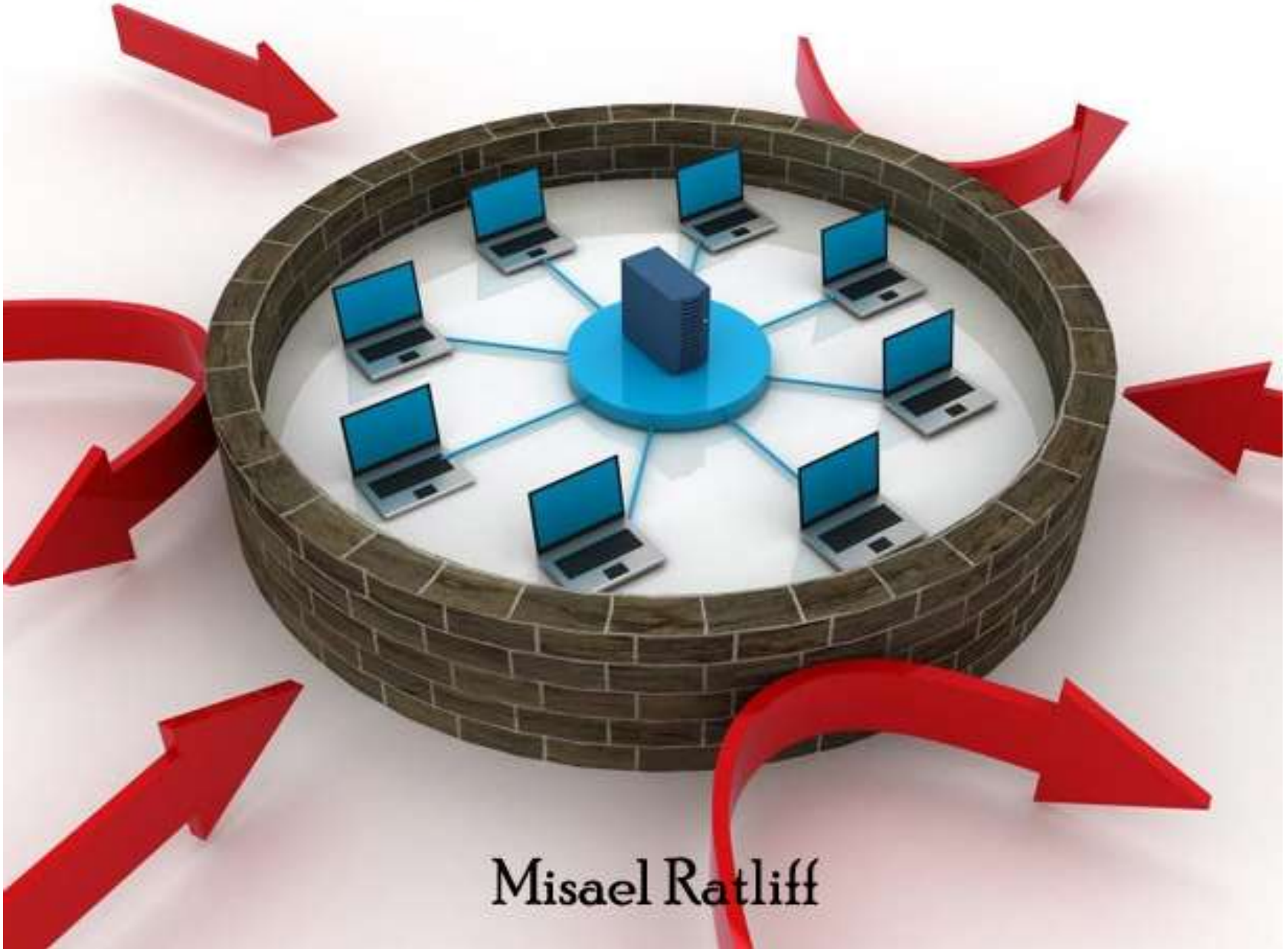
Network Management

(Concepts, Tools & Products)



Misael Ratliff

First Edition, 2012

ISBN 978-81-323-1754-8

Published by:
Learning Press
4735/22 Prakashdeep Bldg,
Ansari Road, Darya Ganj,
Delhi - 110002
Email: info@wtbooks.com

Table of Contents

Introduction

Chapter 1 - Autonomic Networking

Chapter 2 - FCAPS

Chapter 3 - Simple Network Management Protocol and Java Management Extensions

Chapter 4 - Load Balancing (Computing) and Network Administrator

Chapter 5 - Management Information Base and NETCONF

Chapter 6 - NetCrunch and Netcat

Chapter 7 - Network Operations Center and OpenNMS

Chapter 8 - Opsi

Chapter 9 - RRDtool, Rsyslog and Syslog-ng

Chapter 10 - Verax NMS and Zabbix

Chapter 11 - Zenoss and Zentyal

Introduction

Network management refers to the activities, methods, procedures, and tools that pertain to the operation, administration, maintenance, and provisioning of networked systems.

- Operation deals with keeping the network (and the services that the network provides) up and running smoothly. It includes monitoring the network to spot problems as soon as possible, ideally before users are affected.
- Administration deals with keeping track of resources in the network and how they are assigned. It includes all the "housekeeping" that is necessary to keep the network under control.
- Maintenance is concerned with performing repairs and upgrades—for example, when equipment must be replaced, when a router needs a patch for an operating system image, when a new switch is added to a network. Maintenance also involves corrective and preventive measures to make the managed network run "better", such as adjusting device configuration parameters.
- Provisioning is concerned with configuring resources in the network to support a given service. For example, this might include setting up the network so that a new customer can receive voice service.

A common way of characterizing network management functions is FCAPS—Fault, Configuration, Accounting, Performance and Security.

Functions that are performed as part of network management accordingly include controlling, planning, allocating, deploying, coordinating, and monitoring the resources of a network, network planning, frequency allocation, predetermined traffic routing to support load balancing, cryptographic key distribution authorization, configuration management, fault management, security management, performance management, bandwidth management, Route analytics and accounting management.

Data for network management is collected through several mechanisms, including agents installed on infrastructure, synthetic monitoring that simulates transactions, logs of activity, sniffers and real user monitoring. In the past network management mainly consisted of monitoring whether devices were up or down; today performance management has become a crucial part of the IT team's role which brings about a host of challenges—especially for global organizations.

Note: Network management does not include user terminal equipment.

Technologies

A small number of accessories methods exist to support network and network device management. Access methods include the SNMP, command-line interface (CLIs), custom XML, CMIP, Windows Management Instrumentation (WMI), Transaction Language 1, CORBA, NETCONF, and the Java Management Extensions (JMX).

Schemas include the WBEM, the Common Information Model, and MTOSI amongst others.

Medical Service Providers provide a niche marketing utility for managed service providers; as HIPAA legislation consistently increases demands for knowledgeable providers. Medical Service Providers are liable for the protection of their clients confidential information, including in an electronic realm. This liability creates a significant need for managed service providers who can provide secure infrastructure for transportation of medical data.

Chapter 1

Autonomic Networking

Autonomic Networking follows the concept of Autonomic Computing, an initiative started by IBM in 2001. Its ultimate aim is to create self-managing networks to overcome the rapidly growing complexity of the Internet and other networks and to enable their further growth, far beyond the size of today.

Increasing size and complexity

The ever-growing management complexity of the Internet caused by its rapid growth is seen by some experts as a major problem that limits its usability in the future.

What's more, increasingly popular smartphones, PDAs, networked audio and video equipment, and game consoles need to be interconnected. Pervasive Computing not only adds features, but also burdens existing networking infrastructure with more and more tasks that sooner or later will not be manageable by human intervention alone.

Another important aspect is the price of manually controlling huge numbers of vitally important devices of current network infrastructures.

Autonomic nervous system

The autonomic nervous system (ANS) is the part of the nervous system of the higher life forms that is not consciously controlled. It regulates bodily functions and the activity of specific organs. As proposed by IBM, future communication systems might be designed in a similar way to the ANS.

Components of autonomic networking

As autonomics conceptually derives from biological entities such as the human autonomic nervous system, each of the areas can be metaphorically related to functional and structural aspects of a living being. In the human body, the autonomic system facilitates and regulates a variety of functions including respiration, blood pressure and circulation, and emotive response. The autonomic nervous system is the interconnecting fabric that supports feedback loops between internal states and various sources by which internal and external conditions are monitored.

Autognostics

Autognostics includes a range of self-discovery, awareness, and analysis capabilities that provide the autonomic system with a view on high-level state. In metaphor, this represents the perceptual sub-systems that gather, analyze, and report on internal and external states and conditions – for example, this might be viewed as the eyes, visual cortex and perceptual organs of the system. Autognostics, or literally "self-knowledge", provides the autonomic system with a basis for response and validation.

A rich autognostic capability may include many different "perceptual senses". For example, the human body gathers information via the usual five senses, the so-called sixth sense of proprioception (sense of body position and orientation), and through emotive states that represent the gross wellness of the body. As conditions and states change, they are detected by the sensory monitors and provide the basis for adaptation of related systems. Implicit in such a system are imbedded models of both internal and external environments such that relative value can be assigned to any perceived state - perceived physical threat (e.g. a snake) can result in rapid shallow breathing related to fight-flight response, a phylogenetically effective model of interaction with recognizable threats.

In the case of autonomic networking, the state of the network may be defined by inputs from:

- individual network elements such as switches and network interfaces including
  - specification and configuration
  - historical records and current state
- traffic flows
- end-hosts
- application performance data
- logical diagrams and design specifications

Most of these sources represent relatively raw and unprocessed views that have limited relevance. Post-processing and various forms of analysis must be applied to generate meaningful measurements and assessments against which current state can be derived.

The autognostic system interoperates with:

- configuration management - to control network elements and interfaces
- policy management - to define performance objectives and constraints
- autodefense - to identify attacks and accommodate the impact of defensive responses

Configuration management

Configuration management is responsible for the interaction with network elements and interfaces. It includes an accounting capability with historical perspective that provides

for the tracking of configurations over time, with respect to various circumstances. In the biological metaphor, these are the hands and, to some degree, the memory of the autonomic system.

On a network, remediation and provisioning are applied via configuration setting of specific devices. Implementation affecting access and selective performance with respect to role and relationship are also applied. Almost all the "actions" that are currently taken by human engineers fall under this area. With only a few exceptions, interfaces are set by hand, or by extension of the hand, through automated scripts.

Implicit in the configuration process is the maintenance of a dynamic population of devices under management, a historical record of changes and the directives which invoked change. Typical to many accounting functions, configuration management should be capable of operating on devices and then rolling back changes to recover previous configurations. Where change may lead to unrecoverable states, the sub-system should be able to qualify the consequences of changes prior to issuing them.

As directives for change must originate from other sub-systems, the shared language for such directives must be abstracted from the details of the devices involved. The configuration management sub-system must be able to translate unambiguously between directives and hard actions or to be able to signal the need for further detail on a directive. An inferential capacity may be appropriate to support sufficient flexibility (i.e. configuration never takes place because there is no unique one-to-one mapping between directive and configuration settings). Where standards are not sufficient, a learning capacity may also be required to acquire new knowledge of devices and their configuration.

Configuration management interoperates with all of the other sub-systems including:

- autognostics - receives direction for and validation of changes
- policy management - implements policy models through mapping to underlying resources
- security - applies access and authorization constraints for particular policy targets
- autodefense - receives direction for changes

Policy management

Policy management includes policy specification, deployment, reasoning over policies, updating and maintaining policies, and enforcement. Policy management is required for:

- constraining different kinds of behavior including security, privacy, resource access, and collaboration
- configuration management
- describing business processes and defining performance
- defining role and relationship, and establishing trust and reputation

It provides the models of environment and behavior that represent effective interaction according to specific goals. In the human nervous system metaphor, these models are implicit in the evolutionary "design" of biological entities and specific to the goals of survival and procreation. Definition of what constitutes a policy is necessary to consider what is involved in managing it. A relatively flexible and abstract framework of values, relationships, roles, interactions, resources, and other components of the network environment is required. This sub-system extends far beyond the physical network to the applications in use and the processes and end-users that employ the network to achieve specific goals. It must express the relative values of various resources, outcomes, and processes and include a basis for assessing states and conditions.

Unless embodied in some system outside the autonomic network or implicit to the specific policy implementation, the framework must also accommodate the definition of process, objectives and goals. Business process definitions and descriptions are then an integral part of the policy implementation. Further, as policy management represents the ultimate basis for the operation of the autonomic system, it must be able to report on its operation with respect to the details of its implementation.

The policy management sub-system interoperates (at least) indirectly with all other sub-systems but primarily interacts with:

- autognostics - providing the definition of performance and accepting reports on conditions
- configuration management - providing constraints on device configuration
- security - providing definitions of roles, access and permissions

Autodefense

Autodefense represents a dynamic and adaptive mechanism that responds to malicious and intentional attacks on the network infrastructure, or use of the network infrastructure to attack IT resources. As defensive measures tend to impede the operation of IT, it is optimally capable of balancing performance objectives with typically over-riding threat management actions. In the biological metaphor, this sub-system offers mechanisms comparable to the immune system.

This sub-system must proactively assess network and application infrastructure for risks, detect and identify threats, and define effective both proactive and reactive defensive responses. It has the role of the warrior and the security guard insofar as it has roles for both maintenance and corrective activities. Its relationship with security is close but not identical – security is more concerned with appropriately defined and implemented access and authorization controls to maintain legitimate roles and process. Autodefense deals with forces and processes, typically malicious, outside the normal operation of the system that offer some risk to successful execution.

Autodefense requires high-level and detailed knowledge of the entire network as well as imbedded models of risk that allow it to analyze dynamically the current status.

Corrections to decrease risk must be considered in balance with performance objectives and value of process goals – an overzealous defensive response can immobilize the system (like the immune system inappropriately invoking an allergic reaction). The detection of network or application behaviors that signal possible attack or abuse is followed by the generation of an appropriate response – for example, ports might be temporarily closed or packets with a specific source or destination might be filtered out. Further assessment generates subsequent changes either relaxing the defensive measures or strengthening them.

Autodefense interoperates closely with:

- security - receives definition of roles and security constraints, and defines risk for proactive mitigation
- configuration management - receives details of network for analysis and directs changes in elements in response to anticipated or detected attack
- autognostics - receives notification of detected behaviors

It also may receive definition of relative value of various resources and processes from policy management in order to develop responses consistent with policy.

Security

Security provides the structure that defines and enforces the relationships between roles, content, and resources, particularly with respect to access. It includes the framework for definitions as well as the means to implement them. In metaphor, security parallels the complex mechanisms underlying social interactions, defining friends, foes, mates and allies and offering access to limited resources on the basis of assessed benefit.

Several key means are employed by security – they include the well-known 3 As of authentication, authorization, and access (control). The basis for applying these means requires the definition of roles and their relationships to resources, processes and each other. High-level concepts like privacy, anonymity and verification are likely imbedded in the form of the role definitions and derive from policy. Successful security reliably supports and enforces roles and relationships.

Autodefense has a close association with security – maintaining the assigned roles in balance with performance exposes the system to potential violations in security. In those cases, the system must compensate by making changes that may sacrifice balance on a temporary basis and indeed may violate the operational terms of security itself. Typically the two are viewed as inextricably intertwined – effective security somewhat hopefully negating any need for a defensive response. Security's revised role is to mediate between the competing demands from policy for maximized performance and minimized risk with auto defense recovering the balance when inevitable risk translates to threat. Federation represents one of the key challenges to be solved by effective security.

The security sub-system interoperates directly with:

- policy management - receiving high-level directives related to access and priority
- configuration management - sending specifics for access and admission control
- autodefense - receiving over-riding directives under threat and sending security constraint details for risk assessment

Connection fabric

The connection fabric supports the interaction with all the elements and sub-systems of the autonomic system. It may be composed of a variety of means and mechanisms, or may be a single central framework. The biological equivalent is the central nervous system itself – although referred to as the autonomic system, it actually is only the communication conduit between the human body's faculties.

Principles of autonomic networking

Consequently, it is currently under research by many research projects, how principles and paradigms of mother nature might be applied to networking.

Compartmentalization

Instead of a layering approach, autonomic networking targets a more flexible structure termed compartmentalization.

Function re-composition

The goal is to produce an architectural design that enables flexible, dynamic, and fully autonomic formation of large-scale networks in which the functionalities of each constituent network node are also composed in an autonomic fashion

Atomization

Functions should be divided into atomic units to allow for maximal re-composition freedom.

Closed control loop

A fundamental concept of Control theory, the closed control loop, is among the fundamental principles of autonomic networking. A closed control loop maintains the properties of the controlled system within desired bounds by constantly monitoring target parameters.

Chapter 2

FCAPS

FCAPS is the ISO Telecommunications Management Network model and framework for network management. FCAPS is an acronym for *Fault, Configuration, Accounting, Performance, Security*, the management categories into which the ISO model defines network management tasks. In non-billing organizations *Accounting* is sometimes replaced with *Administration*.

