信息。

图 14.1. 运行ZenCommands



2. 例：编写一个ZenCommand (例：check_http)

您可以使用ZenCommand插件(check_http)来来检查一个Web页面的特定内容（隐含着对server/page 200状态的检查）。以下这个例子说明了如何建立一个ZenCommand插件来检查网页的特定内容。

1. 确定您是一个Zenoss用户。

2. 从命令行测试插件。一旦该插件能够在Zenoss下正常工作，check_http -h命令将显示所有的插件选项。下面的这个命令用来测试产品目录。

```
$ZENHOME/libexec/check_http -H www.zenoss.com
```

如果check_http命令能正常运行，那么将会产生类似下面的输出：

```
HTTP OK HTTP/1.0 200 OK - 0.723 second response time |time=0.722758s;;;0.000000
size=7932B;;;0
```

显然，该命令插件返回了一个合法的输出，这说明我们为下一步做好了准备。

4. 将一个您想要检查的设备(一个运行www站点的设备)添加到Zenoss系统中，添加设备时，要注意发现协议要选择'none'。

5. 在Zenoss系统中导航到该设备,使用页菜单选择更多(More)选项,然后选择模板(Templates)。

6.点击创建本地副本（Create Local Copy）按钮。您现在已经拥有了一个可以覆写（override）默认设备模板的本地模板。

7. 进入到该模板中删除sysUpTime数据源，因为对该设备不使用SNMP。

8. 给这个新模板起一个新的名字，比如叫做“Web testing template”。

9. 添加一个叫做rootWebCheck的新数据源。

10. 到rootWebCheck中设置以下变量。
- Source Type = COMMAND
- Component = rootWebCheck
- Cycle Time = 30

11. 通过前台运行zencommand来调试您的ZenCommand，打开调试选项，命令如下：

```
zencommand run -d www.website.com -v10
```

www.website.com是您想要监视的站点。

12.命令模板域是一个TALES表达式，所以我们可以在我们的命令中进行一些替换,这样就可以使我们的命令对任何一个加入该分类的设备都变得通用。首先，让我们设置-H参数，该参数后跟设备的IP，该设备就是我们命令运行所针对的设备。

```
check_http  -H ${here/manageIp}
```

13. 现在，让我们添加一个用于查找页面内容的检查， -r参数将运行一个针对页面文本内容检查的正则表达式。现在，我们的命令已经由三部分构成:

```
check_http  -H ${here/manageIp} -r textstring1
```

在这里，textsrting1是一些你知道的文本，将显示在结果web页面中。

14. 举例来说，您想要这个命令变得通用,所以可以为regex创建一个custom域，regex 可针对设备逐个改变。将custom域默认设置为 “.*”,这将匹配所有。到/Devices/Custom 模式下添加一个新域，如下所示:

- Label = Web Match Regex
- Name = cWebMatchRegex
- Type = string
- Default = .*
- Visible = True

15. 现在返回到我们的模板，将命令改成以下形式：

```
check_http  -H ${here/manageIp} -r ${here/cWebMatchRegex}
```

16. 现在，将regex的值添加到上面这个例子中我们使用的WebMatchRegex中。

17. 从命令行测试您的ZenCommand。

3. 用ZenCommand采集数据

如果想要从上面这个ZenCommand 例子中采集数据并显示这些数据，您可以将这些数据记入日志，这样您就可以以图形的方式看到诸如响应时间等数据了。

1. 到/Web/Device模板。

2. 到您在check_http 例子中创建的数据源（Data Source）。

3. 在底部有一个数据点（Data Points）表，添加一个叫做“time”的数据点。

4. 标尺（GAUGE）的默认类型已经够用，所以不必修改该数据点。

5. 再次测试您的命令，您将会看到一条日志消息，该消息以下面内容开头：

```
DEBUG:zen.zencommand:storing responseTime = 1.0
```

6. 创建一个图表来显示您的数据。在设备模板上，创建一个叫做“Web Response Time”的图表。

7. 对该图表进行以下设置：

- Data Sources = rootWebCheck_responseTime
- Units = Seconds
- Min Y = 0

8. 现在查看一下设备www.website.com 的性能标签页，您可以看到性能图表。图表数据只有在经过数次数据采集之后才会显示。重新启动zencommand,这样新的配置将立即生效。

4. ZenCommands的插件格式

Nagios® 插件的配置是通过使用一个命令模板进行的，该命令模板非常像用于性能监视的RRD模板。所以一个叫做“Device”的模板将绑定到该模板定义下的所有设备。 在每个模板中都有一个可运行的命令的列表。这些命令可以是任何符合Nagios®插件标准的程序。Nagios®插件标准的定义参见下面的链接：

http://nagiosplug.sourceforge.net/developer-guidelines.html

Nagios® 插件有几个域:
- name - 命令对象的名称
- enabled - 该命令是否应当被用在给定设备上
- component - 当zencommand向Zenoss发送事件时使用的组件名称
- event class - 向Zenoss发送事件时使用的事件分类
- severity - 向Zenoss发送事件时使用的默认事件级别
- cycle time - 命令运行的频率，单位为妙
- command template - 要运行的命令

命令模板字符串使用Zope TALES构造，对模板评估时要传递以下参数：

- zCommandPath - 在某台主机上到zencommand插件的路径，该路径来自zCommand-Path这个zProperties。如果命令的起始缺少路径，则zencommand将会被自动添加到命令中。
- devname - 设备名称，命令正针对该设备进行评估。
- dev - 设备对象，命令正针对其进行评估。
- here - 评估环境。对一个设备而言，相当于文件系统或者接口之类的组件。
- compname - 如果该命令针对一个组件进行评估,那么这就是该组件的名称。
- now - 当前时间。

Template值的访问像shell变量一样。它们的语法表达与我们在本手册中介绍的TALES表达式一样，以下是一些例子：

使用url /zport/dmd 对所有的设备运行http check:

```
check_http  -H ${devname}-u /zport/dmd
```

在一个叫FileSystem的模板上，下面的命令将针对一个设备上的所有文件系统(FileSystem)运行 ：

```
check_disk  -w 10% -c 5% -p ${compname}
```

5. 测试ZenCommands

ZenCommand数据源可以通过使用zentestcommand shell脚本来测试，脚本可以从命令行运行，如下所示：

```
zentestcommand  -d devicename --datasource=datasourcename
```

在这里，devicename是您希望命令在上面运行的那台设备，datasourcename 是一个模板上的数据源的名称，该模板与设备关联。zentestcommand 将会把命令的结果打印到stdout上。

第15章. 监视Windows设备

1. 监视Windows设备的准备

在对Windows设备进行监视之前，Zenoss需要做一些准备工作，在监视Windows设备之前，要确保以下工作已经进行：

1. 检查并确认SNMP代理能够正常运行。
在Windows 的控制面板中，添加删除程序?添加删除Windows组件 ->管理和监视工具。.

2. 确保用于WMI连接的DCOM正常运行。

3. 作为另外一种可选操作，您可以使用SNMP-Informant 来获取CPU, Memory和Disk I/O 的状态信息。SNMP-Informant 代理通过服务器上的WMI来读取Windows设备上的信息，并将系统、状态和操作数据转化为SNMP OIDs 来进行广播。这样一来，Zenoss就可以处理SNMP OIDs信息，并基于这些信息产生事件和告警。在Zenoss企业版中，SNMP-Informant并不是必须的，如果不使用SNMP-Informant，将会限制Zenoss能够处理信息的数量。

2. 设定Windows的zProperties

本章节将介绍重要的 zProperties 的设置，这些 zProperties 的设置对从 Windows 设备上读取信息到 Zenoss 系统中来说是非常关键的。对于每个 Windows server，用户可以导航至 zProperties 标签页并进行如下设置：

- zWmiMonitorIgnore – 使用该属性来打开或者关闭 WMI 监视。如果被设置成 False，Windows 监视将打开，否则 Windows 监视将被关闭。用户可以在 Server/windows 级别来设置该属性，这样一来，所有位于该路径下的 Windows 设备都将被自动监视。
- zWinUser – 必须是本地 admin.，zWinUser 的格式是：
- \Username – 本地账户的用户名.
- DOMAIN\Username – 域账户的用户名格式
- zWinPassword – 输入用于登录远程 Windows 主机的口令。

3. 在Windows主机上测试WMI

在 Windows 主机上按照下述步骤进行操作，以检查 WMI 连接是否正常。

1. 运行 wbemtest。
2. 点击 “Connect…”
3. 在命名空间（Namespace ）域中输入 \\HOST\root\cimv2。
4. 分别输入登陆该主机的用户名和口令。
5. 点击查询（“Query…” ）按钮。
6. 输入 “select * from win32_service” 将返回主机上的服务列表。

4. 其它可选的 Windows 配置项

还有一些其它的选项可以用来收集 Windows 设备上的额外信息，如果您在 Windows 设备上装有以下软件，Zenoss 就可以获得更多的信息：

- Dell Open Manage Agent 能够提供有关操作系统和硬件的更详细信息。
- HP Insight Management Agent 能够提供有关操作系统和硬件的更详细信息。

5. 采集 Windows Eventlog 事件

ZenEventlog 用于采集 (WMI) event log 事件。使用以下的 zProperties 来定义如何处理和监视 Windows EventLog 事件：

- zWinEventLog – 该属性决定 Zenoss 是否读取 Windows 主机上的 event log。
- zWinEventLogMinSeverity – 设定采集 Windows Eventlog 的最低级别，要注意，数字越高，级别越低。1 是最为严重的级别，5 是最低级别。

