Version 1.00

ZABBIX Reference Manual

[for ZABBIX 1.0beta9]

[…unfinished…]

Revision: 1.00-0005

ZABBIX

1 Table of Contents

2 Revision History

Version	Date	Reason	Who
1.0	17/01/03	Initial release	Alexei Vladishev

3 Distribution List

Author	Changes
Alexei Vladishev	Author and maintainer of the Manual
Charlie Collins	Many significant improvements to initial (LyX) version of the document.
Shawn Marriott	Proofreading of the Manual.

4 Introduction

Purpose of this document

The purpose of this document is to provide a comprehensive introduction and overview of ZABBIX, its architecture, the features it offers and their functions. This document contains all information necessary for the successful administration of ZABBIX.

What you should already know

No technical knowledge is required, although an understanding of UNIX is essential.

Who should use this document?

Anyone involved in installation and administration of ZABBIX, and anyone else wishing to get an insight into how it works.

Structure of the document

Introduction

Part1 Overview of the ZABBIX system

Part2 Installation guide

Part3 Configuration guide

Part4 ZABBIX Internals

Appendix

5 Document conventions

The ZABBIX Manual uses the typographical conventions shown in the following table.

Convention	Description
ZABBIX	ZABBIX
italic text	Name of file or directory, emphasis
bold text	Programs, applications
NAME	Names
a@b.c	Email addresses, URL
	Shell commands
script	
Note	Notes and comments
Important	Important notes
Tip	Tips, hints

6 Overview of ZABBIX

What is ZABBIX?

ZABBIX was created by Alexei Vladishev, (alex@gobbo.caves.lv).

ZABBIX is software that monitors numerous parameters of a network and the health and integrity of servers.
ZABBIX uses a flexible notification mechanism that allows users to configure e-mail based alerts for virtually any event. This allows a fast reaction to server problems.
ZABBIX offers excellent reporting and data visualisation features based on the stored data. This makes ZABBIX ideal for capacity planning.

ZABBIX supports both polling and trapping. All ZABBIX reports and statistics, as well as configuration parameters are accessed through a web-based front end. A web-based front end ensures that the status of your network and the health of your servers can be assessed from any location. Properly configured, ZABBIX can play an important role in monitoring IT infrastructure. This is equally true for small organisations with a few servers and for large companies with a multitude of servers.

ZABBIX is free of cost. ZABBIX is written and distributed under the GPL General Public License. This means that it's source code is freely distributed and available for the general public.
Both free and commercial support is available and provided by Alexei Vladishev.

What does ZABBIX offer?

ZABBIX offers:

- support for both polling and trapping mechanisms
- server software for Linux, Solaris, HP-UX, AIX, Free BSD, Open BSD, OS X
- native high performance agents (client software for Linux ,Solaris, HP-UX, AIX, Free BSD, Open BSD, OS X, Windows NT4.0, Windows 2000, Windows XP)
- monitoring of servers without any agent installed
- secure user authentication
- flexible user permissions
- web-based interface
- flexible e-mail notification of predefined events

- high-level (business) view of monitored resources

Why use ZABBIX?

- Open Source solution
- highly efficient agents for UNIX and WIN32 based platforms
- low learning curve
- high ROI. Downtimes are very expensive.
- low cost of ownership
- very simple configuration
- centralised monitoring system. All information (configuration, performance data) is stored in relational database
- high-level service tree
- very easy setup
- support for SNMP (v1,v2). Both trapping and polling.
- visualisation capabilities
- built-in housekeeping procedure

Users of ZABBIX

Many organisations of different size around the World rely on ZABBIX as primary monitoring platform.

7 About this Manual

This manual is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. This manual is part of ZABBIX software. The latest version of the manual is available at http://zabbix.sourceforge.net.

The ZABBIX Reference Manual is not distributed under a GPL-style license. Use of the manual is subject to the following terms:

- Conversion to other formats is allowed, but the actual content may not be altered or edited in any way.
- You may create a printed copy for your own personal use.
- For all other uses, such as selling printed copies or using (parts of) the manual in another publication, prior written agreement from Alexei Vladishev is required.
- Please send an e-mail to alex@gobbo.caves.lv for more information.

8 Main Goals and Principles of ZABBIX Development

There are several goals ZABBIX is trying to achieve:

- become recognised Open Source monitoring tool
- create ZABBIX user group, which helps making the software even better
- provide high-quality commercial support

Main principles of ZABBIX development:

- be user friendly
- keep things simple
- use as few resources as possible
- be as fast as possible
- document every aspect of the software

9 Use of ZABBIX

9.1 Performance monitoring

One of most important uses of ZABBIX is performance monitoring. Processor load, number of running processes, number of processes, disk activity, status of swap space, and memory availability are some of the numerous system parameters ZABBIX is able to monitor. ZABBIX provides a system administrator with timely information about performance of a server. In addition, ZABBIX can produce trend graphs to help identify bottlenecks in system performance.