The comprehensive management of an organization's information technology (IT) infrastructure is a fundamental requirement. Employees and customers rely on IT services where availability and performance are mandated, and problems can be quickly identified and resolved. Mean time to repair (MTTR) must be as short as possible to avoid system downtimes where a loss of revenue or lives is possible.

History

In the early 1980s the term FCAPS was introduced within the first Working Drafts (N1719) of ISO 10040, the Open Systems Interconnection (OSI) Systems Management Overview (SMO) standard. At that time the intention was to define five separate protocol standards, one for each functional area. Since initial experiences showed that these protocols would become very similar, the ISO working group responsible for the development of these protocols (ISO/TC97/SC16/WG4, later renamed into ISO-IEC/JTC1/SC21/WG4) decided to create a single protocol for all five areas instead. This protocol is called common management information protocol (CMIP). In the 1990s the ITU-T, as part of their work on Telecommunications Management Network (TMN), further refined the FCAPS as part of the TMN recommendation on Management Functions (M.3400). The idea of FCAPS turned out to be very useful for teaching network management functions; most text books therefore start with a section that explains the FCAPS.

Fault management

A fault is an event that has a negative significance. The goal of fault management is to recognize, isolate, correct and log faults that occur in the network. Furthermore, it uses trend analysis to predict errors so that the network is always available. This can be established by monitoring different things for abnormal behavior.

When a fault or event occurs, a network component will often send a notification to the network operator using a proprietary or open protocol such as SNMP, or at least write a message to its console for a console server to catch and log/page. This notification is supposed to trigger manual or automatic activities. For example, the gathering of more data to identify the nature and severity of the problem or to bring backup equipment on-line.

Fault logs are one input used to compile statistics to determine the provided service level of individual network elements, as well as sub-networks or the whole network. They are also used to determine apparently fragile network components that require further attention.

The leading Fault Management systems are EMC Smarts, CA Spectrum, HP Software, NetIQ, IBM Tivoli Netcool, TTI Telecom Netrac, CA Clarity, Objective Systems Integrators NETeXPERT etc. Fault Isolation tools like Delphi are also available, which are basically used to isolate the fault in any telecom network.

Configuration management

The goals of configuration management include:

- to gather and store configurations from network devices (this can be done locally or remotely).
- to simplify the configuration of the device
- to track changes that are made to the configuration
- to configure ('provision') circuits or paths through non-switched networks

Accounting management

Accounting is often referred to as billing management. The goal is to gather usage statistics for users.

Using the statistics the users can be billed and usage quota can be enforced.

Examples:

- Disk usage
- Link utilization
- CPU time

RADIUS, TACACS and Diameter are examples of protocols commonly used for accounting.

For non-billed networks, "administration" replaces "accounting". The goals of administration are to administer the set of authorized users by establishing users,

passwords, and permissions, and to administer the operations of the equipment such as by performing software backup and synchronization.

Performance management

Performance management enables the manager to prepare the network for the future, as well as to determine the efficiency of the current network, for example, in relation to the investments done to set it up. The network performance addresses the throughput, percentage utilization, error rates and response times areas.

By collecting and analysing performance data, the network health can be monitored. Trends can indicate capacity or reliability issues before they become service affecting.

Performance thresholds can be set in order to trigger an alarm. The alarm would be handled by the normal fault management process (see above). Alarms vary depending upon the severity.

Security management

Security management is the process of controlling access to assets in the network. Data security can be achieved mainly with authentication and encryption. Authorization to it configured with OS and DBMS access control settings.

Chapter 3

Simple Network Management Protocol and Java Management Extensions

Simple Network Management Protocol

Simple Network Management Protocol (**SNMP**) is an "Internet-standard protocol for managing devices on IP networks. Devices that typically support SNMP include routers, switches, Servers, workstations, printers, modem tracks, and more.". It is used mostly in network management systems to monitor network-attached devices for conditions that warrant administrative attention. SNMP is a component of the Internet Protocol Suite as defined by the Internet Engineering Task Force (IETF). It consists of a set of standards for network management, including an application layer protocol, a database schema, and a set of data objects.

SNMP exposes management data in the form of variables on the managed systems, which describe the system configuration. These variables can then be queried (and sometimes set) by managing applications.

Overview and basic concepts

In typical SNMP use, one or more administrative computers called managers have the task of monitoring or managing a group of hosts or devices on a computer network. Each managed system executes, at all times, a software component called an *agent* which reports information via SNMP to the manager.

Essentially, SNMP agents expose management data on the managed systems as variables. The protocol also permits active management tasks, such as modifying and applying a new configuration through remote modification of these variables. The variables accessible via SNMP are organized in hierarchies. These hierarchies, and other metadata (such as type and description of the variable), are described by Management Information Bases (MIBs).

An SNMP-managed network consists of three key components:

- Managed device

- Agent — software which runs on managed devices
- Network management system (NMS) — software which runs on the manager

A *managed device* is a network node that implements an SNMP interface that allows unidirectional (read-only) or bidirectional access to node-specific information. Managed devices exchange node-specific information with the NMSs. Sometimes called network elements, the managed devices can be any type of device, including, but not limited to, routers, access servers, switches, bridges, hubs, IP telephones, IP video cameras, computer hosts, and printers.

An *agent* is a network-management software module that resides on a managed device. An agent has local knowledge of management information and translates that information to or from an SNMP specific form.

A *network management system* (NMS) executes applications that monitor and control managed devices. NMS's provide the bulk of the processing and memory resources required for network management. One or more NMSs may exist on any managed network.

Management information base (MIB)

SNMP itself does not define which information (which variables) a managed system should offer. Rather, SNMP uses an extensible design, where the available information is defined by management information bases (MIBs). MIBs describe the structure of the management data of a device subsystem; they use a hierarchical namespace containing object identifiers (OID). Each OID identifies a variable that can be read or set via SNMP. MIBs use the notation defined by ASN.1.

Protocol details

SNMP operates in the Application Layer of the Internet Protocol Suite (Layer 7 of the OSI model). The SNMP agent receives requests on UDP port 161. The manager may send requests from any available source port to port 161 in the agent. The agent response will be sent back to the source port on the manager. The manager receives notifications (*Traps* and *InformRequests*) on port 162. The agent may generate notifications from any available port.

SNMPv1 specifies five core protocol data units (PDUs). Two other PDUs, *GetBulkRequest* and *InformRequest* were added in SNMPv2 and carried over to SNMPv3.

All SNMP PDUs are constructed as follows:

IP header	UDP header	version	community	PDU-type	request-id	error-status	error-index	variable bindings

The seven SNMP protocol data units (PDUs) are as follows:

GetRequest

A manager-to-agent request to retrieve the value of a variable or list of variables. Desired variables are specified in variable bindings (values are not used). Retrieval of the specified variable values is to be done as an atomic operation by the agent. A *Response* with current values is returned.

SetRequest

A manager-to-agent request to change the value of a variable or list of variables. Variable bindings are specified in the body of the request. Changes to all specified variables are to be made as an atomic operation by the agent. A *Response* with (current) new values for the variables is returned.

GetNextRequest

A manager-to-agent request to discover available variables and their values. Returns a *Response* with variable binding for the lexicographically next variable in the MIB. The entire MIB of an agent can be walked by iterative application of *GetNextRequest* starting at OID 0. Rows of a table can be read by specifying column OIDs in the variable bindings of the request.

GetBulkRequest

Optimized version of *GetNextRequest*. A manager-to-agent request for multiple iterations of *GetNextRequest*. Returns a *Response* with multiple variable bindings walked from the variable binding or bindings in the request. PDU specific *non-repeaters* and *max-repetitions* fields are used to control response behavior. *GetBulkRequest* was introduced in SNMPv2.

Response

Returns variable bindings and acknowledgement from agent to manager for *GetRequest*, *SetRequest*, *GetNextRequest*, *GetBulkRequest* and *InformRequest*. Error reporting is provided by *error-status* and *error-index* fields. Although it was used as a response to both gets and sets, this PDU was called *GetResponse* in SNMPv1.

Trap

Asynchronous notification from agent to manager. Includes current *sysUpTime* value, an OID identifying the type of trap and optional variable bindings. Destination addressing for traps is determined in an application specific manner typically through trap configuration variables in the MIB. The format of the trap message was changed in SNMPv2 and the PDU was renamed *SNMPv2-Trap*.

InformRequest

Acknowledged asynchronous notification from manager to manager. This PDU uses the same format as the SNMPv2 version of *Trap*. Manager-to-manager notifications were already possible in SNMPv1 (using a *Trap*), but as SNMP commonly runs over UDP where delivery is not assured and dropped packets are not reported, delivery of a *Trap* was not guaranteed. *InformRequest* fixes this by sending back an acknowledgement on receipt. Receiver replies with *Response* parroting all information in the *InformRequest*. This PDU was introduced in SNMPv2.

Development and usage

Version 1

SNMP version 1 (SNMPv1) is the initial implementation of the SNMP protocol. SNMPv1 operates over protocols such as User Datagram Protocol (UDP), Internet Protocol (IP), OSI Connectionless Network Service (CLNS), AppleTalk Datagram-Delivery Protocol (DDP), and Novell Internet Packet Exchange (IPX). SNMPv1 is widely used and is the de facto network-management protocol in the Internet community.

The first RFCs for SNMP, now known as SNMPv1, appeared in 1988:

- RFC 1065 — Structure and identification of management information for TCP/IP-based internets
- RFC 1066 — Management information base for network management of TCP/IP-based internets
- RFC 1067 — A simple network management protocol

These protocols were obsoleted by:

- RFC 1155 — Structure and identification of management information for TCP/IP-based internets
- RFC 1156 — Management information base for network management of TCP/IP-based internets
- RFC 1157 — A simple network management protocol

After a short time, RFC 1156 (MIB-1) was replaced by more often used:

- RFC 1213 — Version 2 of management information base (MIB-2) for network management of TCP/IP-based internets

Version 1 has been criticized for its poor security. Authentication of clients is performed only by a "community string", in effect a type of password, which is transmitted in cleartext. The '80s design of SNMP V1 was done by a group of collaborators who viewed the officially sponsored OSI/IETF/NSF (National Science Foundation) effort (HEMS/CMIS/CMIP) as both unimplementable in the computing platforms of the time as well as potentially unworkable. SNMP was approved based on a belief that it was an interim protocol needed for taking steps towards large scale deployment of the Internet and its commercialization. In that time period Internet-standard authentication/security was both a dream and discouraged by focused protocol design groups.

Version 2

SNMPv2 (RFC 1441–RFC 1452), revises version 1 and includes improvements in the areas of performance, security, confidentiality, and manager-to-manager communications. It introduced *GetBulkRequest*, an alternative to iterative GetNextRequests for retrieving large amounts of management data in a single request. However, the new party-based security system in SNMPv2, viewed by many as overly complex, was not widely accepted.

Community-Based Simple Network Management Protocol version 2, or *SNMPv2c*, is defined in RFC 1901–RFC 1908. In its initial stages, this was also informally known as *SNMPv1.5*. SNMPv2c comprises SNMPv2 *without* the controversial new SNMP v2 security model, using instead the simple community-based security scheme of SNMPv1. While officially only a "Draft Standard", this is widely considered the *de facto* SNMPv2 standard.

User-Based Simple Network Management Protocol version 2, or *SNMPv2u*, is defined in RFC 1909–RFC 1910. This is a compromise that attempts to offer greater security than SNMPv1, but without incurring the high complexity of SNMPv2. A variant of this was commercialized as *SNMP v2**, and the mechanism was eventually adopted as one of two security frameworks in SNMP v3.

SNMPv1 & SNMPv2c interoperability

As presently specified, SNMPv2 is incompatible with SNMPv1 in two key areas: message formats and protocol operations. SNMPv2c messages use different header and protocol data unit (PDU) formats from SNMPv1 messages. SNMPv2c also uses two protocol operations that are not specified in SNMPv1. Furthermore, RFC 2576 defines two possible SNMPv1/v2c coexistence strategies: proxy agents and bilingual network-management systems.

Proxy agents

A SNMPv2 agent can act as a proxy agent on behalf of SNMPv1 managed devices, as follows:

- A SNMPv2 NMS issues a command intended for a SNMPv1 agent.
- The NMS sends the SNMP message to the SNMPv2 proxy agent.
- The proxy agent forwards `Get`, `GetNext`, and `Set` messages to the SNMPv1 agent unchanged.
- GetBulk messages are converted by the proxy agent to `GetNext` messages and then are forwarded to the SNMPv1 agent.

The proxy agent maps SNMPv1 trap messages to SNMPv2 trap messages and then forwards them to the NMS.

Bilingual network-management system

Bilingual SNMPv2 network-management systems support both SNMPv1 and SNMPv2. To support this dual-management environment, a management application in the bilingual NMS must contact an agent. The NMS then examines information stored in a local database to determine whether the agent supports SNMPv1 or SNMPv2. Based on the information in the database, the NMS communicates with the agent using the appropriate version of SNMP.

Version 3

Although SNMPv3 makes no changes to the protocol aside from the addition of cryptographic security, its developers have managed to make things look much different by introducing new textual conventions, concepts, and terminology.

SNMPv3 primarily added security and remote configuration enhancements to SNMP.

Security has been the biggest weakness of SNMP since the beginning. Authentication in SNMP Versions 1 and 2 amounts to nothing more than a password (community string) sent in clear text between a manager and agent. Each SNMPv3 message contains security parameters which are encoded as an octet string. The meaning of these security parameters depends on the security model being used.

SNMPv3 provides important security features:

- Confidentiality - Encryption of packets to prevent snooping by an unauthorized source.
- Integrity - Message integrity to ensure that a packet has not been tampered with in transit.
- Authentication - to verify that the message is from a valid source.

As of 2004 the IETF recognizes *Simple Network Management Protocol version 3* as defined by RFC 3411–RFC 3418 (also known as STD0062) as the current standard version of SNMP. The IETF has designated SNMPv3 a full Internet standard, the highest maturity level for an RFC. It considers earlier versions to be obsolete (designating them "Historic").

In practice, SNMP implementations often support multiple versions: typically SNMPv1, SNMPv2c, and SNMPv3.

Applications

Environmental monitoring

Server, rack and appliance operating temperatures and room humidity could be monitored remotely for SNMP-enabled HVAC devices.

Implementation issues

SNMP implementations vary across platform vendors. In some cases, SNMP is an added feature, and is not taken seriously enough to be an element of the core design. Some major equipment vendors tend to over-extend their proprietary command line interface (CLI) centric configuration and control systems.

SNMP's seemingly simple tree structure and linear indexing may not always be understood well enough within the internal data structures that are elements of a platform's basic design. As a result, processing SNMP queries on certain data sets may result in higher CPU utilization than necessary. One example of this would be large routing tables, such as BGP or IGP.

Resource indexing

Modular devices may dynamically increase or decrease their SNMP indices (aka instances) whenever slotted hardware is added or removed. Although this is most common with hardware, virtual interfaces have the same effect. Index values are typically assigned at boot time and remain fixed until the next reboot. Hardware or virtual entities added while the device is 'live' may have their indexes assigned at the end of the existing range and possibly reassigned at the next reboot. Network inventory and monitoring tools need to have the device update capability by properly reacting to the cold start trap from the device reboot in order to avoid corruption and mismatch of polled data.

Index assignments for an SNMP device instance may change from poll to poll mostly as a result of changes initiated by the system admin. If information is needed for a particular interface, it is imperative to determine the SNMP index before retrieving the data needed. Generally, a description table like ifDescr will map a user friendly name like Serial 0/1 (Blade 0, port 1) to a SNMP index.

Security implications

- SNMP versions 1 and 2c are subject to packet sniffing of the clear text community string from the network traffic, because they do not implement encryption.
- All versions of SNMP are subject to brute force and dictionary attacks for guessing the community strings, authentication strings, authentication keys, encryption strings, or encryption keys, because they do not implement a challenge-response handshake. Entropy is an important consideration when selecting keys, passwords and/or algorithms.
- Although SNMP works over TCP and other protocols, it is most commonly used over UDP that is connectionless and vulnerable to IP spoofing attacks. Thus, all versions are subject to bypassing device access lists that might have been implemented to restrict SNMP access, though SNMPv3's other security mechanisms should prevent a successful attack.
- SNMP's powerful configuration (write) capabilities are not being fully utilized by many vendors, partly due to lack of security in SNMP versions before SNMPv3 and partly due to the fact that many devices simply are not capable of being configured via individual MIB object changes.
- SNMP tops the list of the SANS Institute's Common Default Configuration Issues with the issue of default SNMP community strings set to ‘public’ and ‘private’ and was number ten on the SANS Top 10 Most Critical Internet Security Threats for the year 2000.