6. 使用 SNMP-Inform-ant 监视 Windows 性能

Zenoss 能够使用 SNMP-informant 来获取 Windows 设备的 SNMP 信息。首先，用户必须安装 snmp-informant 的免费版本，下载地址是 http://www.snmp-informant.com。要确保 SNMP Informant 进行了正确的安装和运行，运行以下命令来浏览\SNMP Informant MIB:

```
snmpwalk -v1 -c<community> <server> 1.3.6.1.4.1.9600
```

如果 SNMP-informant 配置正确并且正常运行，该命令将返回部分性能信息。

7. 在Windows主机上使用Winexe运行命令

您可以使用 winexe 命令在受 Zenoss 监视的 Windows 主机上运行命令，用法如下：

```
$ZENHOME/bin/winexe [options] //host [command]
```

选项	使用
--uninstall	远程执行后卸载Winexe服务
--reinstall	在远程执行之前重新安装 winexe 服务
-system	使用系统帐号
-runas=[DOMAIN\]USERNAME%PASSWORD	作为用户运行 (注意：口令以明文方式在网络间传送)
帮助选项	**使用**
-?, --help	显示帮助消息
--usage	显示简明用法
Common samba options	用法
-d, --debuglevel=DEBUGLEVEL	设定debug level
--debug-stderr	将 debug 输出发送至STDERR
-s, --configfile=CONFIGFILE	使用替代配置文件
--option=name=value	从命令行设置smb.conf 选项
-l, --log-basename=LOGFILEBASE	log/debug 文件的Basename
--leak-report	启动退出时的talloc leak报告
--leak-report-full	启动退出时的完整talloc leak报告

-V, --version	打印版本
连接选项	**用法**
-R, --name-resolve=NAME-RESOLVE-ORDER	仅使用名字解析服务
-O, --socket-options=SOCKETOPTIONS	要使用的socket选项
-n, --netbiosname=NETBIOSNAME	主netbios 名称
-W, --workgroup=WORKGROUP	设置workgroup 名称
--realm=REALM	设置 realm 名称
-i, --scope=SCOPE	使用该 Netbios scope
-m, --maxprotocol=MAXPROTOCOL	设置最大协议级别
验证选项	**用法**
-U, --user=[DOMAIN\]USERNAME[%PASSWORD]	设置网络用户名
-N, --no-pass	不询问口令
--password=STRING	口令
-A, --authentication-file=FILE	从文件获取证书
-S, --signing=on|off|required	设置客户端的签名状态
-P, --machine-pass	使用存储的机器帐号口令(implies -k)
--simple-bind-dn=STRING	DN to use for a simple bind
-k, --kerberos=STRING	使用 Kerberos
--use-security-mechanisms=STRING	Restricted list of authentication mechanisms available for use with this authentication

第16章. SNMP监视

1. SNMP的命令行使用

OID代表性能图表数据来源的数据点（data points）。有些时候，图表不显示的原因就是因为该图表表的OID对设备不可用。您可以使用命令行来测试OID的可用性，看看OID是否真的有返回值。测试OID的可用性需采取以下步骤：

1. SSH 连接至Zenoss实例。

```
Username: root
Password: zenoss
```

2. Run the command snmp get for one of the OIDs在这个例子中我们使用如下命令:

```
$ snmpget -v1 -cpublic build .1.3.6.1.4.1.2021.4.14.0
```

如果该OID可用则返回一个值。

以下是一些基本的用于采集信息的SNMP操作。

a. 浏览一个基本的系统MIB。

```
snmpwalk -v1 -cpublic <device name> system
```

b. 浏览接口描述

```
snmpwalk -v1 -cpublic <device name> ifDescr
```

c. 获取一个单值。

```
snmpget -v1 -cpublic <device name> ifDescr.2
```

d. 一个OID值的详细描述。

```
snmptranslate -Td RFC1213-MIB::ifDescr
```

e. 将一个名称转换为原生的OID（raw OID）。

```
snmptranslate -On RFC1213-MIB::ifDescr
```

f. 将一个原生的OID转换为短名称

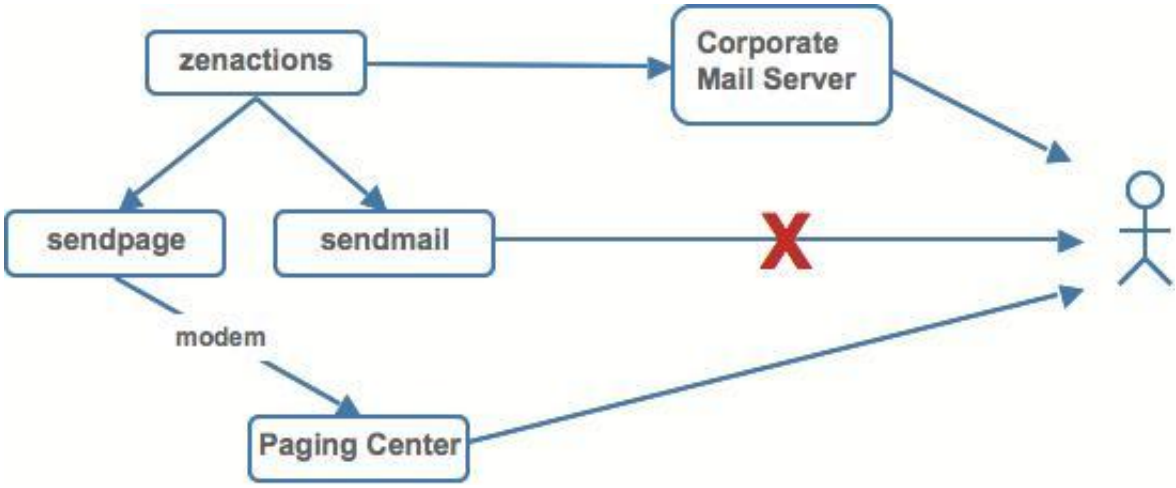
```
snmptranslate -OS .1.3.6.1.2.1.2.2.1.2
```