Autodiscovery

SNMP by itself is simply a protocol for collecting and organizing information. Most toolsets implementing SNMP offer some form of discovery mechanism, a standardized collection of data common to most platforms and devices, to get a new user or implementor started. One of these features is often a form of automatic discovery, where new devices discovered in the network are polled automatically. For SNMPv1 and SNMPv2c, this presents a security risk, in that your SNMP read communities will be broadcast in cleartext to the target device. While security requirements and risk profiles vary from organization to organization, care should be taken when using a feature like this, with special regard to common environments such as mixed-tenant datacenters, server hosting and colocation facilities, and similar environments.

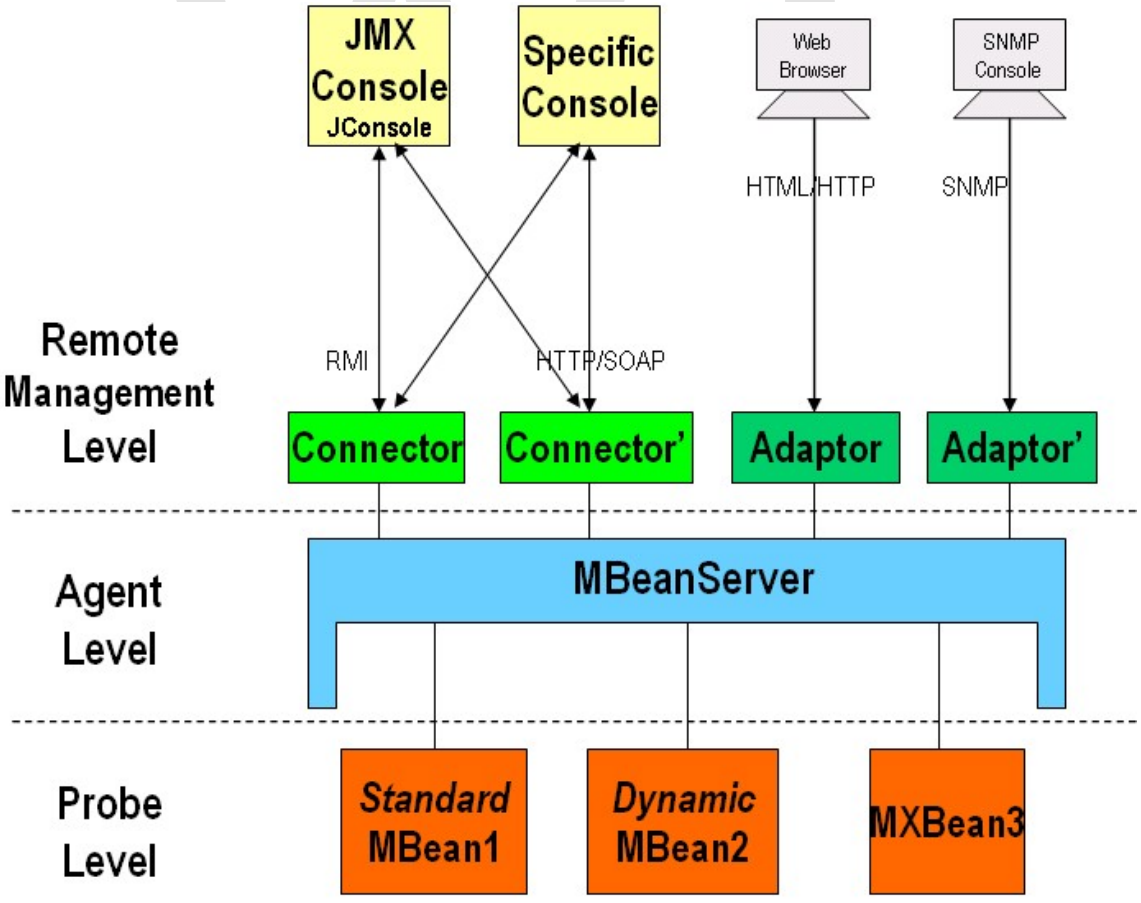
Java Management Extensions

Java Management Extensions (**JMX**) is a Java technology that supplies tools for managing and monitoring applications, system objects, devices (e. g. printers) and service oriented networks. Those resources are represented by objects called MBeans (for *Managed Bean*). In the API, classes can be dynamically loaded and instantiated. Managing and monitoring applications can be designed and developed by Java Dynamic Management Kit.

JMX 1.0, 1.1 and 1.2 were defined by JSR 003 of the Java Community Process. As of 2006, JMX 2.0 is being developed under JSR 255. The JMX Remote API 1.0 for remote management and monitoring is specified by JSR 160. An extension of the JMX Remote API for Web Services is being developed under JSR 262.

Adopted early on by the J2EE community, JMX has been a part of J2SE since version 5.0. It is a trademark of Sun Microsystems, Inc.

Architecture



JMX architecture.

JMX is based on a 3-level architecture:

- The *Probe* level contains the probes (called MBeans) instrumenting the resources. Also called the *Instrumentation* level.
- The *Agent* level, or MBeanServer, is the core of JMX. It is an intermediary between the MBean and the applications.
- The *Remote Management* level enables remote applications to access the MBeanServer through Connectors and Adaptors. A connector provides full remote access to the MBeanServer API using various communication frameworks (RMI, IIOP, JMS, WS-* …), while an adaptor adapts the API to another protocol (SNMP, …) or to Web-based GUI (HTML/HTTP, WML/HTTP, …).

Applications can be generic consoles (such as JConsole and MC4J), or domain-specific (monitoring) applications. External applications can interact with the MBeans through the use of JMX connectors and protocol adapters. Connectors are used to connect an agent with a remote JMX-enabled management application. This form of communication involves a connector in the JMX agent and a connector client in the management application.

Protocol adapters provide a management view of the JMX agent through a given protocol. Management applications that connect to a protocol adapter are usually specific to the given protocol.

Managed Bean

A **managed bean** - sometimes simply referred to as an *MBean* - is a type of JavaBean, created with dependency injection. Managed Beans are particularly used in the Java Management Extensions technology. But, with the Java EE 6 specification provides for a more detailed meaning of a managed bean.

The MBean represents a resource running in the Java virtual machine, such as an application or a Java EE technical service (transactional monitor, JDBC driver, etc.). They can be used for getting and setting applications configuration (pull), for collecting statistics (pull) (e.g. performance, resources usage, problems) and notifying events (push) (e.g. faults, state changes).

Java EE 6 provides that a managed bean is a bean that is implemented by a Java class, which is called its bean class. A top-level Java class is a managed bean if it is defined to be a managed bean by any other Java EE technology specification (for example, the JavaServer Faces technology specification), or if it meets all of the following conditions:

1. It is not a non-static inner class.
2. It is a concrete class, or is annotated @Decorator.
3. It is not annotated with an EJB component-defining annotation or declared as an EJB bean class in ejb-jar.xml.

No special declaration, such as an annotation, is required to define a managed bean.

An MBean can notify the MBeanServer of its internal changes (for the attributes) by implementing the javax.management.NotificationEmitter. The application interested in the MBean's changes registers a listener (javax.management.NotificationListener) to the MBeanServer. Note that JMX does not guarantee that all notifications will be received by the listeners.

Types

There are two basic types of MBean:

- *Standard MBeans* implement a business interface containing setters and getters for the attributes and the operations (i.e., methods).
- *Dynamic MBeans* implement the javax.management.DynamicMBean interface which provides a way to list the attributes and operations, and to get and set the attribute values.

Additional types are *Open MBeans*, *Model MBeans* and *Monitor MBeans*. *Open MBeans* are dynamic MBeans that rely on the basic data types. They are self-explanatory and more user friendly. *Model MBeans* are dynamic MBeans that can be configured during runtime. A generic MBean class is also provided for dynamically configuring the resources during program runtime.

An MXBean (*Platform MBean*) is a special type of MBean that reifies Java Virtual Machine subsystems such as memory pools, garbage collection, multi-threading, JIT compilation, etc.

An MLet (*Management applet*) is a utility MBean to load, instantiate and register MBeans in the MBeanServer from a XML description. The format of the XML descriptor is:

```
<MLET CODE = ''class'' | OBJECT = ''serfile''
  ARCHIVE = ''archiveList''
  [CODEBASE = ''codebaseURL'']
  [NAME = ''objectName'']
  [VERSION = ''version'']
>
  [arglist]
</MLET>
```

Support

JMX is supported at various levels by different vendors:

- JMX is supported by Java application servers such as OpenCloud Rhino Application Server , JBoss, JOnAS, WebSphere Application Server, WebLogic,

SAP Netweaver Application Server, Oracle Application Server 10g and Sun Java System Application Server.

- Systems management tools that support the protocol include IBM Director, HP OpenView, Zyrion, Zenoss, Zabbix, Hyperic, Empirix OneSight and GroundWork Monitor.
- JMX is also supported by servlet containers such as Apache Tomcat.
- MX4J is Open Source JMX for Enterprise Computing.
- jManage is an open source enterprise-grade JMX Console with web and command-line interfaces.
- MC4J is an open source visual console for connecting to servers supporting JMX

Chapter 4

Load Balancing (Computing) and Network Administrator

Load balancing (computing)

In networking, **load balancing** is a technique to distribute workload evenly across two or more computers, network links, CPUs, hard drives, or other resources, in order to get optimal resource utilization, maximize throughput, minimize response time, and avoid overload. Using multiple components with load balancing, instead of a single component, may increase reliability through redundancy. The load balancing service is usually provided by a dedicated program or hardware device (such as a multilayer switch or a DNS server).

It is commonly used to mediate internal communications in computer clusters, especially high-availability clusters. If the load is more on a server, then the secondary server takes some load while the other is still processing requests.

Internet-based services

One of the most common applications of load balancing is to provide a single Internet service from multiple servers, sometimes known as a server farm. Commonly, load-balanced systems include popular web sites, large Internet Relay Chat networks, high-bandwidth File Transfer Protocol sites, Network News Transfer Protocol (NNTP) servers and Domain Name System (DNS) servers.

For Internet services, the load balancer is usually a software program that is listening on the port where external clients connect to access services. The load balancer forwards requests to one of the "backend" servers, which usually replies to the load balancer. This allows the load balancer to reply to the client without the client ever knowing about the internal separation of functions. It also prevents clients from contacting backend servers directly, which may have security benefits by hiding the structure of the internal network

and preventing attacks on the kernel's network stack or unrelated services running on other ports.

Some load balancers provide a mechanism for doing something special in the event that all backend servers are unavailable. This might include forwarding to a backup load balancer, or displaying a message regarding the outage.

An alternate method of load balancing, which does not necessarily require a dedicated software or hardware node, is called **round robin DNS**. In this technique, multiple IP addresses are associated with a single domain name; clients themselves are expected to choose which server to connect to. Unlike the use of a dedicated load balancer, this technique exposes to clients the existence of multiple backend servers. The technique has other advantages and disadvantages, depending on the degree of control over the DNS server and the granularity of load balancing desired.

Another technique for load-balancing using DNS, which is far more intelligent than the simple "round robin", is to delegate `www.example.org` as a sub-domain whose zone will be served out by each of the same servers that are serving the web site. This technique works particularly well where individual servers are spread around the Internet. For example,

```
one.example.org A 1.1.1.1
two.example.org A 2.2.2.2
www.example.org NS one.example.org
www.example.org NS two.example.org
```

However, the zone file for `www.example.org` on each server will be different such that each server will give out its own IP Address as the A-record. On "one" the zone file for `www.example.org` will say:

```
@ in a 1.1.1.1
```

On "two" the same zone file will say:

```
@ in a 2.2.2.2
```

This way, if a server is down, its DNS will not respond and so the web service will not receive any traffic. Also if the line to one server becomes congested the unreliability of DNS will ensure less HTTP traffic will reach that server, further the DNS response that gets back to the resolver the quickest will nearly always be from the network closest server, ensuring geo-sensitive load-balancing. A short TTL on the A-record will also help to ensure traffic is quickly diverted if a server goes down. Consideration must be given the possibility that this technique may cause individual clients to switch between individual servers mid-session.

A variety of scheduling algorithms are used by load balancers to determine which backend server to send a request to. Simple algorithms include random choice or round

robin. More sophisticated load balancers may take into account additional factors, such as a server's reported load, recent response times, up/down status (determined by a monitoring poll of some kind), number of active connections, geographic location, capabilities, or how much traffic it has recently been assigned. High-performance systems may use multiple layers of load balancing.

In addition to using dedicated hardware load balancers, software-only solutions are available, including open source options. Examples of the latter include the Apache web server's mod_proxy_balancer extension and the Pound reverse proxy and load balancer.

In a Multitier architecture, terminology for designs behind a load balancer or network dispatcher may include **Bowties** and **Stovepipes**. A stovepipe presents a situation such that a transaction that is dispatched at a top tier follows a static path through the stack of devices and software behind the load balancer to its final destination. Alternatively, if Bowties are used, at each tier the transaction could take one of many paths after being serviced by the applications at a particular tier. Network diagrams with transaction flows resemble Stovepipes or Bowties, or hybrid architectures based on need at each tier.

Persistence

An important issue when operating a load-balanced service is how to handle information that must be kept across the multiple requests in a user's session. If this information is stored locally on one backend server, then subsequent requests going to different backend servers would not be able to find it. This might be cached information that can be recomputed, in which case load-balancing a request to a different backend server just introduces a performance issue...

One solution to the session data issue is to send all requests in a user session consistently to the same backend server. This is known as "persistence" or "stickiness". A significant downside to this technique is its lack of automatic failover: if a backend server goes down, its per-session information becomes inaccessible, and any sessions depending on it are lost. The same problem is usually relevant to central database servers; even if web servers are "stateless" and not "sticky", the central database is (see below).

Assignment to a particular server might be based on a username, client IP address, or random assignment. Owing to DHCP, Network Address Translation, and web proxies, the client's IP address may change across requests, and so this method can be somewhat unreliable. Random assignments must be remembered by the load balancer, which creates a storage burden. If the load balancer is replaced or fails, this information can be lost, and assignments may need to be deleted after a timeout period or during periods of high load to avoid exceeding the space available for the assignment table. The random assignment method also requires that clients maintain some state, which can be a problem, for example when a web browser has disabled storage of cookies. Sophisticated load balancers use multiple persistence techniques to avoid some of the shortcomings of any one method.

Another solution is to keep the per-session data in a database. Generally this is bad for performance since it increases the load on the database: the database is best used to store information less transient than per-session data. To prevent a database from becoming a single point of failure, and to improve scalability, the database is often replicated across multiple machines, and load balancing is used to spread the query load across those replicas. Microsoft's ASP.net State Server technology is an example of a session database. All servers in a web farm store their session data on State Server and any server in the farm can retrieve the data.

Fortunately there are more efficient approaches. In the very common case where the client is a web browser, per-session data can be stored in the browser itself. One technique is to use a browser cookie, suitably time-stamped and encrypted. Another is URL rewriting. Storing session data on the client is generally the preferred solution: then the load balancer is free to pick any backend server to handle a request. However, this method of state-data handling is not really suitable for some complex business logic scenarios, where session state payload is very big or recomputing it with every request on a server is not feasible, and URL rewriting has major security issues, since the end-user can easily alter the submitted URL and thus change session streams.

Load balancer features

Hardware and software load balancers can come with a variety of special features.

- **Asymmetric load:** A ratio can be manually assigned to cause some backend servers to get a greater share of the workload than others. This is sometimes used as a crude way to account for some servers being faster than others.
- **Priority activation:** When the number of available servers drops below a certain number, or load gets too high, standby servers can be brought online
- **SSL Offload and Acceleration:** SSL applications can be a heavy burden on the resources of a Web Server, especially on the CPU and the end users may see a slow response (or at the very least the servers are spending a lot of cycles doing things they weren't designed to do). To resolve these kinds of issues, a Load Balancer capable of handling SSL Offloading in specialized hardware may be used. When Load Balancers are taking the SSL connections, the burden on the Web Servers is reduced and performance will not degrade for the end users.
- **Distributed Denial of Service (DDoS) attack protection:** load balancers can provide features such as SYN cookies and delayed-binding (the back-end servers don't see the client until it finishes its TCP handshake) to mitigate SYN flood attacks and generally offload work from the servers to a more efficient platform.
- **HTTP compression:** reduces amount of data to be transferred for HTTP objects by utilizing gzip compression available in all modern web browsers
- **TCP offload:** different vendors use different terms for this, but the idea is that normally each HTTP request from each client is a different TCP connection. This feature utilizes HTTP/1.1 to consolidate multiple HTTP requests from multiple clients into a single TCP socket to the back-end servers.

- **TCP buffering:** the load balancer can buffer responses from the server and spoon-feed the data out to slow clients, allowing the server to move on to other tasks.
- **Direct Server Return:** an option for asymmetrical load distribution, where request and reply have different network paths.
- **Health checking:** the balancer will poll servers for application layer health and remove failed servers from the pool.
- **HTTP caching:** the load balancer can store static content so that some requests can be handled without contacting the web servers.
- **Content Filtering:** some load balancers can arbitrarily modify traffic on the way through.
- **HTTP security:** some load balancers can hide HTTP error pages, remove server identification headers from HTTP responses, and encrypt cookies so end users can't manipulate them.
- **Priority queuing:** also known as rate shaping, the ability to give different priority to different traffic.
- **Content aware switching:** most load balancers can send requests to different servers based on the URL being requested.
- **Client authentication:** authenticate users against a variety of authentication sources before allowing them access to a website.
- **Programmatic traffic manipulation:** at least one load balancer allows the use of a scripting language to allow custom load balancing methods, arbitrary traffic manipulations, and more.
- **Firewall:** direct connections to backend servers are prevented, for network security reasons
- **Intrusion Prevention System:** offer application layer security in addition to network/transport layer offered by firewall security.

In telecommunications

Load balancing can be useful when dealing with redundant communications links. For example, a company may have multiple Internet connections ensuring network access even if one of the connections should fail.

A failover arrangement would mean that one link is designated for normal use, while the second link is used only if the first one fails.

With load balancing, both links can be in use all the time. A device or program decides which of the available links to send packets along, being careful not to send packets along any link if it has failed. The ability to use multiple links simultaneously increases the available bandwidth.

Major telecommunications companies have multiple routes through their networks or to external networks. They use more sophisticated load balancing to shift traffic from one path to another to avoid network congestion on any particular link, and sometimes to minimize the cost of transit across external networks or improve network reliability.

Relationship with failover

Load balancing is often used to implement failover — the continuation of a service after the failure of one or more of its components. The components are monitored continually (e.g., web servers may be monitored by fetching known pages), and when one becomes non-responsive, the load balancer is informed and no longer sends traffic to it. And when a component comes back on line, the load balancer begins to route traffic to it again. For this to work, there must be at least one component in excess of the service's capacity. This is much less expensive and more flexible than failover approaches where a single "live" component is paired with a single "backup" component that takes over in the event of a failure. Some types of RAID systems can also utilize hot spare for a similar effect.

Network administrator

A **network administrator** is a person responsible for the maintenance of computer hardware and software that comprises a computer network. This normally includes deploying, configuring, maintaining and monitoring active network equipment. A related role is that of the network specialist, or network analyst, who concentrates on network design and security.

The network administrator (or "network admin") is usually the level of technical/network staff in an organization and will rarely be involved with direct user support. The network administrator will concentrate on the overall integrity of the network, server deployment, security, and ensuring that the network connectivity throughout a company's LAN/WAN infrastructure is on par with technical considerations at the network level of an organization's hierarchy. Network administrators are considered tier 3 support personnel that only work on break/fix issues that could not be resolved at the tier 1 (helpdesk) or tier 2 (desktop/network technician) levels.

Depending on the company, the Network Administrator may also design and deploy networks. However, these tasks may be assigned to a network engineer should one be available to the company. A network engineer designs, implements, and/or troubleshoots computer networks. In general, a network engineer will not regularly perform system administration tasks, but will instead concentrate on high-level network related duties such as network architecture, network design, choosing of network devices, and network policies. Network engineers will rarely be involved with direct user support, as they are normally placed as tier three support.

The actual role of the Network Administrator will vary from company to company, but will commonly include activities and tasks such as network address assignment, assignment of routing protocols and routing table configuration as well as configuration of authentication and authorization – directory services. It often includes maintenance of

network facilities in individual machines, such as drivers and settings of personal computers as well as printers and such. It sometimes also includes maintenance of certain network servers: file servers, VPN gateways, intrusion detection systems, etc.

Network specialists and analysts concentrate on the network design and security, particularly troubleshooting and/or debugging network-related problems. Their work can also include the maintenance of the network's authorization infrastructure, as well as network backup systems.

The administrator is responsible for the security of the network and for assigning IP addresses to the devices connected to the networks. Assigning IP addresses gives the subnet administrator some control over the personnel who connect to the subnet. It also helps ensure that the administrator knows each system that is connected and who is personally responsible for the system.

Duties of a network administrator

Many organizations use a three tier support staff solution, with tier one (help desk) personnel handling the initial calls, tier two (technicians and pc support analysts) and tier three (network administrators). Most of those organizations follow a fixed staffing ratio, and being a network administrator is either the top job, or next to top job, within the technical support department.

Network administrators are responsible for making sure that the computer hardware and network infrastructure for an IT organization is properly maintained. They are deeply involved in the procurement of new hardware (For example: Does it meet existing standardization requirements? Does it do the job required?), rolling out new software installs, maintaining the disk images for new computer installs (usually by having a standardized OS and application install), making sure that licenses are paid for and up to date for software that need it, maintaining the standards for server installations and applications, and monitoring the performance of the network, checking for security breaches, poor data management practices and more.

Most network administrator positions require a breadth of technical knowledge and the ability to learn the ins and outs of new networking and server software packages quickly. While designing and drafting a network is usually the job of a network engineer, many organizations roll that function into a network administrator position as well.

One of the chief jobs of a network administrator is connectivity. Network administrators are in charge of making sure that connectivity works for all users in their organization, and for making sure that data security for connections to the internet is properly handled. (For network administrators doing security aspects, this can be a full time job.)

Trouble tickets work their way through the help desk, then through the analyst level support, before reaching the network administrator's level. As a result, in their day-to-day operations, network administrators should not be dealing directly with end users as a

routine function. Most of their jobs should be on scheduling and implementing routine maintenance tasks, updating disaster prevention programs, making sure that network backups are run and doing test restores to make sure that those restores are sound.

Chapter 5

Management Information Base and NETCONF

Management information base

A **management information base (MIB)** is a virtual database used for managing the entities in a communications network. Most often associated with the Simple Network Management Protocol (SNMP), the term is also used more generically in contexts such as in OSI/ISO Network management model. While intended to refer to the complete collection of management information available on an entity, it is often used to refer to a particular subset, more correctly referred to as MIB-module.

Objects in the MIB are defined using a subset of Abstract Syntax Notation One (ASN.1) called "Structure of Management Information Version 2 (SMIv2)" RFC 2578.The software that performs the parsing is a MIB compiler.

The database is hierarchical (tree-structured) and entries are addressed through object identifiers. Internet documentation RFCs discuss MIBs, notably RFC 1155, "Structure and Identification of Management Information for TCP/IP based internets", and its two companions, RFC 1213, "Management Information Base for Network Management of TCP/IP-based internets", and RFC 1157, "A Simple Network Management Protocol".

Abstract Syntax Notation One (ASN.1)

In telecommunications and computer networking, Abstract Syntax Notation One (ASN.1) is a standard and flexible notation that describes data structures for representing, encoding, transmitting, and decoding data. It provides a set of formal rules for describing the structure of objects that are independent of machine-specific encoding techniques and is a precise, formal notation that removes ambiguities.

ASN.1 is a joint ISO and ITU-T standard, originally defined in 1984 as part of CCITT X.409:1984. ASN.1 moved to its own standard, X.208, in 1988 due to wide applicability. The substantially revised 1995 version is covered by the X.680 series.

An adapted subset of ASN.1, Structure of Management Information (SMI), is specified in SNMP to define sets of related MIB objects; these sets are termed MIB modules.

MIB hierarchy

The MIB hierarchy can be depicted as a tree with a nameless root, the levels of which are assigned by different organizations. The top-level MIB OIDs belong to different standards organizations, while lower-level object IDs are allocated by associated organizations. This model permits management across all layers of the OSI reference model, extending into applications such as databases, email, and the Java reference model, as MIBs can be defined for all such area-specific information and operations.

A managed object (sometimes called a MIB object, an object, or a MIB) is one of any number of specific characteristics of a managed device. Managed objects are made up of one or more object instances (identified by their OIDs), which are essentially variables.

Two types of managed objects exist:

- Scalar objects define a single object instance.
- Tabular objects define multiple related object instances that are grouped in MIB tables.

An example of a managed object is `atInput`, which is a scalar object that contains a single object instance, the integer value that indicates the total number of input AppleTalk packets on a router interface.

An object identifier (or object ID or OID) uniquely identifies a managed object in the MIB hierarchy.

SNMPv1 and SMI-specific data types

The first version of the SMI (SMIv1) specifies the use of a number of SMI-specific data types, which are divided into two categories:

- Simple data types
- Application-wide data types

Simple data types

Three simple data types are defined in the SNMPv1 SMI, all of which are unique values:

- The integer data type is a signed integer in the range of -2^{31} to $2^{31}-1$.
- Octet strings are ordered sequences of 0 to 65,535 octets.
- Object IDs come from the set of all object identifiers allocated according to the rules specified in ASN.1.

Application-wide data types

The following application-wide data types exist in the SNMPv1 SMI:

- *Network addresses* represent addresses from a particular protocol family. SMIv1 supports only 32-bit (IPv4) addresses (SMIv2 uses Octet Strings to represent addresses generically, and thus are usable in SMIv1 too. SMIv1 had an explicit IPv4 address datatype.)
- *Counters* are non-negative integers that increase until they reach a maximum value and then roll over to zero. SNMPv1 specifies a counter size of 32 bits.
- *Gauges* are non-negative integers that can increase or decrease between specified minimum and maximum values. Whenever the system property represented by the gauge is outside of that range, the value of the gauge itself will vary no further than the respective maximum or minimum, as specified in RFC 2578.
- *Time ticks* represent time since some event, measured in hundredths of a second.
- *Opaques* represent an arbitrary encoding that is used to pass arbitrary information strings that do not conform to the strict data typing used by the SMI.
- *Integers* represent signed integer-valued information. This data type redefines the integer data type, which has arbitrary precision in ASN.1 but bounded precision in the SMI.
- *Unsigned integers* represent unsigned integer-valued information, which is useful when values are always non-negative. This data type redefines the integer data type, which has arbitrary precision in ASN.1 but bounded precision in the SMI.

SNMPv1 MIB tables

The SNMPv1 SMI defines highly structured tables that are used to group the instances of a tabular object (that is, an object that contains multiple variables). Tables are composed of zero or more rows, which are indexed in a way that allows SNMP to retrieve or alter an entire row with a single `Get`, `GetNext`, or `Set` command.

SMIv2 and structure of management information

The second version of the SMI (SMIv2) is described in RFC 2578 - RFC 2579. It makes certain additions and enhancements to the SMIv1-specific data types, such as including bit strings, network addresses, and counters. Bit strings are defined only in SMIv2 and comprise zero or more named bits that specify a value. Network addresses represent an address from a particular protocol family. Counters are non-negative integers that increase until they reach a maximum value and then return to zero. In SMIv1, a 32-bit counter size is specified. In SMIv2, 32-bit and 64-bit counters are defined.

SMIv2 also specifies information modules, which specify a group of related definitions. Three types of SMI information modules exist: MIB modules, compliance statements, and capability statements.

- MIB modules contain definitions of interrelated managed objects.

- Compliance statements provide a systematic way to describe a group of managed objects that must be implemented for conformance to a standard.
- Capability statements are used to indicate the precise level of support that an agent claims with respect to a MIB group. A NMS can adjust its behavior toward agents according to the capabilities statements associated with each agent.

Updating MIBs

MIBs are periodically updated to add new functionality, remove ambiguities and to fix defects. These changes are made in conformance to section 10 of RFC 2578. An example of an MIB that has been updated many times is the important set of objects that was originally defined in RFC 1213 "MIB-II". This MIB has since been split up and can be found in MIBs such as RFC 4293 "Management Information Base for the Internet Protocol (IP)", RFC 4022 "Management Information Base for the Transmission Control Protocol (TCP)", RFC 4113 "Management Information Base for the User Datagram Protocol (UDP)", RFC 2863 "The Interfaces Group MIB" and RFC 3418 "Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)".

MIBs index

There are a large number of MIBs defined by both standards organizations like the IETF, private enterprises and other entities.

IETF maintained

There are 318 RFCs in the first 5000 RFCs from the IETF that contain MIBs. This list is merely a fraction of the MIBs that have been written:

- **SNMP - SMI**:RFC 1155 - Defines the Structure of Management Information (SMI)
- **MIB-I**: RFC 1156 - Historically used with CMOT , not to be used with SNMP
- **SNMPv2-SMI**: RFC 2578 - Structure of Management Information Version 2 (SMIv2)
- **MIB-II**: RFC 1213 - Management Information Base for Network Management of TCP/IP-based internets
- **SNMPv2-MIB**: RFC 3418 - Management Information Base (MIB) for the Simple Network Management Protocol (SNMP)
- **TCP-MIB**: RFC 4022 - Management Information Base for the Transmission Control Protocol (TCP)
- **UDP-MIB**: RFC 4113 - Management Information Base for the User Datagram Protocol (UDP)
- **IP-MIB**: RFC 4293 - Management Information Base for the Internet Protocol (IP)
- **IF-MIB**: RFC 2863 - The Interfaces Group MIB
- **ENTITY-MIB**: RFC 4133 - Entity MIB (Version 3)
- **ENTITY-STATE-MIB**: RFC 4268 - Entity State MIB
- **ALARM-MIB**: RFC 3877 - Alarm Management Information Base (MIB)

- Fibre Channel
  - **FC-MGMT-MIB**: RFC 4044 Fibre Channel Management MIB
  - **FIBRE-CHANNEL-FE-MIB**: RFC 2837 Definitions of Managed Objects for the Fabric Element in Fibre Channel Standard
- **HPR-IP-MIB**: RFC 2584 - Definitions of Managed Objects for APPN/HPR in IP Networks

IEEE maintained

The IETF and IEEE have agreed to move MIBs relating to IEEE work (for example Ethernet and bridging) to their respective IEEE workgroup. This is in process and a few items are complete.

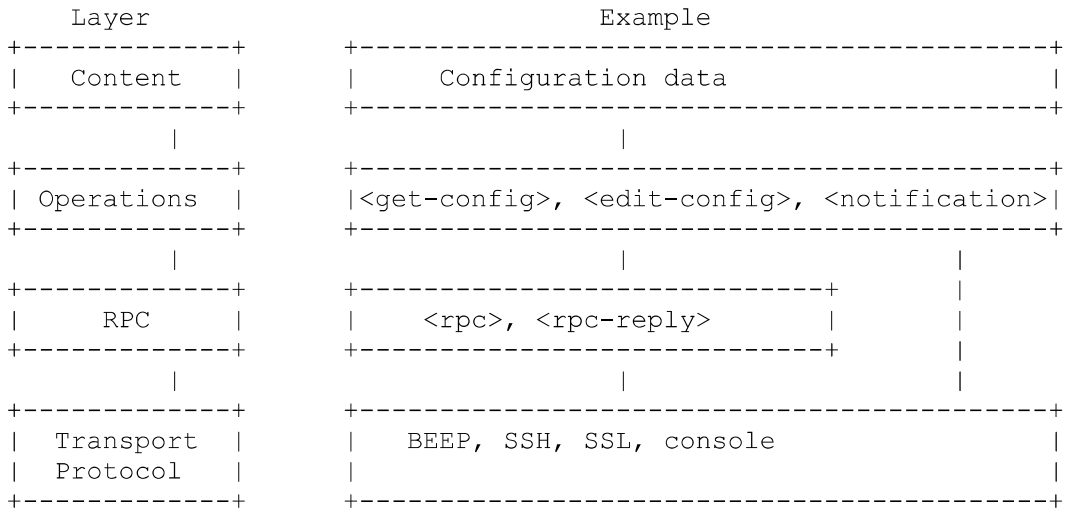
- Network bridge
  - IEEE 802.1ap-2008 consolidated the IEEE and IETF RFCs related to bridging networks into eight related MIBs.

NETCONF

The Network Configuration Protocol, **NETCONF**, is an IETF network management protocol. It was developed in the NETCONF working group and published in December 2006 as RFC 4741.

NETCONF provides mechanisms to install, manipulate, and delete the configuration of network devices. Its operations are realized on top of a simple Remote Procedure Call (RPC) layer. The NETCONF protocol uses an Extensible Markup Language (XML) based data encoding for the configuration data as well as the protocol messages. This in turn is realized on top of the transport protocol.

The NETCONF protocol can be conceptually partitioned into four layers:



Operations

Basic Operations

The base protocol includes the following protocol operations: <get>, <get-config>, <edit-config>, <copy-config>, <delete-config>, <lock>, <unlock>, <close-session>, <kill-session>.

Capabilities

Basic NETCONF functionality can be extended by the definition of NETCONF capabilities. The set of additional protocol features an implementation supports is communicated between the server and the client during the capability exchange portion of session setup. Mandatory protocol features are not included in the capability exchange since they are assumed. RFC 4741 defines a number of optional capabilities including :xpath and :validate.

A capability to support subscribing and receiving asynchronous event notifications is published in RFC 5277. It defines the <create-subscription> operation, which enables creating real-time and replay subscriptions. Notifications are then sent asynchronously using the <notification> construct. The RFC also defines the :interleave capability, which when supported with the basic :notification capability facilitates the processing of other NETCONF operations while the subscription is active.