第17章. 告警规则

1. 告警的创建和使用

ZenActions进程提供基于系统接收或者发送的事件来发送邮件或者寻呼的功能。该进程持续对每个用户的寻呼规则进行判断，每个用户都有自己单独的告警规则。

图 17.1. 发送告警

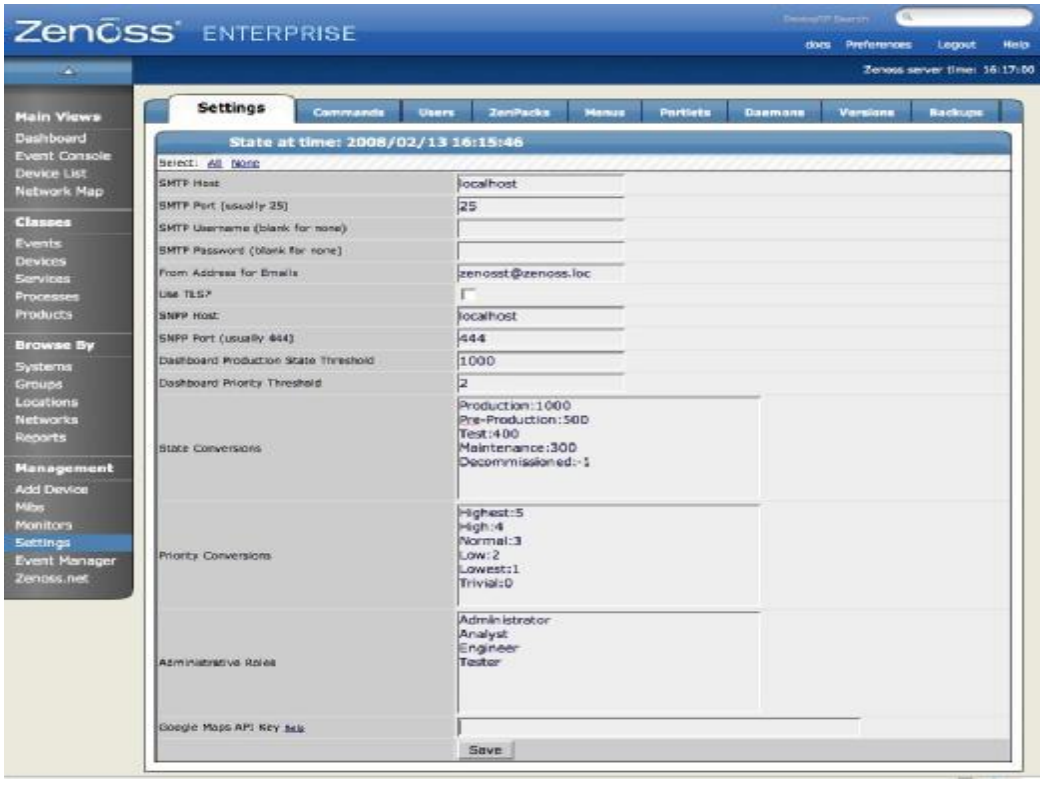


1.1. 为告警配置SMTP设置

如果要使用电子邮件或者寻呼告警，用户必须指定Zenoss的邮件中继：

1. 从左侧的导航菜单中选择设置（Settings），系统将显示设置页。

图 17.2. SMTP设置的配置页标签



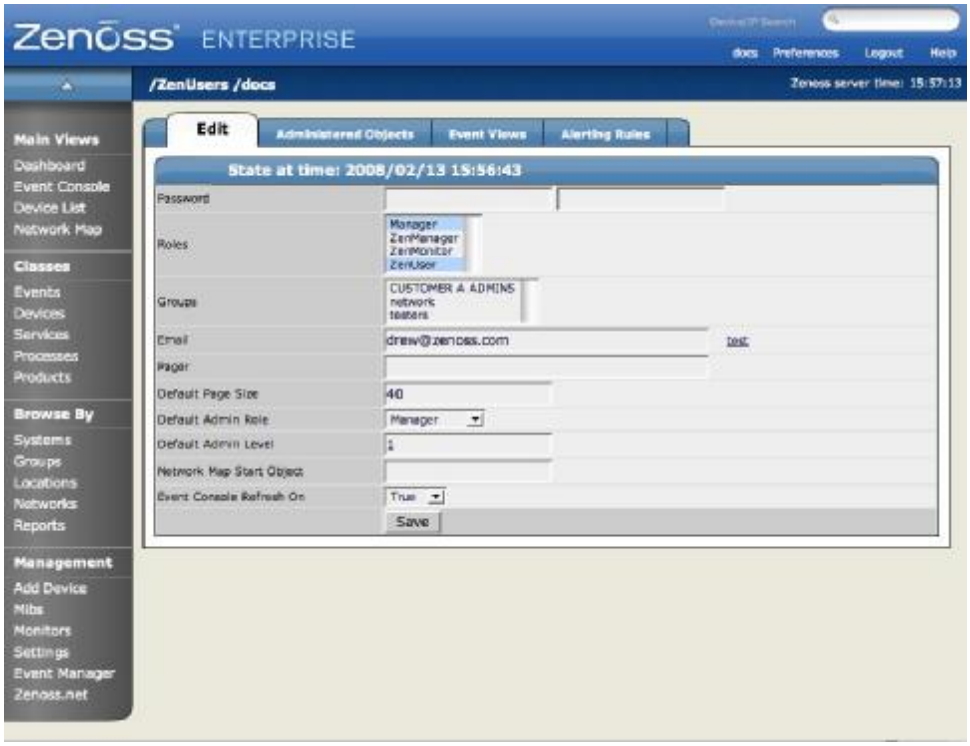
2．要设定邮件服务器，就必须配置SMTP Host,、 SMTP Port、 SNPP Host以及 SNPP Port，之后用户就可以进行告警规则的设置了。

2.新建告警规则

告警规则是基于单个用户创建的，可以为告警规则添加一个额外的接收人，但告警规则创建时是与一个用户帐号进行关联的。

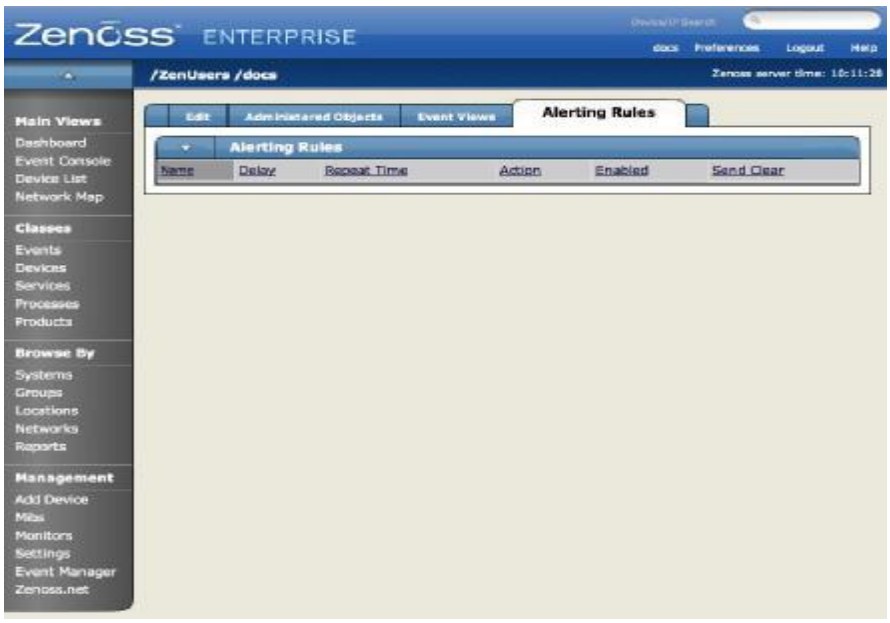
1．从仪表盘的右上侧选择Preferences链接，系统将显示Preferences页面。

图 17.3. Preferences 标签页 –编辑标签页



2．选择告警规则标签页，如下图所示：

图 17.4. 告警规则页标签



3．从告警规则表菜单中，选择添加告警规则（Add Alerting Rule），系统弹出添加告警规则对话框。

图 17.5. 添加告警规则对话框

4．在ID域中输入告警规则名称.

5．点击OK 按钮。系统显示主要告警规则页，在该页面上显示用户刚刚创建的告警规则。

6．点击刚刚创建的告警规则的名称，系统弹出告警明细页，如下图所示：

图 17.6. 告警明细编辑页标签

2.1. 定义与使能告警

在该页面上进行属性设置，之后点击编辑标签页（Edit tab）。

1. 使用Delay域，可以设置告警发送前等待的时间。如果一个事件在延迟时间到来之前已被清除，那么系统将不发送告警。

2. 如果要启用告警，将Enable设置为True。

3. 使用Repeat Time来设置重复时间，也就是告警每隔多长时间发送一次，直至事件被确认或

者被清除。

4．在Action区域内，选择您希望系统发送电子邮件或者是寻呼。如果选择的是发送电子邮件，那么事件将会以电子邮件的方式发送。如果选择发送寻呼，那么就必须要设置SMTP服务器了。很多移动通信系统都有SMTP到SMS的网关，因此可以将发送电子邮件等同于发送手机短消息。默认情况下，邮件告警将发送至用户的电子邮件帐户，而寻呼告警需要用户设定寻呼地址。

5．用户可以在Where区域中设置告警的门限。系统默认的告警规则包含这样一些门限，那就是事件的状态为“New”，事件等级高于”Error”，生产状态为”Production”，突破这样门限的事件将触发告警。您可以通过在弹出菜单中修改门限的值来修改这些门限的定义。

6.您还可以在Where区域中添加更多的过滤条件，方法是，在添加过滤器（Add Filter）菜单中添加一个过滤条件，系统将会弹出一个菜单，您可以在该菜单中添加一个额外的值来过滤事件。要移除某告警的任何一个过滤条件，可点其前面的减号（-）按钮。

7．点击保存（Save）按钮来保存您刚刚在该标签页上输入的信息。

2.2. 1.1.1 创建告警消息的上下文

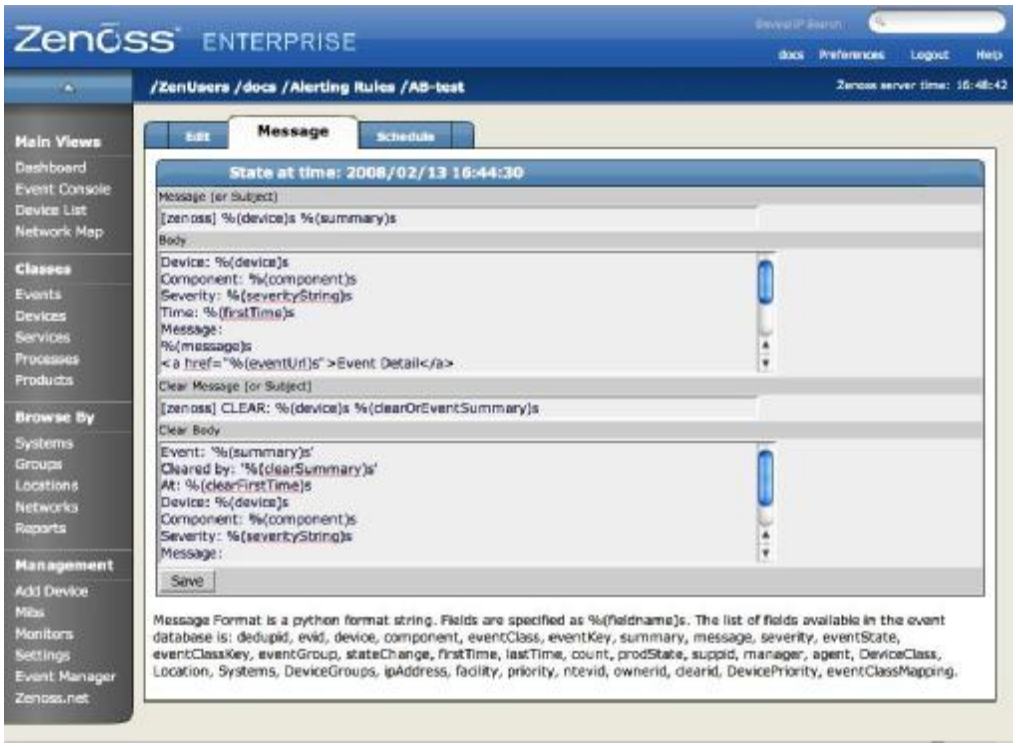
1.从告警规则页上，点击消息（Message）标签页可以对发送到特定地址的消息进行定制，如下图17.7所示：

2．使用消息标签页来指定电子邮件的标题和正文，也就是说，实际上您会有两个消息需要创建。第一个（称为消息）是门限被突破时要发送的消息，第二个消息（称为清除消息）是事件被清除时要发送的消息。标题和消息正文区域需要填写Python格式的字符串。

在页面的底部有一个告警可用的字段列表。清除事件的字段前面都有前缀”clear”。举例来说，prodState字段将变成clearProdState字段。 清除事件还有一个比较特殊的字段clearOrEventSummary，该字段将打印事件清除的小结，如果该字段不存在，将打印原始的告警小结。该功能对一个清除告警的标题是非常有用的，比如说，如果一个告警没有清除（该告警被从用户界面上删除），那么系统将自动创建一个有意义的告警标题。

3．点击保存(Save)按钮，保存您刚刚输入的信息。

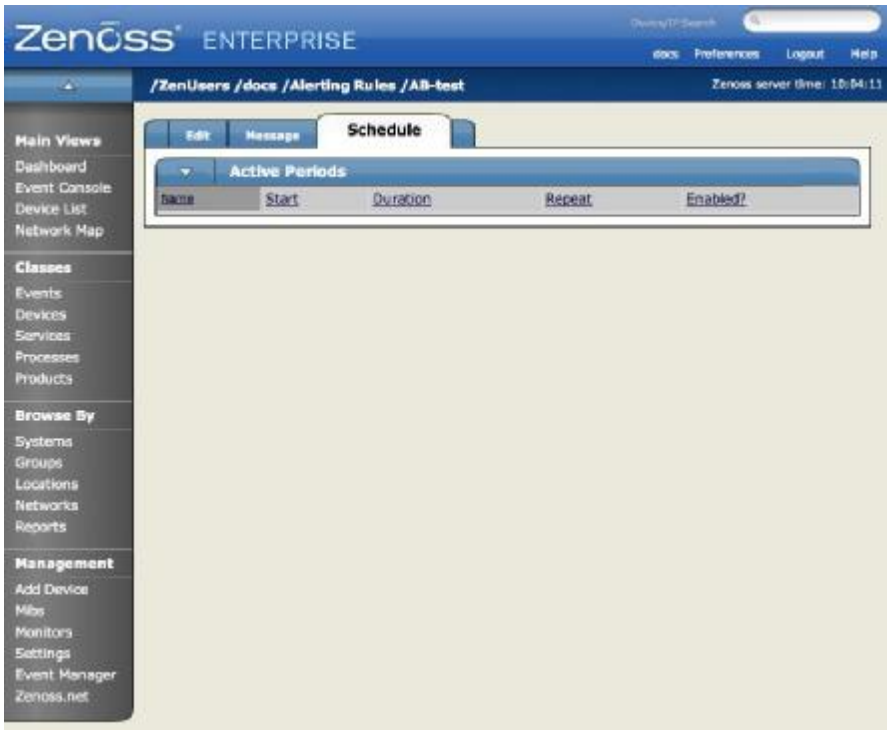
图 17.7. 告警规则消息页标签



2.3. 创建告警发送日程表

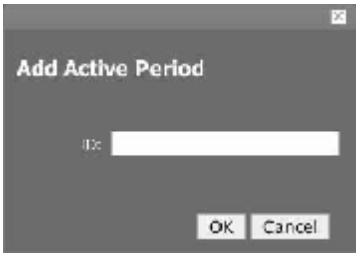
1.从告警规则页中点击日程表（Schedule）标签页来建立告警发送的日程表。如下图所示：

图 17.8. 告警规则 –日程表标签页



2.要为告警添加一个新的日程表，从活动周期（Add Active Period）表菜单中选择添加告警规则，系统弹出添加活动周期对话框，如下图所示：

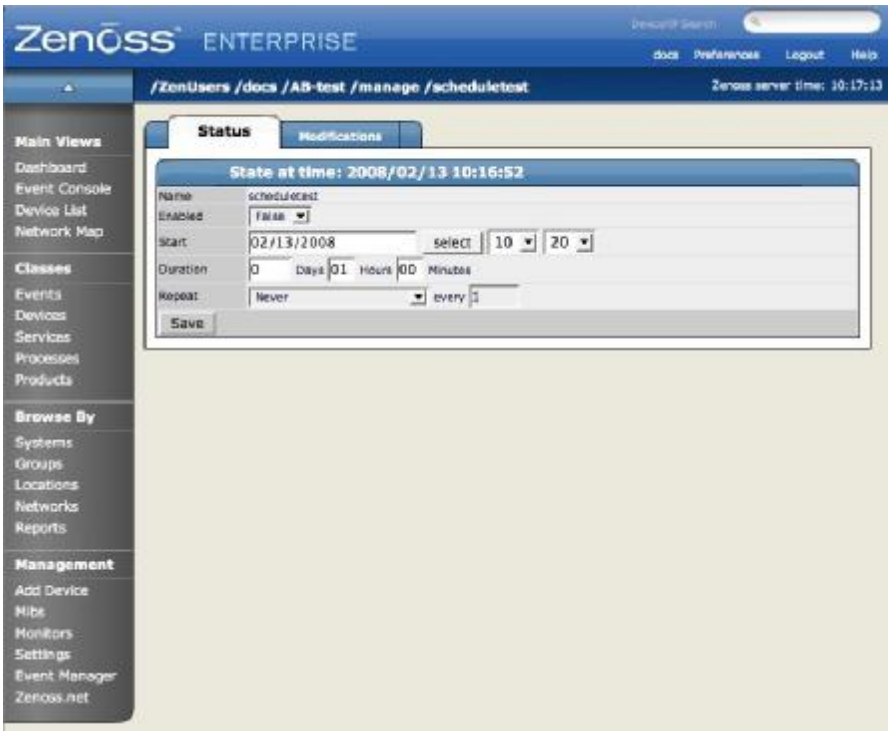
图 17.9. 添加活动周期对话框



3.在 ID 域中输入日程表的名称，然后点击 OK 按钮，之后该日程表将出现在活动周期列表中。

4.点击新添加到日程表的名称，对该日程表进行具体的设置。

图 17.10.告警规则- 时间表明细标签页



5. 如果您希望该告警仅在给定的时间段内出现有限的次数，那么需要将 Enabled 域设置为 True.

6. 在 Start 区域内，输入您希望告警开始的日期，然后点击 Select 按钮从日历控件中选择一个日期。

7. 在日期的右边，选择告警开始的小时和分钟。

8. 使用 Duration 区域来指定告警被监听的时间长度。

9. 如果您希望告警周期重复，您可以从弹出菜单中选择一个事件段，可供选择的选项有：

- Never
- Daily
- Every Weekday
- Weekly
- Monthly
- First Sunday of the Month

10. 选择一个重复的次数。

11. 点击 Save，您现在保存了创建一个新告警的所有设置。

3. Zenoss中告警消息的升级

您可以使用Zenoss提供的各种管理工具来创建一个消息体系。

3.1. 创建一个告警体系

您可以基于事件的状态以及根据告警规则产生的延迟来创建一个告警体系。举例来说，您向某人 A 发送了一个告警（比如说是第一级告警，on call tech），如果该事件在给定的时间内（比如一个小时内）没有被确认或者被清除（改变事件的状态），您可以再创建一个告警规则，将该事件通过电子邮件发送给告警体系中的另外一个人 B。

如果要创建这个体系，首先您必须为系统默认情况（DefaultCase）创建一个告警规则。该告警规则就是，在任何时间只要有任何级别的事件发生就首先向某人 A 发送告警。创建这个默认的告警规则与通常创建告警规则的方法毫无二致。设置延迟为 0 秒。在告警体系的下一个层次（某人 B），您可能要说：“好吧，如果 A 没有在一个小时之内确认或者清除该事件，就将告警发送至告警层次体系中的下一个人 B”。要在 Zenoss 中创建这样的告警层次体系，您必须为 B 再创建一个告警规则。有两种方法可以为 B 创建该告警规则，您可以（A）以 B 帐号登录，并为 B 账户创建一个告警规则，或者（B）您可以仅仅为 A 账户添加一些额外的信息（B 的电子邮件地址），该电子邮件地址将覆盖 A 账户的电子邮件地址，因此告警消息也将会发送到 B 的电子邮件帐户中。

对于第二条告警规则，要设置延迟时间为数秒，也就是说，也就是说，在事件进入系统但仍未被确认或者被清除之前，您希望等待多长时间。比如，我们已经决定了延迟时间为 1 个小时，以秒为单位计算，那么该告警规则的延迟应该被设置为 3600。

现在我们需要传递一个条件，该条件将防止所有的事件都产生该告警，甚至要防止在延迟期到期之前 A 已经确认或者清除的那些事件产生该告警。在添加过滤器（Add filter）域内，选择事件状态（Event State），然后选择组织告警规则在所有事件上执行的事件状态。在这种情况下，除非事件被确认或者被清除，否则事件就只有一个状态“New”，我们保留“New”的设置不变。这样该告警规则在事件的“New”状态超过 3600 秒后将向第二个人发送邮件。

在上文中，我们定义了一个只有两人的告警体系，但 Zenoss 允许您使用各种告警过滤条件来创建更为复杂的告警体系以及场景。设备的优先级是一个新属性，该属性能够很好的用于定义哪个人得到哪些告警，设备优先级就是用于限定并非所有人都能接收到告警。

4. 向告警规则添加延迟和日程表

在创建告警规则时，您可以使用延迟和日程表对告警规则进行限定。使用延迟将允许您指定，如果一个事件在某段时间内未被确认的话，系统将向告警体系中的第二个人发送邮件。=您可以通过过滤事件状态（'New' not acknowledged）以及添加一个延迟来达到上述目标。

为上文告警体系第一层中的工程师创建一个告警规则，该规则没有延迟，所以第一层的工程师可以立即发现告警并对告警进行确认处理。

1. 创建第二条告警规则(如上文所述场景)并启用该规则。.