A capability to support partial locking of the running configuration is defined in RFC 5717. This allows multiple sessions to edit non-overlapping sub-trees within the running configuration. Without this capability, the only lock available is for the entire configuration.

The working group is also working on a new capability to retrieve the schema definitions (XML Schema, Relax NG, etc) that define NETCONF content.

Transport Protocols

NETCONF defines four transport mappings

- SSH (RFC 4742), which is mandatory to implement
- SOAP (RFC 4743)
- BEEP (RFC 4744)
- TLS (RFC 5539)

Content

The content of NETCONF operations is well-formed XML. Most content is related to network management.

The NETMOD working group has completed work to define a "human-friendly" modeling language for defining the semantics of operational data, configuration data, notifications, and operations, called YANG. YANG is defined in RFC 6020, and is accompanied by the "Common YANG Data Types" found in RFC 6021.

During the summer of 2010, the NETMOD working group was re-chartered to work on core configuration models (system, interface, and routing) as well as work on compatibility with the SNMP modeling language.

History

The IETF developed SNMP in the late 1980s and it proved to be a very popular network management protocol. In the early part of the 21st century it became apparent that in spite of what was originally intended, SNMP was not being used to configure network equipment, but was mainly being used for network monitoring. In 2002, the Internet Architecture Board and key members of the IETF's network management community got together with network operators to discuss the situation. The results of this meeting are documented in RFC 3535. It turned out that operators were primarily using proprietary Command Line Interfaces (CLI) to configure their boxes. This had a number of features that the operators liked, including the fact that it was text-based, as opposed to the BER-encoded SNMP. In addition, many equipment vendors did not provide the option to completely configure their devices via SNMP. As operators generally liked to write scripts to help manage their boxes, they did find the CLI lacking in a number of ways. Most notably was the unpredictable nature of the output. The content and formatting of output was prone to change in unpredictable ways.

Around this same time, Juniper Networks had been using an XML-based network management approach. This was brought to the IETF and shared with the broader community.

Collectively, these two events led the IETF to the creation of a protocol which it hopes will better align with the needs of network operators and equipment vendors.

Chapter 6

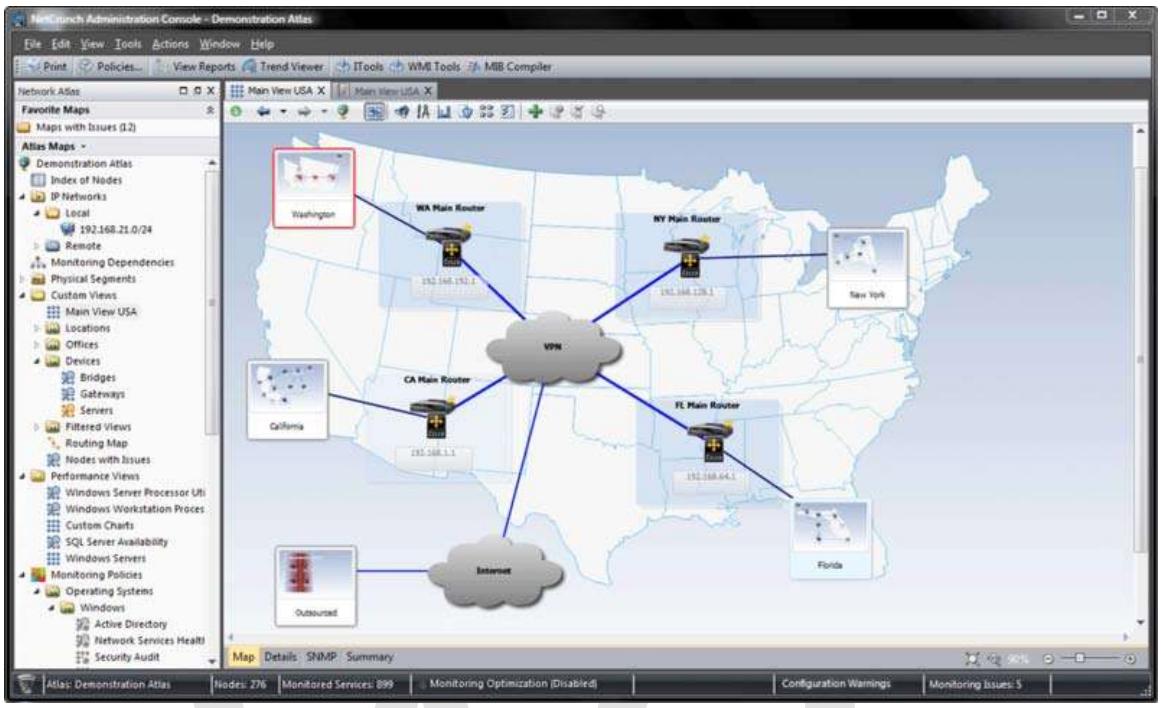
NetCrunch and Netcat

NetCrunch

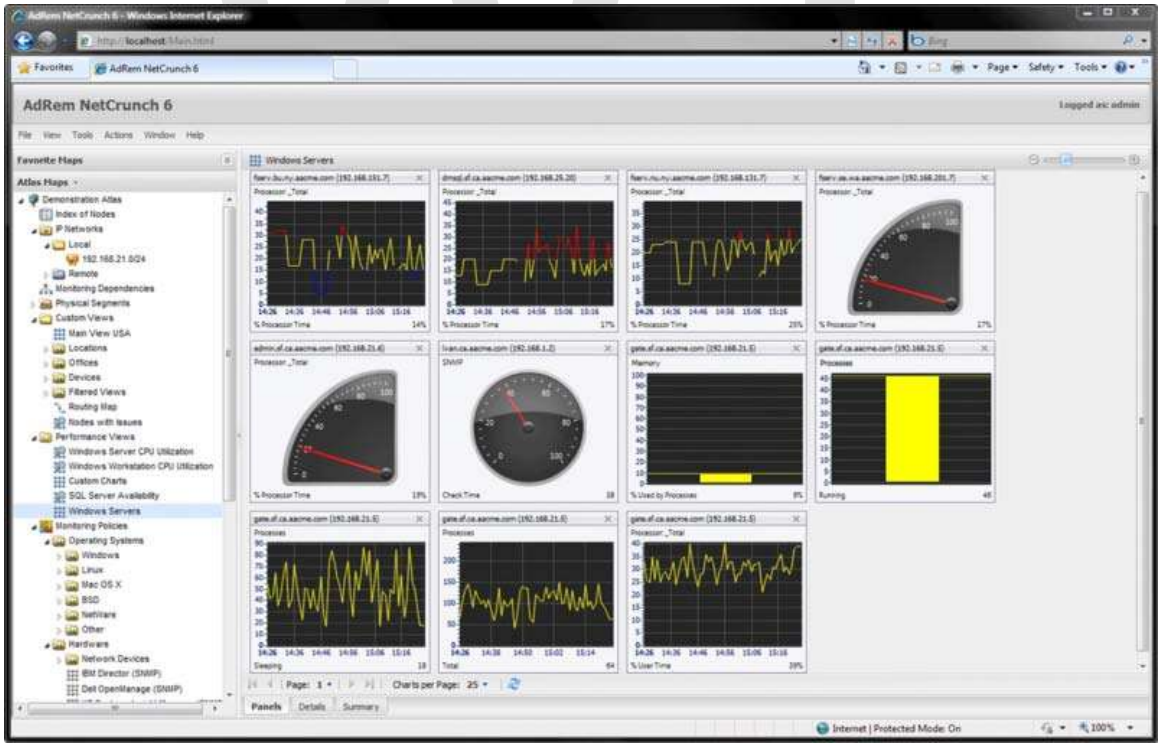
NetCrunch

Developer(s)	AdRem Software, Inc.
Type	Network management system
License	Commercial

AdRem **NetCrunch** is a commercial software solution for agentless, cross-platform network monitoring developed by AdRem Software, Inc. The program monitors 65 network services, Windows applications; Windows, Linux, NetWare, BSD, Mac OS X systems and SNMP (v1-3) devices without agents; centralizes fault management by collecting and alerting on events from sources including Windows Event Log, syslogs, and SNMP traps; presents physical and logical network topology as automatically updated dynamic graphical views.



AdRem NetCrunch 6 custom network map



AdRem NetCrunch 6 performance view

AdRem NetCrunch 6 physical network map

Features

AdRem NetCrunch key features:

Auto Discovery - initial and scheduled discovery and classification of network resources

Network Views - physical and logical network topology including predefined views like: IP networks, Routing Map, Physical Segments, Servers, Maps with Issues, and others.

Event Management - events from Windows, Syslog and SNMP traps

Monitoring Policies - sets of rules defining events to be monitored and data to be collected for later reporting, including predefined policies for:

1. *Operating System* - Windows (Active Directory, Network Services Health, Security Audit, Terminal Services, Windows Server, Basic Windows Monitoring), Linux, Mac OS, BSD, Netware, Other (AIX, AS/400, MIB-II host resources).

2. *Hardware* – Network Devices (Nortel, Alcatel OmniSwitch, Cisco), IBM Director, Dell OpenManage, HP Systems Insight Manager, APC PowerChute.

3. *Applications* – Microsoft (Exchange 2003, IIS 5.0/6.0, ISA Server 2000, ISA Server 2004,MS SQL Server 7.0/2000, MS SQL Server 2005), APC Windows Events, ARCServe, CA eTrust Alert Manager, Lotus Notes 6, McAfee AlertManager, Norton

AntiVirus Corporate Edition, Oracle 9i, Sophos Enterprise Manager, Trend Micro ServerProtect, Veritas Backup Exec.

Availability Monitoring - network devices and services (HTTP, POP3, SMTP, etc.)

Performance Monitoring - real-time statistics, multi-server charts, and performance trends (devices, systems, and applications)

Long-term Trend Analysis – reports generated on demand or delivered on schedule via email.

Remote Access – Remote Administration Console and access via web browser.

User Experience Monitors – advanced monitoring of crucial network services. The program simulates user's action e.g. sends an email for POP3 service monitoring, etc.

Smart Monitoring - limiting monitoring traffic for specific sub networks.

Inventory - Ability to gather the basic inventory information of Windows machines (i.e. mainboard, processor type and memory), with options to schedule inventory audits to be performed, automatically discover and view installed software

Platform

Recommended platforms for NetCrunch Server are Windows 2008 x32/x64 or Windows 2003 SP2 x32/x64. AdRem NetCrunch Remote Administration Console runs on Windows 7/Vista SP2/XP SP3 or Windows Server 2003/2008 (x32/x64).

Editions

There are two editions: Premium and Premium XE. The Premium XE edition is designed for large networks and intended to run on a dedicated Windows 2008 server.

Common Premium XE Usage Scenarios include:

- Network with more than a 1000 nodes for availability monitoring
- Network with more than a 1000 network services for availability monitoring
- Network with more than a 100 servers and routers for performance monitoring
- Network with sub networks
- Network with external servers connected over WAN Links
- Network with CISCO or Nortel switches
- MS SQL, Exchange application performance monitoring and trending

Language versions

There are several language versions of AdRem NetCrunch: english, japanese, polish, german. French, spanish and italian are being prepared.

Technology overview

Client-Server Architecture - user manages NetCrunch server using Remote Administration Consoles.

Multithreading - NetCrunch Premium XE uses multithreading to take advantage of multi-core x64 machines' performance characteristics.

Prioritized monitoring - the program automatically sets up node monitoring order and time upon monitoring dependencies hierarchy.

Event Suppression - in case of failure of an intermediate node, the program suppresses alerts from nodes located beyond that node.

SQL event database (up to 128GB)

Web Access - by implementing AJAX technology the server/browser interactions run asynchronously without reloading the GUI Web Access page

Netcat

netcat

```
% echo "GET / HTTP/1.0¥n" | netcat localhost 80

HTTP/1.1 200 OK
Date: Sat, 07 Jan 2006 08:43:27 GMT
Server: Apache
Last-Modified: Wed, 28 Dec 2005 08:09:31 GMT
ETag: "13c6e-14-1ea644c0"
Accept-Ranges: bytes
Content-Length: 20
Connection: close
Content-Type: text/html

nothing to see here

%
```

Developer(s) *Hobbit*

Stable release	1.10 / March 20, 1996
Operating system	UNIX
Type	Network utility
License	Permissive free software

Netcat is a computer networking service for reading from and writing network connections using TCP or UDP. Netcat is designed to be a dependable “back-end” device that can be used directly or easily driven by other programs and scripts. At the same time, it is a feature-rich network debugging and investigation tool, since it can produce almost any kind of correlation you would need and has a number of built-in capabilities.

In 2000, according to www.insecure.org, **Netcat** was voted the second most functional network security tool. Also, in 2003 and 2006 it gained fourth place in the same category. Netcat is often referred to as a "Swiss-army knife for TCP/IP." Its list of features includes port scanning, transferring files, and port listening, and it can be used as a backdoor.

Features

Some of netcat's major features are:

- Outbound or inbound connections, TCP or UDP, to or from any ports
- Full DNS forward/reverse checking, with appropriate warnings
- Ability to use any local source port
- Ability to use any locally-configured network source address
- Built-in port-scanning capabilities, with randomization
- Built-in loose source-routing capability
- Can read command line arguments from standard input
- Slow-send mode, one line every N seconds
- Hex dump of transmitted and received data
- Optional ability to let another program service established connections
- Optional telnet-options responder
- Featured tunneling mode which allows also special tunneling such as UDP to TCP, with the possibility of specifying all network parameters (source port/interface, listening port/interface, and the remote host allowed to connect to the tunnel.

Examples

Opening a raw connection to port 25 (like telnet)

```
nc mail.server.net 25
```

Setting up a one-shot webserver on port 8080 to present a file

```
{ echo -ne "HTTP/1.0 200 OK\r\n\r\n"; cat some.file; } | nc -l 8080
```

The file can then be accessed via a webbrowser under http://servername:8080/. Netcat only serves the file once to the first client that connects and then exits.

Checking if UDP ports (-u) 80-90 are open on 192.168.0.1 using zero mode I/O (-z)

```
nc -vzu 192.168.0.1 80-90
```

PS: UDP tests will always show as “open”. The -uz argument is useless.

Pipe via UDP (-u) with a wait time (-w) of 1 second to 'loggerhost' on port 514

```
echo '<0>message' | nc -w 1 -u loggerhost 514
```

Port scanning

An uncommon use of netcat is port scanning. Netcat is not considered the best tool for this job, but it can be sufficient (a more advanced tool is Nmap)

```
nc -v -n -z -w 1 192.168.1.2 1-1000
```

The “-n” parameter here prevents DNS lookup, “-z” makes nc not receive any data from the server, and “-w 1″ makes the connection timeout after 1 second of inactivity.

Making any process a server

On a computer A with IP 192.168.1.2:

```
nc -l -p 1234 -e /bin/bash
```

The “-e” option spawns the executable with its input and output redirected via network socket. It connects to computer A from any other computer on the same network:

```
nc 192.168.1.2 1234
ls -las
total 4288
4 drwxr-xr-x 15 imsovain users 4096 2009-02-17 07:47 .
4 drwxr-xr-x 4 imsovain users 4096 2009-01-18 21:22 ..
8 -rw------- 1 imsovain users 8192 2009-02-16 19:30 .bash_history
4 -rw-r--r-- 1 imsovain users 220 2009-01-18 21:04 .bash_logout
...
```

The consequences are that nc is a popular cracker tool as it is so easy to create a backdoor on any computer. On a Linux computer you may spawn /bin/bash and on a Windows computer cmd.exe to have total control over it.

Port Forwarding or Port Mapping

On Linux, NetCat can be used for port forwarding. Below are nine different ways to do port forwarding in NetCat (-c switch not supported though):

```
nc -l -p port1 -c ' nc -l -p port2'
nc -l -p port1 -c ' nc host2 port2'
nc -l -p port1 -c ' nc -u -l -p port2'
nc -l -p port1 -c ' nc -u host2 port2'
nc host1 port1 -c ' nc host2 port2'
nc host1 port1 -c ' nc -u -l -p port2'
nc host1 port1 -c ' nc -u host2 port2'
nc -u -l -p port1 -c ' nc -u -l -p port2'
nc -u -l -p port1 -c ' nc -u host2 port2'
```

Variants

There are several implementations on POSIX systems, including rewrites from scratch like GNU netcat or OpenBSD netcat (this last has also new features like IPv6 support). Mac OS X users can use the Netcat Darwin Port. There is also a Microsoft Windows version of netcat created by Chris Wysopal, and a Cygwin version is available.

Known ports for embedded systems includes versions for the Windows CE (named Netcat 4 wince) or for the iPhone.

BusyBox includes by default a lightweight version of netcat.

Socat is a more complex cousin of netcat. It is larger and more flexible and has more options that must be configured for a given task.

Cryptcat is a version of netcat with integrated transport encryption capabilities.

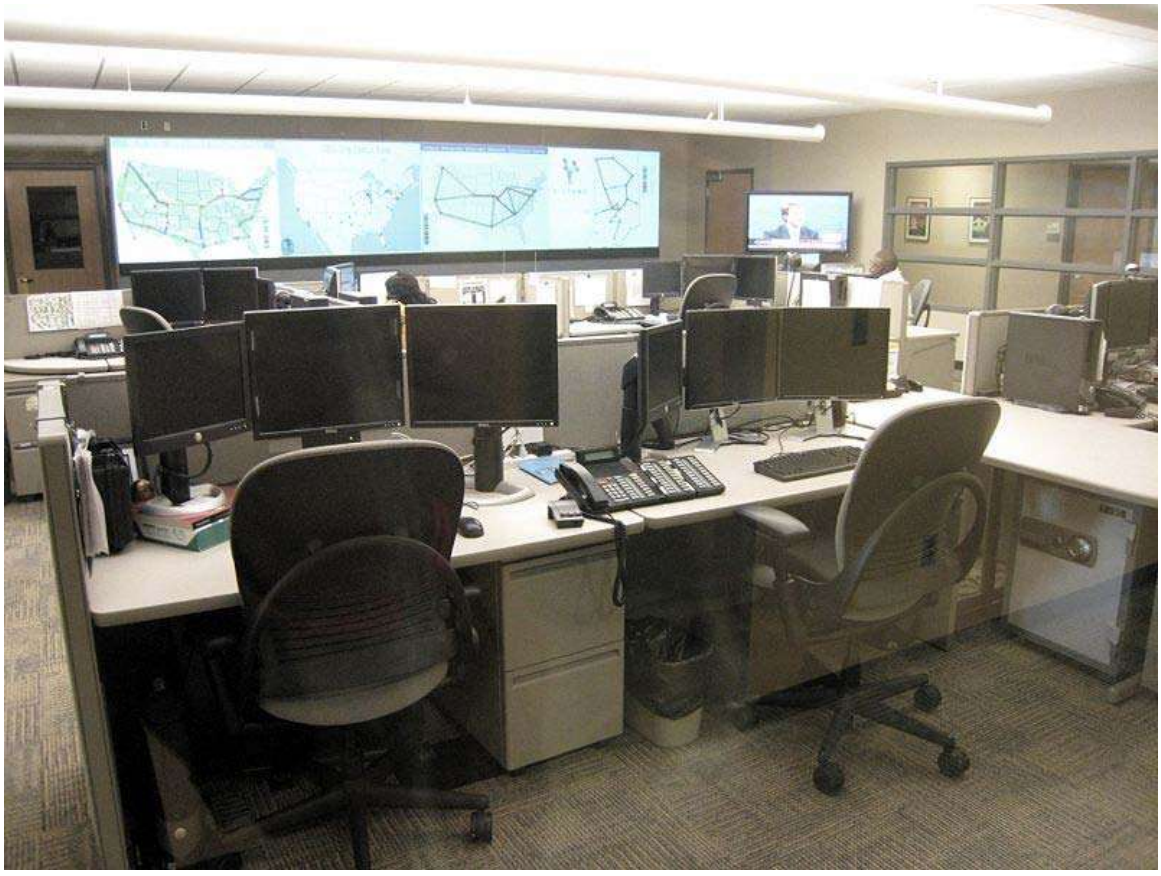
Middle 2005 the Nmap announced another netcat incarnation called Ncat. It features new possibilities such as "Connection Brokering", TCP/UDP Redirection, SOCKS4 client and server support, ability to "Chain" Ncat processes, HTTP CONNECT proxying (and proxy chaining), SSL connect/listen support and IP address/connection filtering. Like Nmap, Ncat is cross-platform.

On some systems, modified versions or similar netcat utilities go by the command name(s) *nc*, *ncat*, *pnetcat*, *socat*, *sock*, *socket*, *sbd*.

Chapter 7

Network Operations Center and OpenNMS

Network operations center



Overview of a typical NOC. Lot of monitors (front), backbone overview (back) and news broadcast on TV-set (right)

A **network operations center** (or **NOC**, pronounced "nok," like the word "knock") is one or more locations from which control is exercised over a computer, television broadcast, or telecommunications network.

Large organizations may operate more than one NOC, either to manage different networks or to provide geographic redundancy in the event of one site being unavailable or offline.

NOCs are responsible for monitoring the telecommunication network for alarms or certain conditions that may require special attention to avoid impact on the network's performance. For example, in a telecommunications environment, NOCs are responsible for monitoring for power failures, communication line alarms (such as bit errors, framing errors, line coding errors, and circuits down) and other performance issues that may affect the network. NOCs analyse problems, perform troubleshooting, communicate with site technicians and other NOCs, and track problems through resolution. If necessary, NOCs escalate problems to the appropriate personnel. For severe conditions that are impossible to anticipate – such as a power failure or optical fiber cable cut – NOCs have procedures in place to immediately contact technicians to remedy the problem.

Technicians in Architel NOC

NOCs are frequently laid out with several rows of desks, all facing a video wall, which typically shows details of highly significant alarms, ongoing incidents and general network performance; a corner of the wall is sometimes used for showing a news or weather TV channel, as this can keep the NOC technicians aware of current events which may have an impact on the network or systems they are responsible for.

The back wall of the NOC is sometimes glazed; there may be a room attached to this wall which is used by members of the team responsible for dealing with serious incidents to meet whilst still able to watch events unfolding within the NOC.

Individual desks are generally assigned to a specific network, technology or area. A technician may have several computer monitors on their desk, with the extra monitors used for monitoring the systems or networks covered from that desk.

NOCs often escalate issues in a hierarchic manner, so if an issue is not resolved in a specific time frame, the next level is informed to speed up problem remediation. Many NOCs have multiple "tiers", which define how experienced/skilled a NOC technician is. A newly-hired NOC technician might be considered a "tier 1", whereas a technician that has been there for several years may be considered a "tier 3" or "tier 4". As such, some problems are escalated within a NOC before a site technician or other network engineer is contacted.

Additionally, the NOC staff may perform extra duties; a network with equipment in public areas (such as a mobile network Base Transceiver Station) may be required to have a telephone number attached to the equipment for emergencies; as the NOC may be the only continuously staffed part of the business, these calls will often be answered there.

The term *NOC* is normally used when referring to telecommunications providers, although a growing number of other organizations such as public utilities (e.g., SCADA) and private companies also have such centers, both to manage their internal networks and to provide monitoring services.

The location housing a NOC may also contain many or all of the primary servers and other equipment essential to running the network, although it is not uncommon for a single NOC to monitor and control a number of geographically dispersed sites.

In broadcast television

Operations center during the Athens 2004 Olympic games

NOCs at television broadcast facilities are responsible for the technical and operational overview of all broadcast network services, including monitoring, correcting, and troubleshooting day-to-day issues.

Duties that fall under broadcast NOCs include:

- Serial Digital Video, ASI, Multiplexed and DVB data streams technical monitoring
- Networking
- RF and IF distribution
- Monitoring turnaround video services

OpenNMS

OpenNMS is an enterprise grade network monitoring and network management platform developed under the free software or open source model. It consists of a community-supported, free-software project as well as an organization offering commercial services, training and support.

The goal is for OpenNMS to be a truly distributed, scalable platform for all aspects of the FCAPS network management model, and to make this platform available to both free software / open-source and commercial applications.

All code associated with the project is available under the GNU General Public License.

OpenNMS is currently maintained by Tarus Balog, The OpenNMS Group, and The Order of The Green Polo.

Features

- Service polling - determining service availability and latency, including distributed measurement of availability and latency, and reporting on the results
- Data collection - collecting, storing and reporting on data collected from nodes via protocols including SNMP, JMX, HTTP, Windows Management Instrumentation, JDBC, and NSClient
- Thresholding - evaluating polled latency data or collected performance data against configurable thresholds, creating events when these are exceeded or rearmed
- Event management - receiving events, both internal and external, including via SNMP traps
- Alarms and automations - reducing events according to a reduction key and scripting automated actions centered around alarms
- Notifications - sending notices regarding noteworthy events via e-mail, XMPP, or other means

Supported platforms

As OpenNMS is written mainly in Java; it can theoretically run on any system that supports a 1.5 (or higher) SDK.

The following operating systems are supported:

- Linux
- Solaris
- Mac OS X
- Microsoft Windows
- FreeBSD

Awards

On August 11, 2005, OpenNMS won the *Product Excellence Award* in the category *Best Systems Management Tools* at LinuxWorld Conference and Expo . .

The other 3 nominees were:

- Userful DiscoverStation 4.0
- IBM Tivoli Intelligent Orchestrator
- Novell ZENworks 7 Linux Management

OpenNMS won the Gold award in the *Network and IT management platforms* category of SearchNetworking.com's Product Leadership Awards 2007 , beating out HP OpenView and IBM Tivoli.

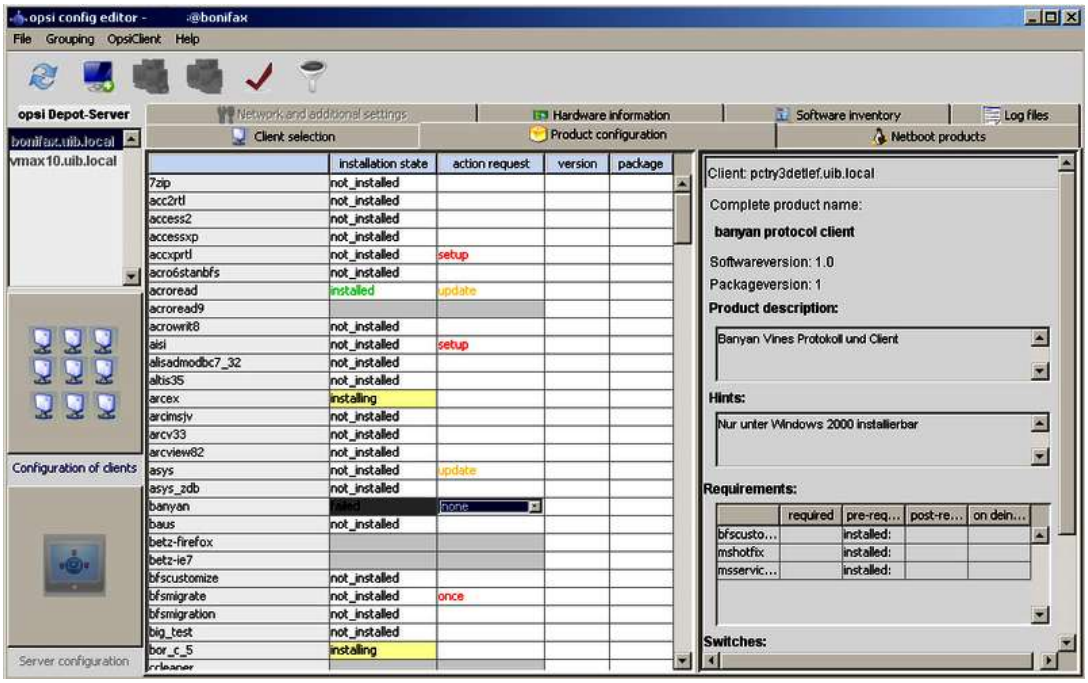
In March 2008, OpenNMS again appeared in the SearchNetworking.com Product Leadership Awards, this time taking bronze or third place in the *Applications and network management* category.

In August 2009, OpenNMS won in the *Networking and network management* category of InfoWorld's BOSSIE (Best of Open Source Software) awards.

Chapter 8

Opsi

opsi



opsi management interface

Developer(s)	uib gmbh, Mainz, Germany
Stable release	4.0 / 1. oct 2010
Written in	Python Java
Operating system	Linux, Windows
Available in	English, French, German, Spanish, Turkish
Type	Network management System administration
License	GPL

Opsi (open pc server integration) is a software distribution and management system for Windows Clients, based on Linux servers.

Features

The core features of opsi are:

- Automatic operating system installation (OS Deployment)
- Software distribution
- Patch-Management
- Inventory (Hardware and Software)
- License Management / Software Asset Management
- Administrative Tasks (Configuration Management)

Developer and maintainer is the company uib gmbh in Mainz, Germany. The product is Open Source licensed under the GNU General Public License.

Supported client operating systems are Windows 2000, Server 2000, Windows XP, Server 2003. Support for Windows Vista, Server 2008 and Windows 7 is available, but subject to a Co-funding Project and not released as open source yet. For the installation of an opsi-server there are packages available for the Linux distributions Debian, Ubuntu and Suse.

Automatic operating system installation

Via management interface a client may be selected for OS-Installation. If the client boots via PXE it loads a boot image from the opsi-depotserver. This bootimage prepares the hard disk, copies the required installation files, drivers and the opsi client agent and starts finally an unattended OS-Installation. An OS-installation via Disk image is also supported.

Software distribution

For the automatic software distribution a client agent (the opsi-preloginloader) has to be installed on the client. This client agent starts at boot time, connects to the opsi configuration-server and starts if scheduled a script driven installation program (opsi-winst) which installs the required software on the client. During the installation process the user login can be blocked for integrity reasons. To integrate a new software packet into the software deployment system, a script must be written to specify the installation process. This script provides all the information on how this software packet has to be installed silent or unattended or by using tools like AutoIt or Autohotkey.

Patch-Management

The mechanism of the software deployment can also be used to deploy software patches and hotfixes.

Inventory (Hardware and Software)

The Hard- and Software Inventory uses also the opsi client agent. The Hardware Information is collected via calls to WMI while the Software Information is gathered from the registry. The inventory data are sent back to the opsi-configuration-server by web service.

License Management / Software Asset Management

The opsi License Management module supports the administration of different kinds of licenses like Retail, OEM and Volume licenses. It keeps track of the licenses that are used while the software deployment. Using the connection between the License Management and the software inventory, Software Asset Management reports on the number of free and installed licenses can be generated. The License Management module is part of a Co-funding Project and not released as open source yet.

Administrative tasks (Configuration Management)

The opsi client runs a script in a administrative context, which can be used also for configuration purpose. The opsi script interpreter supports:

- Start of programs and exit code detection
- Detection of the running OS, language and national settings as well as evaluation of Ini-files, text files, registry entries and environment variables
- Editing of registry, start menu and desktop entries, Ini-files, XML files and text files
- Editing of user specific profile registry entries and files (in case of not using 'roaming profiles')
- Calling external programs and scripts, catch and provide their output as variables for further processing
- File copy with or without version control

opsi-server

The opsi server provides the following services:

- The configuration-server stores the configuration data for the clients and the software packages. For the data storage a file- or LDAP based storage can be chosen. For administration of these data the configuration-server provides a graphical management interface via https as well as a command line interface.
- The depot-server stores software packages that may be installed by the clients. To provide support for multiple locations, multiple depot-servers may be controlled by one configuration-server.
- A TFTP-Server provides the boot images for the OS-Installations.
- A DHCP-Server may be integrated in the opsi-server

Management-Interface

For the management of opsi there is a graphical user interface, which is available as application as well as Browser Applet. Management is also possible with a command line tool or via web service.

Co-funding Projects

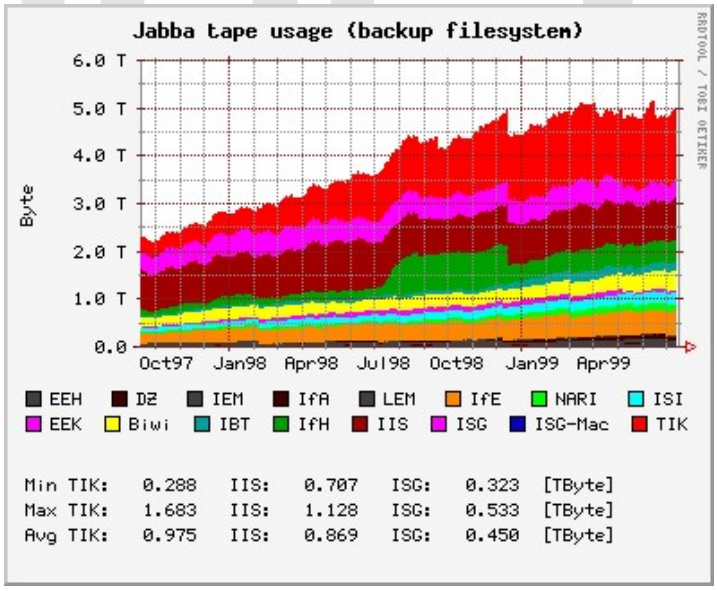
Even though opsi is open source, there are some components which are not free at the moment. These components are developed in a co-funding project which means that until the complete development costs are paid by co-funders, they are only allowed to use by the co-funders or for evaluation purposes.