2. 使用'where' 条件来设定该规则仅对那些尚未被确认的事件有效。

- Delay = 300 (in seconds, 5 minutes)
- In the Where area,
- Production State = Production
- Severity >= Error
- Event State = New

该规则的作用是，如果新产生的事件在系统中五分钟内未得到确认，就将该事件通过邮件发送给用户。

3. 点击消息（Message）标签页，在 Message (或 subject)域内输入如下内容：

```
[Zenoss-delayed] %(device)s %(summary)s
```

4. 在清除消息（Clear message）或主题 (Subject) 域内，输入如下内容：

```
[Zenoss-delayed] CLEAR: %(device)s %(clearOrEventSummary)s
```

5. 点击日程表（Schedule）标签页来编辑日程表。

6.在 ADD 域内输入新日程表的名称。

7. 新创建的日程表将出现在列表中，点击该日程表的名称，可对该日程表进行如下设置：

- Name = 新日程表的名称
- Enabled = True
- Start = 您希望规则开始生效的日期和时间 t
- Duration = 您希望规则保持有效的时间
- Repeat = 日程表重复的次数
- Every = 多少个周期重复一次

8. 点击 Save 按钮保存。

第18章.UI命令

1. 关于UI命令

用户命令允许用户在 Zenoss 内执行任意的 shell 命令。这些命令可以针对一个设备或者一组命令手工运行。用户命令在 Zenoss 服务器上执行，而不是在远程设备上（除非用户命令明确地使用了 SSH 连接到远程设备上）执行。用户命令可以以任何设备分类、系统、分组或者位置为基础进行定义。也可以从设置（Settings）页面的管理（Manage）标签页定义全局性的用户命令。

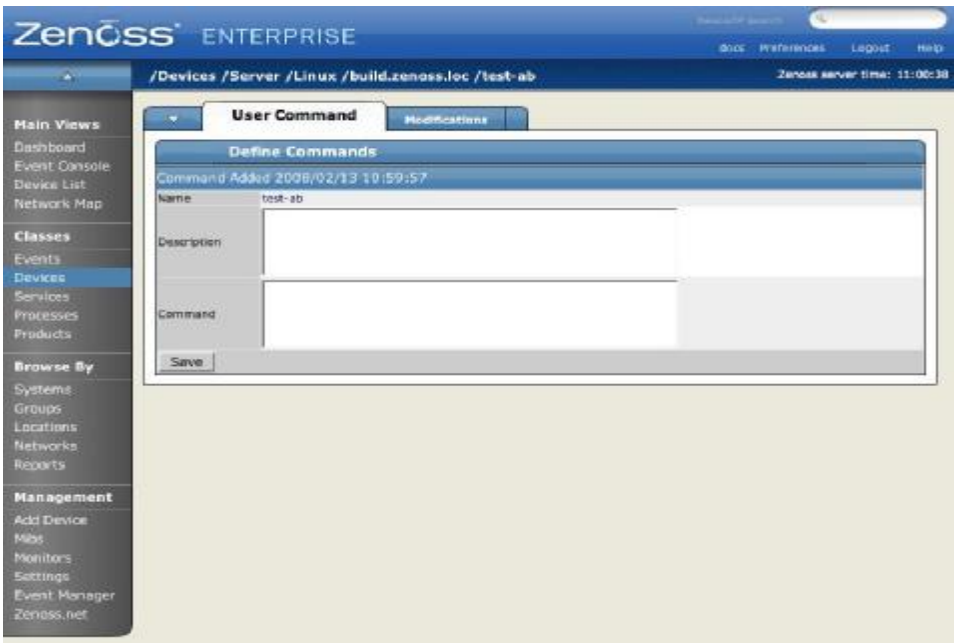
2. 定义 UI 命令

1. 要定义用户命令，可以在系统中任何一台设备的设备页菜单中，选择更多（More）>>管理（Administration）选项，系统显示管理标签页。
2. 定义命令（Define Commands）表菜单中，选择添加用户命令（Add User Command），系统弹出添加用户命令对话框，如下所示：

图 18.1. 添加用户命令对话框

3. 在Command Id 域中，添加用户命令的名称，点击OK按钮，系统将显示定义命令页面。

图 18.2.定义用户命令对话框



4. 在Description域中，输入该命令内容的简述。

5. 在Command域中，输入基于命令的TALES表达式。

6. 点击保存（Save）按钮，用户名命令被保存，此后可以在命令下拉菜单中选择该命令。

2.1. UI命令举例: Echo Command

下面，我们将详细介绍如何创建爱你一个echo UI命令，同时您还将看到TALES表达式的使用。

1. 从主界面左侧的导航菜单中，选择设置（Settings）——>管理（Manage）标签页。

2. 添加一个新的命令叫做“echoDevice”。

3. 回显设备的名称以及IP地址。

```
echo name = ${here/id} ip = ${here/manageIp}
```

在TALES表达式中，’here’是表达式要针对执行的目标对象。Zenoss中有一些TALES表达式有其它的变量，比如使用evt来代表事件，使用dev或者device来代表设备。有关TALES表达式的详细信息，请参阅附录。

4. 到设备页运行该命令。

5. 现在，重新回到该命令的编辑页面，向该命令中添加一些额外的信息。

```
echo name = ${here/id} ip = ${here/manageIp} hw = ${here/getHWProductName}
```

6. 现在，从一组设备上运行该命令，看看结果输出是什么。

3. 从Zenoss的Web界面上运行UI命令

Zenoss 允许用户命令通过用户界面运行。Zenoss已经有了一些内置的用户命令，比如ping和traceroute，这些命令可以基于单个设备运行，也可以基于一组设备运行。

1.导航到您想要运行命令的设备或者设备分组。

从设备页菜单上，选择运行命令（Run Commands），然后选择您要运行的命令，命令运行后，结果如下图所示：

图 18.3. 命令输出

4. 基于事件的自动运行的命令

Zenoss 允许您创建命令，且该命令根据进入到系统中的事件，针对一个设备或者一组设备自动运行。

1. 从Zenoss左侧的导航菜单中，选择事件经理（Event Manager），系统将显示事件经理的编辑标签页。

2. 点击命令（Commands ）标签页。

3. 输入新命令的名称，点击添加（Add）按钮，之后新命令将出现在命令列表中。

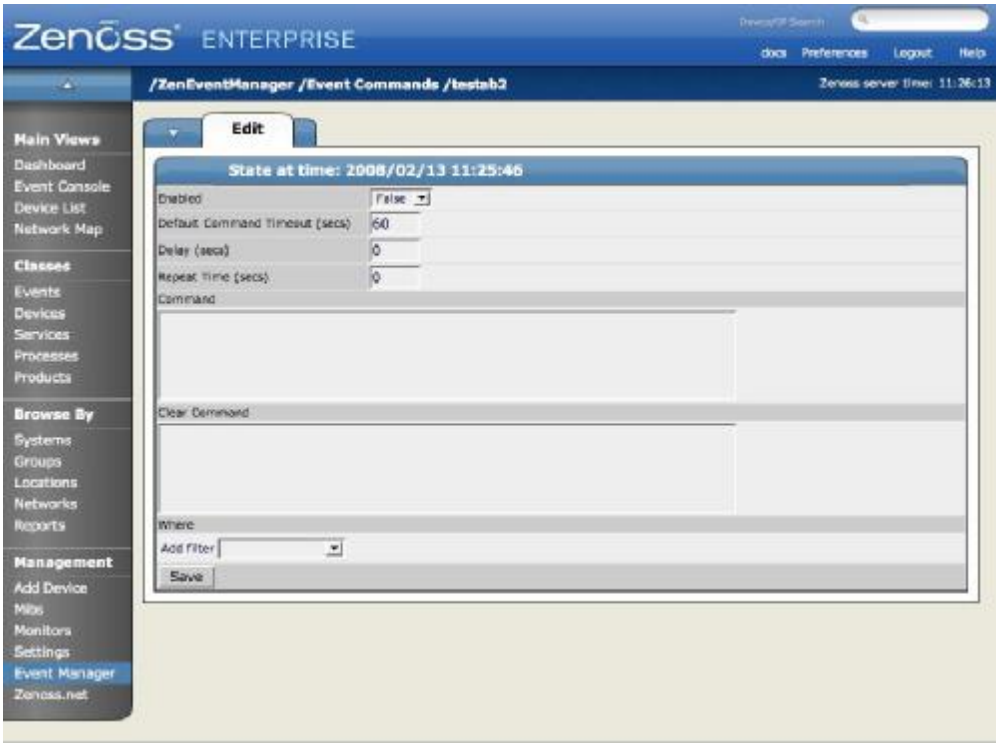
4. 在命令列表中点击新命令的名称，系统将在编辑命令标签页中显示新命令的各种属性，如下图18.4所示：