Chapter 9

RRDtool, Rsyslog and Syslog-ng

RRDtool

RRDtool	
Original author(s)	Tobi Oetiker
Stable release	1.4.4 / July 5, 2010
Written in	C
License	GNU General Public License



RRDtool has a graph function, which presents data from an RRD in a customizable graphical format

RRDtool (acronym for **r**ound-**r**obin **d**atabase **tool**) aims to handle time-series data like network bandwidth, temperatures, CPU load etc. The data are stored in a round-robin database (circular buffer), thus the system storage footprint remains constant over time.

It also includes tools to extract **RRD** data in a graphical format.

Tobi Oetiker wrote RRDtool as a replacement for MRTG and licenses it as free software under the terms of the GNU General Public License (GPL).

Bindings exist for Perl, Python, Ruby, Tcl, PHP and Lua.

General data storage

RRDtool assumes time-variable data in intervals of a certain length. This interval, usually named **step**, is specified upon creation of an RRD file and cannot be changed afterwards. Because data may not always be available at just the right time, RRDtool will automatically interpolate any submitted data to fit its internal time-steps.

The value for a specific step, that has been interpolated, is named a primary data point (**PDP**). Multiple primary data points may be consolidated according to a consolidation function (**CF**) to form a consolidated data point (**CDP**). Typical consolidation functions are average, minimum, maximum.

After the data have been consolidated, the resulting CDP is stored in a round-robin archive (**RRA**). A round-robin archive stores a fixed amount of CDPs and specifies how many PDPs should be consolidated into one CDP and which CF to use. The total time covered by an RRA can be calculated as follows:

```
time covered = (#CDPs stored) * (#PDPs per CDP) * step
```

After this time the archive will "wrap around": the next insertion will overwrite the oldest entry. This behavior is sometimes referred to as "round-robin" and is the reason for the program's name.

To cover several timespans and/or use several consolidation functions, an RRD file may contain multiple RRAs. The data retrieval function of RRDtool automatically selects the archive with the highest resolution that still covers the requested timespan. This mechanism is also used by RRDtool's graphing subsystem.

Release history

Colour	Meaning
Red	Release no longer supported
Green	Release still supported
Blue	Future release

RRDTool is sponsored since 1.2, each release comes with a list of sponsors.

The following table contains the **release history of RRDtool**, showing most of its release versions.

Version number	Date	Links	Notable changes
1.0	July 16, 1999	Full release notes, Announce	First release. Basically MRTG "done right".
1.1	April 25, 2005	Full release notes, Announce	libart; output EPS, PDF & SVG; VDEF; trends; percentiles; updatev; Holt-Winters Forecasting; COMPUTE; .rrd format change.
1.3	June 11, 2008	Full release notes, Announce	Safer & faster file access; cairo/pango; anti-aliasing; TEXTALIGN; dashed lines; new HWPREDICT; libxml; i18n; XML dump;
1.4	October 27, 2009	Full release notes, Announce	Caching daemon; VDEF PERCENTNAN; CDEF PREDICT & PREDICTSIGMA; libDBI; graph legends positioning; Lua bindings; 3D border width; and more ...

Rsyslog

Rsyslog	
Original author(s)	Rainer Gerhards
Stable release	5.6.2 / November 30, 2010; 12 days ago
Preview release	6.1.1 / November 30, 2010; 12 days ago
Written in	C
Operating system	Unix-like
Type	System logging
License	GNU General Public License v3

Rsyslog is an open source program for forwarding log messages in an IP network for UNIX and Unix-like systems. It implements the basic syslog protocol, extends it with content-based filtering, rich filtering capabilities, flexible configuration options and adds important features such as using TCP for transport.

Protocol

Rsyslog uses the quasi-standard BSD syslog protocol, specified in RFC 3164. As the text of RFC 3164 is just a vague informational description and not a standard, various incompatible extensions of it emerged. Rsyslog supports many of these extensions. The format of relayed messages can be customized.

The most important extensions of the original protocol supported by rsyslog are:

- ISO 8601 timestamp with millisecond granularity and timezone information
- the addition of the name of relays in the host fields to make it possible to track the path a given message has traversed
- reliable transport using TCP
- support GSS-API and TLS
- logging directly into various database engines.
- support for the upcoming new IETF syslog RFC series
- support for buffered operation modes where messages are buffered locally if the receiver is not ready

History

The rsyslog project began in 2004, when Rainer Gerhards, the primary author of rsyslog, decided to write a new strong syslog daemon to compete with syslog-ng, because; and according to the author "A new major player will prevent monocultures and provide a rich freedom of choice."

Distributions

rsyslog is available for a number of Unix systems and Linux distributions, among others:

- Fedora (In November 2007, rsyslog has become the default syslogd for the Fedora project) Fedora was the first major distribution to adopt this software.
- openSUSE (default since 11.2; November 2009)
- Debian GNU/Linux (As of Debian 5.0, rsyslog has become the default syslog)
- Ubuntu
- Red Hat Enterprise Linux (from the upcoming version 6)
- Solaris
- FreeBSD
- OpenBSD
- Gentoo

Related RFCs and working groups

- RFC 3164 - The BSD syslog Protocol (obsoleted by RFC 5424)
- RFC 5424 - The Syslog Protocol (obsoletes RFC 3164)
- RFC 5425 - Transport Layer Security Mapping for Syslog

- RFC 5426 - Transmission of Syslog Messages over UDP

Syslog-ng

syslog-ng	
Original author(s)	Balázs Scheidler
Initial release	1998
Stable release	3.2.1 / November 30, 2010; 12 days ago
Operating system	Unix-like
Type	System logging
License	GNU Lesser General Public License(core) GNU General Public License version 2(plugins)

syslog-ng is an open source implementation of the Syslog protocol for Unix and Unix-like systems. It extends the original syslogd model with content-based filtering, rich filtering capabilities, flexible configuration options and adds important features to syslog, like using TCP for transport. As of today syslog-ng is developed by Balabit IT Security Ltd. It have two editions with common codebase. The fist is called syslog-ng OSE (with the license LGPL) and have additional plugins (modules) under proprietary license. This edition is called Premium Edition (PE).

Protocol

syslog-ng uses the quasi-standard BSD syslog protocol, specified in RFC 3164. As the text of RFC 3164 is vague and is just an informational description and not a standard, various incompatible extensions of it emerged. Since version 3.0 also supports the standard syslog protocol specified in RFC 5424 which was released in 2009. syslog-ng tries hard to interoperate with a wide variety of devices, and the format of relayed messages can be customized.

The most important extensions of the original protocol endorsed by syslog-ng are:

- ISO 8601 timestamp with millisecond granularity and timezone information

- the addition of the name of relays in the host fields to make it possible to track the path a given message has traversed
- reliable transport using TCP
- TLS encryption (Since 3.0.1 in OSE)

History

The syslog-ng project began in 1998, when Balázs Scheidler, the primary author of syslog-ng, ported the existing nsyslogd code to Linux. The 1.0.x branch of syslog-ng was still based on the nsyslogd sources and are available in the syslog-ng source archive.

Right after the release of syslog-ng 1.0.x, a reimplementation of the code base started to address some of the shortcomings of nsyslogd and to address the licensing concerns of Darren Reed, the original nsyslogd author. This reimplementation was named stable in the October of 1999 with the release of 1.2.0. This time around, syslog-ng depended on some code originally developed for lsh by Niels Möller.

Three major releases (1.2, 1.4 and 1.6) were using this code base, the last release of the 1.6.x branch in February 2007. In this period of about 8 years, syslog-ng became one of the most popular alternative syslog implementations.

In a volunteer based effort, yet another rewrite was started back in 2001, dropping lsh code and using the more widely available GLib library. This rewrite of the codebase took its time, the first stable release of 2.0.0 happened in October 2006.

Development efforts are focused on improving the 2.0.x branch; support for 1.6.x is expected to be dropped in the near future (as of May 2007). Balabit, the company behind syslog-ng, started a parallel, commercial fork of syslog-ng, called syslog-ng Premium Edition. Portions of the commercial income are used to sponsor development of the free version.

Syslog-ng version 3.0 was released in the fourth quarter of 2008.

Starting with the 3.0 version developments efforts were parallel on the Premium and on the Open Source Editions. PE efforts were focused on quality, transport reliability, performance and encrypted log storage. The Open Source Edition efforts focused on improving the flexibility of the core infrastructure to allow more and more different, non-syslog message sources.

Both the OSE & PE forks produced two releases (3.1 and 3.2) in 2010.

Features

syslog-ng has a much larger scope than merely transporting syslog messages and storing them to plain text log files:

- the ability to format log messages using UNIX shell-like variable expansion;
- the use of this shell-like variable expansion when naming files, thus covering thousands of destination files with a single statement;
- the ability to send log messages to local applications;
- ability to message flow-control in network transport;
- logging directly into a database (since syslog-ng OSE 2.1);
- rewrite portions of the syslog message with set and substitute primitives (since syslog-ng OSE 3.0);
- classify incoming log messages and at the same time extract structured information from the unstructured syslog message (since syslog-ng OSE 3.0);
- generic name-value support: each message is just a set of name-value pairs, which can be used to store extra information (since syslog-ng OSE 3.0);
- the ability to process structured message formats transmitted over syslog, like extract columns from CSV formatted lines (since syslog-ng OSE 3.0);
- the ability to correllate multiple incoming messages to form a more complex, correllated event (since syslog-ng OSE 3.2);

Distributions

syslog-ng is part of a number of different GNU/Linux and Unix distributions. Some distributions install it as the default system logger, others only provide a package and an upgrade path from the standard syslogd.

Among others:

- openSUSE used it prior to openSUSE 11.2
- Debian GNU/Linux used before version 5.0 syslogd and klogd (Lenny (5.0) uses Rsyslog)

- Gentoo Linux
- Fedora used it prior to Fedora 10
- Arch Linux
- Hewlett-Packard's HP-UX
- FreeBSD
- A Cygwin port is available for Microsoft Windows

Portability

syslog-ng is highly portable to many Unix systems, old and new alike. A list of the currently known to work Unix versions are found below:

- Linux on i386, SPARC and x86-64 CPUs
- FreeBSD 6.x, 7.x on i386 CPUs
- AIX 5 on IBM POWER CPUs
- HP-UX 11iv1, 11iv2 on PA-RISC and Itanium CPUs
- Solaris 8, 9, 10 on SPARC, x86-64 and i386 CPUs

- Tru64 5.1b on Alpha CPUs

The list above is based on BalaBit's current first hand experience, other platforms may also work, but your mileage may vary.

Related RFCs & working groups

- RFC 3164 - The BSD syslog protocol
- RFC 5424 - The Syslog Protocol
- RFC 5425 - Transport Layer Security (TLS) Transport Mapping for Syslog
- RFC 5426 - Transmission of Syslog Messages over UDP

Log Viewers

Name	Free	Description
XLog-Solution	Free and Commercial	Log aggregation from distributed systems. Web-based dashboard with control over filters. Email alerts. Interface with Jira.

Chapter 10

Verax NMS and Zabbix

Verax NMS

Verax NMS	
Developer(s)	Verax Systems
Platform	cross-platform
Type	Network & application monitoring

Verax NMS is a highly scalable, integrated network management and IT service assurance solution for cross-silos management and monitoring of networks, data centers and applications developed by Verax Systems. Verax NMS helps management of large enterprises, IT infrastructure and networks by providing scalability and flexibility in mission-critical environments.

Overview

Verax NMS provides full FCAPS functionality for both infrastructure and application monitoring. The key features include node and application discovery, visualization of network topologies and business aspects with drill-down capabilities, user-defined dashboards, correlation and management of events, alarm handling, gathering performance data and others.

Supported hardware and applications

Verax NMS architecture is plugin based. Each managed entity (either a node or an application) is designed as a plug-in.

Supported hardware

Verax NMS supports over 1900 types of network elements from vendors such as Cisco, HP, Brocade/Foundry, Juniper, Adva and others, as well as Unix and Windows hosts. Other devices may be managed using the built-in, standard SNMP MIB-2 plugin.

Supported applications

Verax NMS provides management plugins for:

- Database systems such as Oracle, MySQL, Microsoft SQL Server and others.
- Application servers such as Apache Tomcat, IBM WebSphere, JBOSS and others.
- Web-servers: Apache and Microsoft Internet Information Server.
- Microsoft business applications: Exchange and Active Directory.
- Messaging systems: ActiveMQ.

Virtualization

Verax NMS is virtualization-aware and provides management plugins for VM*Ware and Citrix.

Business rules and IT automation

Verax NMS is equipped with a JRuby-based business rules/scripting engine allowing to create IT automation scenarios such as: “if a service does not respond, restart the virtual machine hosting the service”.

SOA readiness

Verax NMS supports Service Oriented Architecture (SOA) and is open for integration with other systems via SOAP or RMI. It is also compatible with Enterprise Service Bus products such as TIBCO.

Technology and system requirements

Verax NMS has been implemented as a three-tier Java Enterprise application using Adobe Rich Internet technology as the front end.

Server side

Commercial		
Application servers	**Operating Systems**	**Databases**
• IBM WebSphere • Oracle	• Oracle/Sun Solaris • IBM AIX • Microsoft Windows	• Oracle • IBM DB2 • Microsoft SQL

• Sun	• AS/400 • Linux (commercial editions)	Server
Open source		
Application servers	**Operating Systems**	**Databases**
• Apache Tomcat • JBOSS	• Linux • FreeBSD • Open Solaris	• MySQL

Verax NMS can also run in distributed environments for improved performance and fault tolerance.

Client side

Web browser supporting Adobe Flash technology version 9 or higher, such as: Internet Explorer, Firefox, Google Chrome, Opera, Safari and others.

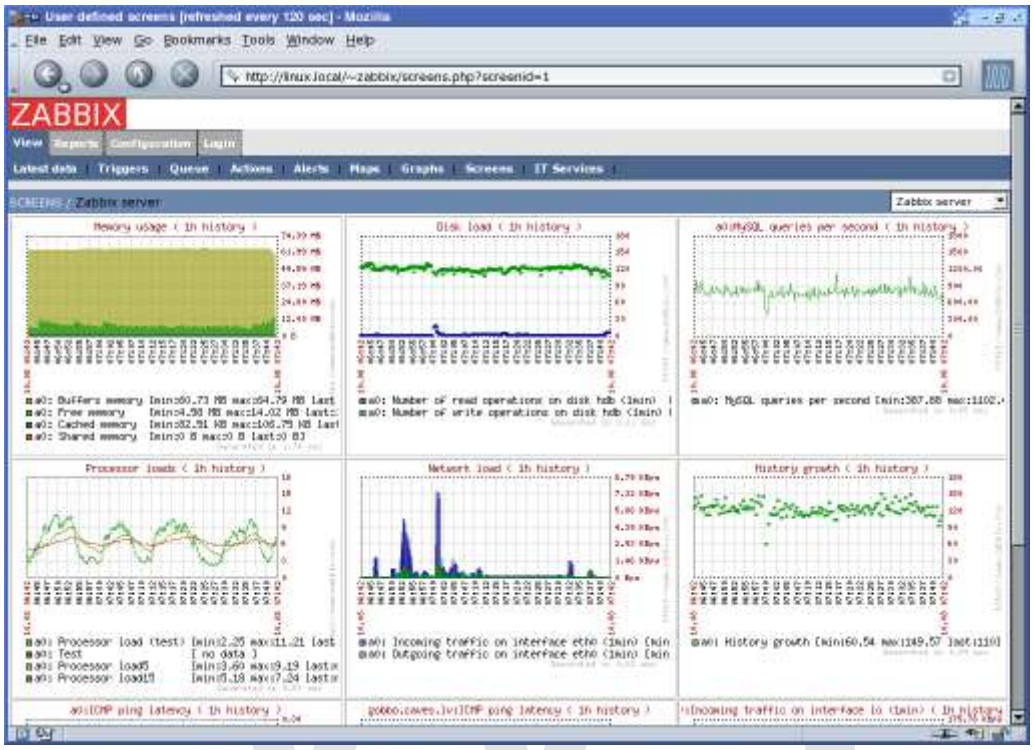
Mobile client

Verax NMS mobile client application operates on following platforms:

- Android (2.0 and above).
- iPhone (iOS 3.0 and above).
- Windows Mobile (5.0 and above).

Zabbix

Zabbix



ZABBIX 1.1 alpha 6 running under Linux

Developer(s)	Zabbix SIA
Stable release	1.8.3 / August 16, 2010; 3 months ago
Preview release	1.9.0 / October 26, 2010; 47 days ago
Development status	active
Written in	C (server), PHP (frontend)
Operating system	Cross-platform
Type	Network management system
License	GNU General Public License

Zabbix is a network management system created by Alexei Vladishev. It is designed to monitor and track the status of various network services, servers, and other network hardware.

It uses MySQL, PostgreSQL, SQLite or Oracle to store data. Its backend is written in C and the web frontend is written in PHP. Zabbix offers several monitoring options. Simple checks can verify the availability and responsiveness of standard services such as SMTP or HTTP without installing any software on the monitored host. A Zabbix agent can also be installed on UNIX and Windows hosts to monitor statistics such as CPU load, network utilization, disk space, etc. As an alternative to installing an agent on hosts, Zabbix includes support for monitoring via SNMP, TCP and ICMP checks, IPMI and custom parameters. Zabbix supports a variety of real-time notification mechanisms, including XMPP.