5. 将Enabled 设置为True启用该命令。

6. 您还可以改变命令的其它属性，比如默认超时（default timeout），系统的默认超时为60秒。

7. 您还可以改变命令的延迟，这个延迟是指，命令得以运行的条件被满足和实际命令运行之间的这段时间差。

图 18.4.编辑事件命令标签页



8. 在Command域内,输入条件满足时您想要运行的命令，这些命令使用TALES表达式。

9. 在清除命令（clear Command ）域内，输入一个命令，该命令在事件被清除后运行。注：上述命令均使用TALES 表达式书写，当Where条件满足时，命令将运行。命令的功能以及命令在什么地方运行都完全取决于命令脚本以及命令执行所在主机的权限。当然,清除命令（clear Command ）是一个可选项,它在某些情况下是非常有用的。比如,您在告警规则之外正运行着一个SMS Modem，而且您希望您的命令在Ping Down 时发送一个SMS短消息， 当Ping Down事件结束后， 您还希望从SMS Modem再发送一个短消息表明Ping已经正常，此时就可以使用清除命令（clear Command ）。

10. 使用“Where”设定命令可以执行的条件。

11. 在添加过滤条件（Add Filter）下拉菜单中选择命令的过滤条件,可供选择的过滤条件有：

- Agent
- Count
- Device Class
- Device Priority
- Event Class Key
- Facility
- Manage
- ntevid
- Priority

- Component
- Device
- Device Groups
- Event Class
- Event State
- IP Address
- Message
- OwnerId

如果有必要，您可以添加任意多的过滤条件，这样可以进一步对命令的执行进行限制。

13. 一旦完成了上述设置，点击保存（Save）按钮，该命令将被保存入系统，当一定条件满足时，该命令将自动执行。

第19章.生产状态与维护窗口

1. 关于生产状态和维护窗口

Zenoss拥有支持“维护窗口”的能力，换句话说，在维护窗口这段时间内，用户可以对设备的监视或者告警规则进行修改。如果一个设备有一系列的规则来控制其监视以及告警的显示，那么设备的状态可以被称为“生产状态”（Production States）。如果生产状态有临时的改变，且设备的原始生产状态有了小幅调整，我们即将其称之为一个维护窗口。

2. 生产状态

生产状态是Zenoss中的一个重要概念。它决定了设备是否被监控，同时生产状态还被用于控制事件系统的多个元素，比如一个事件是否产生一个远程告警（电子邮件或寻呼）。通常，设备都已“预生产”（Pre-Production）状态开始它们的生命。在这种状态下，设备仅受到默认的系统监视，此时由该设备产生的远程告警或者事件将不显示在仪表盘上。一旦设备进入生产（Production）状态，由该设备产生的远程告警将被发送。如果您需要在设备上进行一些维护工作，您可以暂时将设备的生产状态置为“维护”状态（Maintenance），此时系统将暂时阻止远程告警的产生。

有三个因素影响设备生产状态的定义：

1. 设备是否正被监视。
2. 您是否需要设备发生告警。
3. 设备是否显示在仪表盘上。

可用的生产状态是以上三个因素的组合：

- Production – 您希望设备被监视，生成设备告警，同时设备信息显示在仪表盘上。
- Pre-production –您仅仅希望监视设备，但不希望产生告警，同时设备也不显示在仪表盘上。
- Test – 您希望监视设备并同时希望设备产生告警，但不希望将设备信息显示在仪表盘上。
- Maintenance – 您希望对设备进行监视并希望开始采集设备数据，而且无论设备信息是否显示在仪表盘上，都不希望设备有告警发生。
- Decommissioned – 不对设备进行监视，设备信息不出现在仪表盘上，同时也不产生告警。

2.1. 定义设备的生产状态

如果要设定一个设备或者一组设备的生产状态，可以到该设备或者该设备分组的编辑（Edit）标签页，在生产状态下拉列表中选择您想要的生产状态。当您向系统中添加一个新设备后，设备默认的生产状态将会是“生产”（Production）。如果您改变了设备层次树中某一层的生产状态设置，该层及其之下所有层次的设备的生产状态将随之改变，直至您在该层之下重新对生产状态

进行设置。

3. 维护窗口

Zenoss 的维护窗口允许一个设备或者一个系统、分组护或者位置中的所有设备进行定期的生产状态改变。维护窗口在这些对象的管理标签页上进行定义。每个维护窗口都拥有开始时间、持续时间、重复周期以及生产状态的起始和结束等属性。一个典型的维护窗口的例子是，开始时将设备从生产状态改为维护状体，结束时再将设备改回生产状态。

维护窗口的“启动生产状态”（Start Production State）是指，当设备正处于监视状态下时，此时维护窗口开始或者“打开”。举例来说，如果您的设备正在以“生产”状态运行（意味着您目前正在对该设备进行正常的监视，且该设备能够正常地产生告警），此时维护窗口的开放时间到来，那么您设备的生产状态经会变成维护窗口的启动生产状态。

再举例来说，如果启动生产状态被设置为“维护”（Maintenance），那么这意味着您还希望设备处于监视状态之下，且您希望继续采集设备的数据，但您不希望设备再产生告警或者设备信息再显示在仪表盘上。在这种情况下，您可以重启设备，或者修改设备的配置信息，而这通常都会引发告警的产生，但此时设备将不会发送告警。

您可以定期安排设备的维护窗口，或者在您需要改变设备配置时手工改变设备的生产状态。当维护窗口关闭时，设备的维护窗口将变成“结束生产状态”（End Production State）。您可以为维护窗口定义结束生产状态。这里的“结束生产状态”指的就是，维护窗口结束后，设备要变回的生产状态，而“启动生产状态”就是指维护窗口打开时，设备将要变成的状态。

如果是我正在建立一个维护窗口的话，我将会这样定义维护窗口，那就是，当维护窗口打开时间到来时，我希望“启动生产状态”为“维护”状态，当维护窗口关闭时，我希望“结束生产状态”是“生产”状态，这意味着设备重新又回到系统监视之下，并开始正常发送告警。这个过程中，由于您对设备进行重启或者其它操作而产生的告警将不再产生。

3.1. 创建并使用维护窗口

您可以为单独的一个设备创建一个维护窗口，同时也可以为设备分类层次树中的任意一个或者多个设备创建维护窗口。要创建一个新的维护窗口，必须采取以下步骤：

1. 导航至您希望建立维护窗口的设备或者设备分组，打开页菜单，选择更多（More）选项，然后选择管理（Administration），系统将显示管理页面。
2. 从维护表菜单中，选择添加维护窗口（Add Maint Window），系统将显示维护窗口页。
3. 在ID域中输入维护窗口的名称，并点击OK按钮，该维护窗口的名称将出现在维护窗口列表中。
4. 要定义维护窗口，可以点一下维护窗口的名称，系统将显示维护窗口状态标签页。
5. 定义维护窗口的属性：

- Name - 维护窗口的名称
- Enabled - 该维护窗口是否激活
- Start - 维护窗口激活的起始时间
- Duration - 维护窗口的有效期
- Start Production State - 维护窗口的开始时间。
- Stop Production State - 维护窗口的结束时间。

6. 点击保存（Save）按钮。

第21章. Zenoss常用管理

1. 使用Zenoss的命令行进行工作

当在命令行下进行Zenoss的管理工作时，您总会想以用户”Zenoss”来登录。如果您是以root用户登录，那么您必须通过下面这个命令转为”zenoss”用户。

```
su - zenoss
```

2. 最小化Zenoss - ZEO 与 Zope

Zenoss的用户界面可以在不运行其它Zenoss进程的情况下单独运行，在这种情况下，系统中仅有ZEO和ZOPE在运行。这种模式对于Zenoss排障是很有帮助的。

1. 停止所有的Zenoss进程。

```
$ zenoss stop
```

2. 启动zope对象数据库。

```
$ zeoctl start
```

3. 启动Zope

```
$ zopectl start
```

到Web界面下确认您可以访问仪表盘。

4. 启动其它Zenoss进程。

3.检查 Zenoss版本

您可以使用版本（Version）标签页来检查Zenoss及其所有相关组件的版本。操作步骤是，到左侧的导航菜单中，选择设置（Settings），然后点击版本（Versions）标签页。该标签页将显示以下组件的版本：

- Zenoss
- Zope
- Database
- Twisted
- OS
- Python
- RRD• SNMP

该页面还显示Zenoss的运行时间。

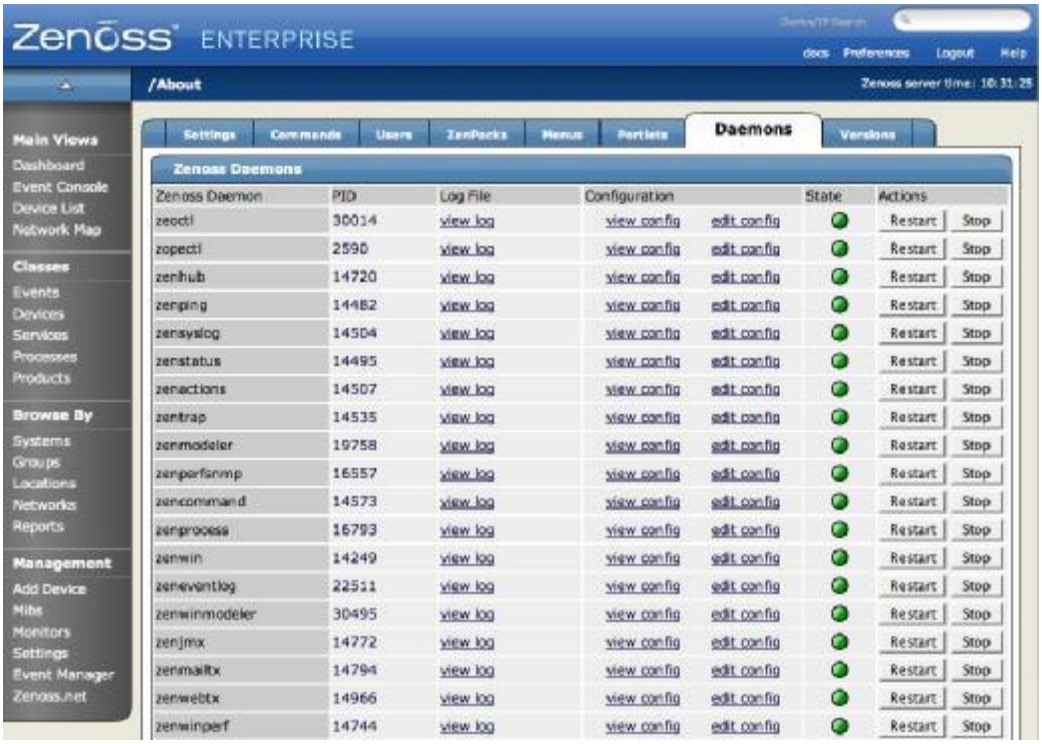
4. 检查Zenoss更新

同时，在Zenoss的版本标签页中还可以检查Zenoss的软件更新。您可以让Zenoss每天检查一下是否有软件更新，或者是点一下现在就检查Zenoss版本（Check Zenoss Version Now）按钮来检查Zenoss的软件更新。

5. Zenoss进程的停止和启动

您可以使用Zenoss的管理控制台来启动或者停止Zenoss进程，同时还可以检查Zenoss进程的状态。操作步骤是，从左侧导航菜单中选择设置（Settings），然后点击进程（Daemons）标签页。

图 21.1. 设置页面 –进程页标签



该页显示了一个Zenoss进程的列表，在列表中，我们可以看到Zenoss进程的名称、PID、状态指示器以及启动和停止按钮各一个。如果一个进程停止了，可以点击Restart按钮来重启该进程。

6. Zenoss 进程命令与选项

所有的Zenoss进程都共享一些命令。这些命令可以在安装Zenoss的主机的命令行下运行。每个Zenoss进程都有以下一些命令：

- run – 启动进程但并不把进程放到后台执行，该命令对调试非常有用。
- start – 启动一个进程并在后台运行，进程启动后与shell分离。

- stop – 停止一个进程。.
- restart – 停止并启动进程。有时启动命令会在进程停止之前运行，如果出现了这种情况，重新运行一下该命令就好。
- status – 检查进程的状态，使用该参数将打印当前进程的进程号。
- help – 显示该进程的所有选项列表。

6.1. 配置Zenoss进程

任何Zenoss进程都可以被配置，方法是在一个名为$ZENHOME/etc/DAEMONNAME.conf 的配置文件中，添加一个键/值对。合法的键都是Zenoss进程命令的选项，这些合法的键可以通过进程的help参数获得。

6.2. 所有进程的通用选项

命令	说明
--version	显示程序的版本号并退出
-h,	显示帮助消息并退出
-vLOGSEVERITY,--logseverity=LOGSEVERITY	日志等级门限
--logpath=LOGPATH	覆写默认的日志路径
-CCONFIGFILE,--	配置文件
--uid=UID	运行时使用的用户，默认为:zenoss
-c,	Cycle continuously on cycleInterval from zope
-D,	变成Unix进程
--host=HOST	Zeo server的主机名
--port=PORT	Zeo server的端口
-RDATAROOT,	数据加载的根对象(比如. /zport/dmd)
--cachesize=CACHESIZE	内存中缓存的大小，默认为: 1000
--pcachename=PCACHENAME	永久缓存文件的名称，默认为:None
--pcachedir=PCACHEDIR	永久缓存文件的路径

6.3. zenhub选项

命令	说明
-x	XML RPC调用要监听的端口
--pbport=PBPORT	Pb端口
--passwd=PASSWORDFILE	存储口令的文件

6.4. zenmodeler Options

命令	说明
--debug	进程不生成线程
--parallel=PARALLEL	并行采集设备的数量（同时能从几台设备上采集数据）

--cycletime=CYCLETIME	采集数据的周期
命令	**说明**
--ignore=IGNOREPLUGINS	用逗号分隔的忽略的采集插件列表
--collect=COLLECTPLUGINS	用逗号分隔的要使用的采集插件列表
-pPATH, --path=PATH	采集数据的开始路径，比如/NetworkDevices
-dDEVICE, --device=DEVICE fully qualified device name ie: www.zenoss.com	完全限定性域名，比如: www.zenoss.com
-aCOLLAGE, --collage=COLLAGE	do not collect from devices whose collect date is within this many minutes
--writetries=WRITETRIES	发现有读冲突后，尝试去写的次数
-F, --force	强制采集配置数据 (even without change to the device)
--portscantimeout=PORTSCANTIMEOUT	端口扫描连接失败时要等待的时间
-uUSERNAME, --user=USERNAME	登录用户名
-PPASSWORD, --password=PASSWORD	登录口令
-tLOGINTRIES, --loginTries=LOGINTRIES	尝试登录的次数
-LLOGINTIMEOUT, --loginTimeout=LOGIN-TIMEOUT	登陆超时
-TCOMMANDTIMEOUT, --commandTimeout=COM-MANDTIMEOUT	发出命令时的超时
-KKEYPATH, --keyPath=KEYPATH	寻找Keys时要使用的路径
-sSEARCHPATH, --searchPath=SEARCHPATH	用于寻找命令时的路径
-eEXISTENCETEST, --existenceTest=EXISTEN-CETEST	如何检查命令
-rPROMPTTIMEOUT, --promptTimeout=PROMPT-TIMEOUT	发现命令提示符时的超时
-xLOGINREGEX, --loginRegex=LOGINREGEX	用于找到登录提示符的正则表达式
-wPASSWORDREGEX, --passwordRegex=PASS-WORDREGEX	用于找到口令提示符的正则表达式
--enable	在思科设备上进入enable模式
--termlen	在思科设备上输入send terminal length 0
--enablePause=ENABLEPAUSE	发送enable命令之前要等待的时间
--enableRegex=ENABLEREGEX	找到enable口令提示符的正则表达式

NOTE: --ignore and --collect are mutually exclusive

6.5. zenperfsnmp Options

命令	说明
-zZOPEURL, --zopeurl=ZOPEURL	性能配置服务器的XMLRPC url路径
-uZOPEUSERNAME, --zopeusername=ZOPEUSER-NAME	登录 zope server的用户名

--zopepassword=ZOPEPASSWORD	登录 zope server的口令
--zem=ZEM	到一个ZenEventManager实例的XMLRPC路径
-dDEVICE, --device=DEVICE	给监视器指定一个特定的设备
--monitor=MONITOR	为监视器配置指定一个名称

6.6. zenperfxmlrpc Options

命令	说明
-zZOPEURL, --zopeurl=ZOPEURL	性能配置服务器的XMLRPC url路径
-uZOPEUSERNAME, --zopeusername=ZOPEUSER- NAME	登录 zope server的用户名
--zopepassword=ZOPEPASSWORD	登录 zope server的口令
--zem=ZEM	到一个ZenEventManager实例的XMLRPC路径
-dDEVICE, --device=DEVICE	给监视器指定一个特定的设备
--monitor=MONITOR	为监视器配置指定一个名称

6.7. zenProcess Options

命令	说明
-zZOPEURL, --zopeurl=ZOPEURL	性能配置服务器的XMLRPC url路径
-uZOPEUSERNAME, --zopeusername=ZOPEUSER-NAME	登录 zope server的用户名
--zopepassword=ZOPEPASSWORD	登录 zope server的口令
--zem=ZEM	到一个ZenEventManager实例的XMLRPC路径
-dDEVICE, --device=DEVICE	给监视器指定一个特定的设备
--monitor=MONITOR	为监视器配置指定一个名称

6.8. zenping Options

命令	说明
--configpath=CONFIGPATH	到监视器配置文件的路径，比如：/Monitors/StatusMonitors/localhost
--name=NAME	在dmd中寻找记录时使用的名字，默认为fqdn名字
--test	以测试模式运行:并不真正执行Ping，但要从/tmp/testping中读取IP地址列表。

6.9. zensyslog 选项

命令	说明
--statcycle=STATCYCLE	写入统计数据的周期
--dmdpath=DMDPATH	到dmd 的ZOPE路径 /zport/dmd
--parsehost	试着分析syslog头中的主机名部分
--stats	每两秒钟向log打印一次统计数据
--logorig	向日志写入原始消息
--debug	调试模式无线程
--minpriority=MINPRIORITY	Syslog接受的最低优先级
--heartbeat=HEARTBEAT	心跳周期，单位以秒计算
--syslogport=SYSLOGPORT	用于syslog 事件的端口号

6.10. zenstatus 选项

命令	说明
--configpath=CONFIGPATH	到监视器配置的路径，比如: /Devices/Server
--parallel=PARALLEL	并行采集设备的数量（同时对几台设备采集数据）
--cycletime=CYCLETIME	检查事件的周期，单位为秒

6.11. zenactions 选项

命令	说明
--cycletime=CYCLETIME	检查事件的周期，单位为秒
--fromaddr=FROMADDR	邮件从哪里发送的地址
--zopeurl=ZOPEURL	到zope server根的http路径

6.12. zentrap 选项

命令	说明
--statcycle=STATCYCLE	写入统计数据的周期，单位为秒
-tTRAPPORT, --trapport=TRAPPORT	监听陷阱的端口号

6.13. zencommand 选项

命令	说明
-zZOPEURL, --zopeurl=ZOPEURL	性能配置服务器的XMLRPC url路径
-uZOPEUSERNAME,	登陆到zope server的用户名