Released under the terms of version 2 of the GNU General Public License, Zabbix is free software.

History

Zabbix started as an internal software project in 1998. After 3 years, in 2001, it was released to the public under GPL. It took three more years until first stable version, 1.0, was released in 2004.

Timeline of releases

Date	Release
Zabbix 1.0	
1998	Zabbix started as an internal project in a bank by Alexei Vladishev
7 April 2001	Zabbix 1.0alpha1 is released as GPL
23 March 2004	Zabbix 1.0 released
Zabbix 1.1	
6 February 2006	Zabbix 1.1 released
Zabbix 1.4	
29 May 2007	Zabbix 1.4 released
Zabbix 1.6	
11 September 2008	Zabbix 1.6 released
Zabbix 1.8	
7 December 2009	Zabbix 1.8 released

Development

Zabbix today is primarily developed by a dedicated company, Zabbix SIA.

Source code

Zabbix consists of several separate modules:

- Server

- Agents
- Frontend

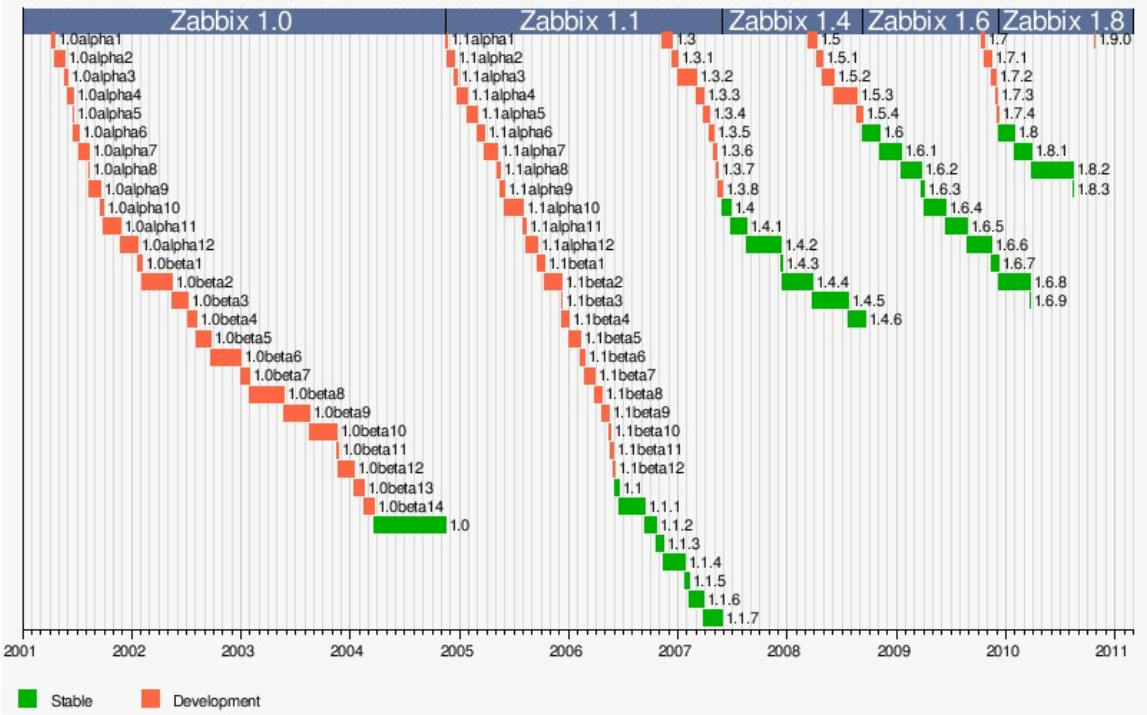
While server and agents are written in C, frontend is implemented in PHP and Javascript.

Releases

Since the first stable version was released as 1.0, Zabbix versioning has only increased minor version numbers. Each minor release actually implements many new features, while change level releases mostly introduce bugfixes.

Zabbix version numbering scheme has changed. While first two stable branches were 1.0 and 1.1, after 1.1 it was decided to use odd numbers for development versions and even numbers for stable versions. As a result, 1.3 followed 1.1 as a development release to be released as 1.4.

Timeline

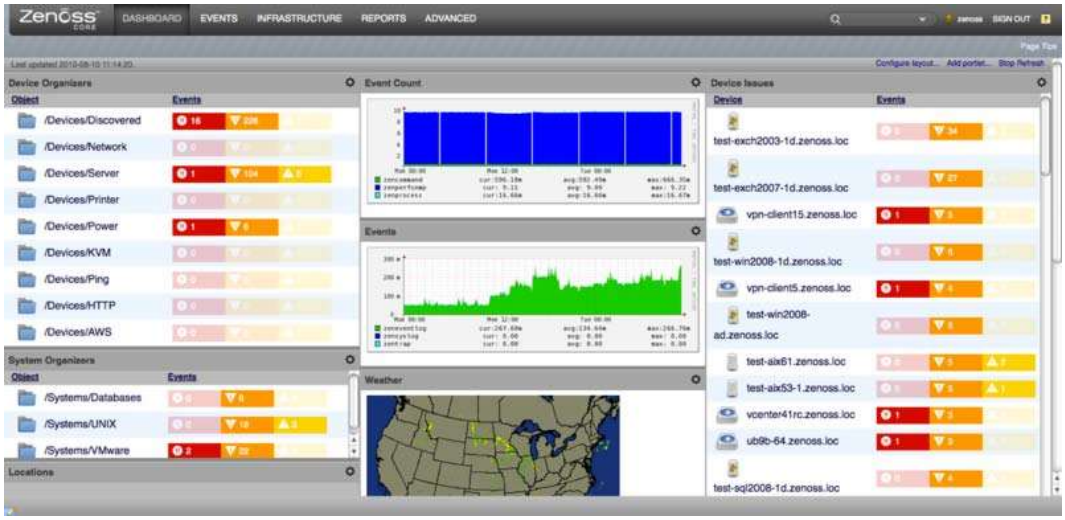


Chapter 11

Zenoss and Zentyal

Zenoss

Zenoss



Zenoss running under Linux

Developer(s)	Zenoss Inc.
Written in	Python
Type	Network management system
License	GNU General Public License

Zenoss (Zenoss Core) is an open source application, server and network management platform based on the Zope application server. Released under the GNU General Public License (GPL) version 2, Zenoss Core provides a web interface that allows system administrators to monitor availability, inventory/configuration, performance and events.

Development of Zenoss Core began in 2002 and in August 2005 the corporate patron of the project Zenoss, Inc. was founded. Zenoss, Inc. sponsors the development of Zenoss Core and sells an enterprise version based on the core version.

Project Milestones

Zenoss maintains an active developer community. Notable project milestones include:

2002
: Erik Dahl began development on Zenoss.

August 2005
: Erik Dahl and Bill Karpovich form Zenoss Inc.

March 2006
: Zenoss made available on SourceForge.net.

November 2006
: Zenoss Core Version 1.0 released.

June 2007
: Zenoss Core 2.0 released.

July 2007
: Zenoss Enterprise 2.0 released.

October 2007
: Zenoss Core 2.1 released.

November 2007
: Zenoss Enterprise 2.1 released.

May 2008
: Zenoss Core 2.2 released.

November 2008
: Zenoss Core 2.3 released.

May 2009
: Zenoss Core 2.4 released.

October 2009
: Zenoss Core 2.5 released.

July 2010
: Zenoss Core 3.0 released.

Technology Overview

Zenoss Core combines original programming and several open source projects to integrate data storage and data collection processes with a web-based user interface.

Zenoss Core is built upon the following open source technologies:

- Zope Application server: An object-oriented web server written in Python.
- Python: Extensible programming language.
- Net-SNMP: Monitoring protocol that collects systems status information.
- RRDtool: Graph and log time series data.

- MySQL: A popular open source database.
- Twisted: An event-driven networking engine written in Python.

Zenoss Core provides the following capabilities:

- Monitoring availability of network devices using SNMP, SSH, WMI
- Monitoring of network services (HTTP, POP3, NNTP, SNMP, FTP)
- Monitoring of host resources (processor, disk usage) on most network operating systems.
- Time-series performance monitoring of devices
- Extended Microsoft Windows monitoring via Windows Management Instrumentation using SAMBA and Zenoss open source extensions
- Event management tools to annotate system alerts
- Automatically discovers network resources and changes in network configuration
- Alerting system provides notifications based on rule sets and on-call calendars
- Supports Nagios plug-in format

Platform

Zenoss Inc. lists the following operating systems for Zenoss Core on their download page:

- Red Hat Enterprise Linux / CentOS (4, 5)
- Fedora (9, 10, 11)
- Ubuntu (6.06, 8.04)
- Debian (5)
- SuSE (10.X)
- OpenSUSE (10.3, 11.1)
- Mac OS X (10.5 Intel, PPC from source, 10.6)
- VMWare Appliance
- FreeBSD (6.x and 7.x from source)
- Solaris (10 from source)
- Gentoo (from source)

Other Linux versions will work with the stack installers as well as source for any other Unix systems.

A web-based portal provides operating system agnostic access to configuration and administration functions. Both Firefox and Internet Explorer are supported.

ZenPacks

ZenPacks provide a plug-in architecture that allows community members to extend Zenoss's functionality. The authors are free to choose how they license their individual ZenPacks. ZenPacks are encapsulated in Python eggs and provide instrumentation and reports for monitored infrastructure components.

Enterprise

The enterprise version builds on the core version by providing commercial support and additional features, such as synthetic web transactions and global dashboards. "In the enterprise edition," writes Sean Michael Kerner, "Zenoss is adding something it calls end-user experience monitoring which is intended to more accurately simulate end-user application activity." Kerner continues, "Enterprise users also get certified application monitors specifically geared for Microsoft SQL and Exchange."

Related products

Zenoss competes with other open source and proprietary enterprise systems management products. Open source systems management products are available from GroundWork Open Source, and Hyperic. In an interview with Jack Loftus of SearchEnterpriseLinux.com, Bill Karpovich explains what makes Zenoss different:

"Companies like GroundWork are similar to the Red Hat approach, where a company gathers up the pieces and puts support behind it. Our approach is we have always had the code and we are in control of its roadmap and indemnification. The Hyperic model is where a company comes from a commercial background and makes some of the code open source."

Closed-source vendors include BMC, HP OpenView, Orion, AdRem NetCrunch and CA.

Industry reviews

In a Network Computing review, Jeff Ballard singles out the Zenoss Core 2.0 user interface and event management system as highlights. Of the event management system, Ballard says, "By aggregating all events through a single rules-processing engine, Zenoss Core eliminates duplication, making for a manageable user interface."

In his review, Ballard finds the installation troubling. "Unfortunately, getting started was challenging as Zenoss provided no context-sensitive help to guide us through a truly staggering number of configuration options."

In the "Clear Choice Tests" Network World reviewer Barry Nance offers the following praise for Zenoss Core 2, "Even more impressive than its discovery of our network is its remediation features, which can automatically execute start or stop operations for a Windows service, for example." Nance's review finds that "Zenoss Core doesn’t support as many diverse devices as HP OpenView or Argent Extended Technologies, nor does it monitor Microsoft Exchange or SQL Server as closely as a commercial tool does."

SYS-CON Media awards Zenoss Core the 2007 Enterprise Open Source Reader's choice award for best Linux systems management software. Reader choice awards are nominated and voted on by the community of *Enterprise Open Source Magazine* readers.

zentyal

Zentyal (formely eBox Platform)

Zentyal Dashboard

Company / developer	eBox Technologies
OS family	Unix-like
Working state	current
Source model	Open Source

Initial release	July 15, 2009
Latest stable release	2.0 / September 1, 2010; 3 months ago
Available language(s)	English (partially translated in 27 languages)
Update method	APT (Web front-end available through ebox-software)
Package manager	dpkg
Supported platforms	i386(x86), amd64(x86-64)
Default user interface	Web user interface
License	GPL

Zentyal (formely eBox Platform) is an open source unified network server (or an Unified Network Platform) for SMEs. Zentyal can act as a Gateway, Network Infrastructure Manager, Unified Threat Manager, Office Server, Unified Communications Server or a combination of them. Besides, Zentyal includes a development framework to ease the development of new Unix based services.

The project's source code is available under terms of the GNU General Public License, as well as under a variety of proprietary agreements. Zentyal is owned and sponsored by a single for-profit firm, the Spanish company eBox Technologies S.L., which holds the copyright to the codebase.

Zentyal (eBox Platform) development was first published in 2005 as an open-source, collaborative project of two companies. On 16 November 2006 Zentyal (eBox Platform) was officially approved as a NEOTEC project, receiving public funds from the CDTI (a Spanish public organisation, under the Ministry of Industry, Commerce and Tourism) to complete the development of version 1.0. Zentyal (eBox Platform) was first included in Ubuntu in 2007, in the Gutsy Gibbon Tribe 3, the third alpha release of Ubuntu 7.10. The first stable release candidate of Zentyal (eBox Platform 1.0) was published in 2008.

Features

As of September 2010 **Zentyal 2.0** offers the following features:

- **Networking**
  - Firewall and routing
    - Filtering
    - NAT and port redirections
    - VLAN 802.1Q
    - Support for multiple PPPoE and DHCP gateways
    - Multi-gateway rules, load balancing and automatic failover
    - Traffic shaping (with application layer support)
    - Graphical traffic rate monitoring

        - Network intrusion detection system
        - Dynamic DNS client
    - Network infrastructure
        - DHCP server
        - NTP server
        - DNS server
            - Dynamic updates via DHCP
        - RADIUS server
    - VPN support
        - Dynamic routes autoconfiguration
    - HTTP proxy
        - Internet cache
        - User authentication
        - Content filtering (with categorized lists)
        - Transparent antivirus
        - Delay pools
    - Intrusion Detection System
    - Mail Server
        - Virtual domains
        - Quotas
        - SIEVE support
        - External account retrieval
        - POP3 and IMAP with SSL/TLS
        - Spam and antivirus filtering
            - Greylisting, blacklisting, whitelisting
        - Transparent POP3 proxy filter
        - Catch-all account
- **Webmail**
- **Web server**
    - Virtual hosts
- **Certification authority**
- **Workgroup**
    - Centralized users and groups management
        - Master/slave support
        - Windows Active Directory Synchronization
    - Windows PDC
        - Password policies
        - Support for Windows 7 clients
    - Network resource sharing
        - File server
            - Antivirus
            - Recycle bin
        - Print server
    - Groupware: calendar, address book, webmail, etc.
    - VoIP server
        - Voicemail

        - Conference rooms
        - Calls through an external provider
        - Call transfers
        - Call parking
        - Music on hold
        - Queues
        - Logs
- **Jabber/XMPP server**
    - Conference rooms
- **Zentyal User Corner for self users info updating**
- **Reporting and monitoring**
    - Dashboard for centralized service information
    - Monitor CPU, load, disk space, thermal, memory
    - Disk usage and RAID status
    - Summarized and full system reports
    - Event notification via mail, RSS or Jabber
- **Software updates**
- **Backups (configuration and remote data backup)**

Development

Zentyal uses an open source model, with all the source code available for its users.

Design

Zentyal is a web application using Apache webserver with mod perl as foundation and Mason components as building blocks, mainly written in object oriented Perl, with some Javascript for visual improvements.

Its design incorporates modern programming techniques as:

- Design patterns: an Observer design pattern is used mainly to integrate different modules across Zentyal. E.g. each service reports about which ports it needs to be open. Besides this, a Singleton holds global configuration and behavior details.
- Presentation and logic decoupling: user interface uses CSS and Ajax, and include several Mason components, as a generic table used to configure services. Program logic is held inside library packages and CGI-like code.
- Fault tolerance: errors and warnings are managed through software exceptions, flowing from core to its handling routine.

It also offers debugging facilities, integrating the layout of the execution stack of the Perl 5 interpreter.

Service are monitored and automatically respawned if they die.

Community

Main Zentyal community work and support takes place at Zentyal Forum.

Zentyal's (eBox Platform) inclusion on Ubuntu Gutsy Gibbon was preceded by some comments at Ubuntu Forums community.

There is also a very active group of Linkstation users which succeeded in porting Zentyal (eBox Platform) to PowerPC.

Documentation

- Installation guide: holds instructions about different installation methods like CD and Ubuntu packages, and how to get source code and run Zentyal from scratch.
- Official documentation (available both in English and Spanish): aimed at Zentyal users, introduces concepts and terminology, and explains different services and use cases.
- Instructions for developers: Tutorials and tips aimed at software developers, including module development guide, how to create Zentyal Debian packages or a development environment.
- Other documentation: Full list of available documentation including How-Tos, FAQ, Screencasts and other documents.
- API reference: class and method's description and parameters