--zopeusername=ZOPEUSER-NAME	
--zopepassword=ZOPEPASSWORD	登陆到zope server的口令
--zem=ZEM	到一个ZenEventManager 实例的XMLRPC路径
-dDEVICE, --device=DEVICE	给监视器指定一个设备
--monitor=MONITOR	指定一个监视器配置的名称
--parallel=PARALLEL	并行采集设备的数量（同时对几台设备采集数据）

7. Zenoss进程排障

当Zenoss出现问题时，Zenoss将把堆栈跟踪消息写入到日志文件。这些日志对于排查Zenoss故障是非常重要的。下面我们将介绍如何生成一个堆栈跟踪：

1. 以zenoss的一个用户身份登录后，停止所有的zenoss进程。
2. 停止MySQL数据库。

```
$ sudo service mysqld stop
```

3. 启动Zope对象数据库。

```
$ zeoctl start
```

4. 在前台以调试模式运行zenperfsnmp。

```
$ zenperfsnmp run
```

5.注意，此时有堆栈跟踪消息说该进程无法连接到zenoss。

6. 启动Zope.

```
$ zopectl start
```

7.到仪表盘查看错误信息“Lost connection to Zenoss”。出现该提示消息的原因是多样的。

8. 因此，转到日志文件$ZENHOME/log/event.log寻找更多的相关信息。

9. 重新启动mysqld。

```
$ sudo service mysqld start
```

10. 重新启动Zenoss.

```
$ zenoss start
```

11. 查看所有zenoss日志文件的末尾，看是否有错误信息出现。

```
$ tail $ZENHOME/log/*.log | less
```

8. Linux 环境下的自动启动

Zenoss能够被bin/zenoss下的脚本完全控制，要在Linux环境下自动启动Zenoss，您必须把Zenoss链接到/etc/rc.d/init.d目录：

```
$ ln -s $ZENHOME/bin/zenoss /etc/rc.d/init.d
```

注意，一定要将$ZENHOME这个变量添加到root环境，或者添加到Zenoss脚本中去。

9. Zenoss使用远程MySQL实例

如果您想为Zenoss使用远程MySQL数据库，您可以在远程主机上安装该MySQL数据库实例，安装方法与在本地主机上如出一辙。然后登录到Zenoss管理控制台上进行如下操作：

1．首先改变mysql数据库的地址。从Zenoss仪表盘中，在左侧的导航菜单中选择事件经理（Event Manager）。

2.在随后出现的ZenEventManager 页面中，改变MySQL数据库的主机地址。

3．重新启动Zenoss，您刚刚输入的MySQL数据库即成为Zenoss的默认数据库。

10. 向Zenoss注册MIB

Zenoss的MIB加载器叫做zenmib，要向Zenoss中加载MIB，首先将MIB文件拷贝到$ZENHOME/share/mibs/site，然后在命令行下运行该命令：

```
zenmib run mibfile
```

10.1. 解决MIB依赖性

如果你已经在系统中加载了aaa.mib，当你尝试再向系统加载bbb.mib时，您可能会碰到下面的错误消息：INFO:zen.zenmib:Unable to find a file providing the MIB AAA-MIB

要解决依赖性问题，采用下面这个加载方法：

```
zenmib run aaa.mib bbb.mib
```

第22章.备份、恢复与维护

1. 备份与恢复

Zenoss本身提供一套工具，在工具的帮助下，您可以从Zenoss的一个安装实例中备份配置信息和数据，在必要时，还可以在工具的帮助下恢复这些配置信息和数据。该功能在对您的安装实例进行周期性快照（出于备份目的）时是非常有用的，同时，还可以使用系统提供的备份和恢复工具，将您的数据从一个Zenoss安装数据转移到另外一个安装实例，或者在执行一个全新安装后恢复您的设置。Zenoss 的备份与恢复要涉及到以下内容：

- 位于mysql中的整个事件数据库。
- Zope数据库，该数据库包括了所有的设备、用户、事件映射等等。
- $ZENHOME/etc 目录,该目录包含了zenoss进程的配置文件。
- $ZENHOME/perf 目录，该目录包含了性能数据。

下面，我们将详细描述备份和恢复的脚本及控制脚本行为的选项。Zenbackup的典型用法如下所示：

```
> zenbackup --save-mysql-access --file=BACKUPFILEPATH
```

Zenrestore的典型用法如下所示:

```
> zenrestore --file=BACKUPFILEPATH
```

以下是一些备份/恢复操作的建议：

- 如果您有可用的磁盘空间，在进行任何备份或者恢复操作之前,打包（tar）并压缩（zip）$ZENHOME目录。这样做的好处是，一旦备份或者恢复出现问题，您还有一个恢复的机会。
- 在执行恢复操作之前，要确保Zenoss，包括所有的进程已经停止。
- 使用备份恢复工具从一个新版本的zenoss恢复到老版本的zenoss是一种极不明智的选择,应当避免这种操作。
- 您使用旧版本的zenoss数据在新版本的zenoss平台上执行恢复操作后,恢复操作后一定要执行zenmigrate 。
- 如果使用的并非是最初的备份文件进行恢复操作时,要确保$ZENHOME/etc/*.conf 文件适用于您恢复后的新环境。

1.1. 详细备份操作

用于备份的脚本是$ZENHOME/bin/zenbackup。如果zenoss正在运行，您可以不带任何参数运行Zenbackup,备份文件将被放在$ZENHOME/backups目录下。zenbackup --help 将列出该命令的所有可用参数，以下是一些常用的参数：

--dbname：这是zenoss用于保存事件数据的mysql数据库的名字。默认情况下为"zenoss"。但这个数据库的明在在安装时是可由用户指定的。用户可以在事件经理（Event Manager）页面中看

到该参数的值。如果您在安装时不指定该参数，zenbackup将会试着从zeo中读取该参数的值，除非您使用了 --dont-fetch-args参数。

--dbuser, --dbpassword：这两个参数是用来访问事件数据库的mysql用户名/口令。如果您不指定--dbuser 或 --db-password 参数，zenbackup 将会尝试着从zeo中读取这两个参数的值，除非您使用了--dont-fetch-args参数。

--dont-fetch-args：该参数用来命令zenbackup进程不要从zeo中读取dbname, dbuser和 dbpassword 的值。

--file=FILE:该参数指定备份文件的位置,默认情况下,备份文件被命名为zenoss_<DATE>.tgz ，且该文件被放置在$ZENHOME/backups目录下。

--stdout：该参数告诉zenbackup进程，将备份信息发送到stdout而不是发送到一个文件，不能与—verbose参数同时使用。

--save-mysql-access：该参数告诉zenbackup进程，将dbname, dbuser 和 dbpassword 作为备份文件的一部分进行保存。zenrestore 在执行恢复操作时可以利用上述信息。使用该参数时一定要谨慎，因为您的备份文件中将包含mysql的用户名和口令信息。

--no-eventsdb：执行备份操作时不备份事件数据库。

-v, --verbose：打印进度消息，不能与—stdout参数一起使用。

1.2. 详细恢复操作

恢复zenoss的脚本叫做$ZENHOME/zenrestore，要确保执行恢复操作前zenoss已经停止。如果您在备份操作时使用了--save-mysql-access 参数，那么在恢复操作时您只需要使用--file 参数来指定用于恢复的备份文件。否则您还需要指定dbname, dbuser 和 dbpassword 这些参数。

--file：使用zenbackup 创建的备份文件，您必须指定—file或—dir参数其中之一。

--dir：备份文件的解压缩目录，您必须指定—file或—dir参数其中之一。

--dbname:Zenoss 用于保存事件数据的mysql数据库。在zenrestore运行之前该数据库必须存在。如果该数据库中有zenoss的相关库表，那么在zenrestore恢复备份表或者数据之前，这些已有的库表将被清除。如果您在恢复时使用了一个与备份不同的dbname，那么在恢复操作之后，您必须在事件经理页面中重新设定数据库名称。

--dbuser, --dbpassword：这两个参数是用来访问事件数据库的mysql用户名/口令。如果您不指

定--dbuser 或 --db-password 参数,zenrestore将会试着使用保存在备份文件中的用户名和口令，前提是，在创建备份文件时使用了--save-mysql-access 参数。

--no-eventsdb：不恢复mysql事件数据库。如果备份文件中不包含mysql事件数据，即便您不使用--no-eventsdb参数，zenrestore也不会修改您的事件数据库。

-v, --verbose：打印进度消息。

1.3. 周期性备份

对您的zenoss数据和配置进行周期性备份是一个非常好的做法。$ZENHOME/bin/zenbackup能够创建您的zenoss备份，该备份中包括了您的zeo数据库（设备等）、RRD文件（性能数据)、MySQL库表（事件)以及您的Zenoss配置文件。

1.3.1. 压缩 ZEO 数据库

我们需要对Zeo数据库进行周期性的压缩以节省磁盘空间，做法是，创建一个cron job来每周运行一次以下命令：

```
$ZENHOME/bin/zeopack.py -p 8100
```

1.3.2. Log Rotate Script

如果您的系统使用日志滚动（LogRotate）的方法来管理日志文件的话，您可以将下面的脚本放在files put the following in /etc/logrotate.d/zenoss 中来管理Zenoss的日志文件：

```
/usr/local/zenoss/log/*.log {
weekly
rotate 2
copytruncate
}
```

1.3.3. Backing up the MySQL Event Backend

MySQL的备份应该遵循MySQL操作手册。