9.2 Alerting users

Having performance monitoring is good, but it is almost useless without a powerful notification mechanism. With ZABBIX, an administrator can define virtually any possible condition for a trigger, using flexible expressions. Any time these expressions become true (or false), an alert will be emailed to any address defined by the administrator.
External programs can be used for user-defined notification methods such as SMS, phone notifications, etc.
ZABBIX can predict future behaviour of monitored parameters using Least Square Algorithm. This allows user to be notified even before system state achieves critical level.

Note: Prediction of system behaviour will be completed in ZABBIX 1.0.

9.3 Monitoring of log files

ZABBIX can be used for centralised monitoring of log files.

Note: This functionality will be completed in ZABBIX 1.0

9.4 Integrity Checking

ZABBIX is capable of server integrity monitoring. All critical configuration files, binaries, kernel, scripts, and web server HTML pages can be monitored by ZABBIX so that the administrator can be alerted to modifications made to these files.

9.5 Logging services

All values of monitored parameters are stored in a database. The collected data can be used later for any purposes.

9.6 Capacity planning

Viewing trends of process load, disk usage, database activity, or other important metrics allows a system administrator to clearly see when the next hardware upgrade should be made.

9.7 Assuring and monitoring of SLA

ZABBIX is able to monitor Service Level Agreements (SLA). It also keeps SLA-related historical data that helps to identify and improve weak areas of an IT infrastructure.

9.8 High level view of IT resources and services

A High level service tree allows the creation of dependencies between various IT resources. Such representation enables the following questions to be answered:

What IT services depends on availability of resource X?

Example: If processor load is too high on server A, then these IT services will be affected: Oracle server, WEB banking, Online transaction processing, etc.

What resources specific IT service depends on?

Example: WEB portal may depend on the following resources:

1. processor load on server A
2. connection to ISP provider
3. disk space on volume /data on server A
4. availability of Oracle DB engine on server B
5. speed of execution of user requests
6. availability of Apache server on server C
7. etc etc

Such a dependency tree helps identify weak points in IT infrastructure.

Example: If several critical services offered by IT department depends on, for example, availability of disk space on some server, then it's time to think about distribution of the volume across different servers or disk arrays to eliminate possible risks.

9.9 Other

- availability analysis
- graphical representation of collected information
- Network maps
- custom screens

10 Installation Guide

10.1 How to Get ZABBIX

Check the ZABBIX Home Page at http://zabbix.sourceforge.net for information about the current version and for downloading instructions.

10.2 Hardware requirements

Memory requirements

ZABBIX requires both physical and disk memory. 32 Mb of physical memory and 20 Mb of free disk memory could be a good starting point. However, the amount of required disk memory obviously depends on the number of hosts and parameters that are being monitored. If you're planning to keep a long history of monitored parameters, you should be thinking of at least a couple of gigabytes to have enough space to store the history in the database.

Each ZABBIX daemon process requires several connections to a database server. Amount of memory allocated for the connection depends on configuration of the database engine.

Note:	**zabbix_trapperd** can be configured so that connections to the database will only be made during process of new value. See description of parameter *DBConnectOnEach*.

Remember, the more physical memory you have, the faster the database (and therefore ZABBIX) works!

10.2.1 Supported Platforms

Due to security requirements and mission-critical nature of monitoring server, UNIX is the only operating system that can consistently deliver the necessary performance, fault tolerance and resilience. ZABBIX operates on market leading versions.

ZABBIX is tested on the following platforms:

- Linux 2.xx
- Solaris 2.xx
- HP-UX 11.xx
- AIX 4.3
- Free BSD 4.3
- SCO Open Server 5.0.5

Note:	Note that ZABBIX may work on other Unix-like operating systems.

10.2.2 Software Requirements

ZABBIX is built around modern Apache WEB server, MySQL or PostgreSQL database engine, and the PHP scripting language.

The following software is required to run ZABBIX:

- Apache

  Version 1.3.12 or later required. Apache can be found at http://www.apache.org

- MySQL (or PostgreSQL)

  Version 3.22 or later required. MySQL can be found at http://www.mysql.com

- PostgreSQL (or MySQL)

  Version 7.0.2 or later required. PostgreSQL can be found at http://www.postgresql.org

- MySQL or PostgreSQL development files (headers and libraries).

  Note: Usually provided as part of mysql-dev or postgresql-dev packages.

- PHP

  Version 4.0 or later required. PHP and its modules can be found at http://www.php.net. PHP must be compiled as Apache module.

- PHP GD module

  The module is required for displaying graphs and maps. The module must support images in PNG format.

- PHP 4.0 MySQL or PostgreSQL module

- GNU Make

  Required for building ZABBIX or its agents from sources. In case if pre-built binaries are used, GNU make is not necessary.

- WEB browser on client side

  Support for HTML and PNG images required. MS Explorer (5.xx and 6.xx) and Mozilla 1.x work perfectly. Cookies and JavaScript must be enabled. Other browsers may work with ZABBIX as well.

- NET-SNMP (or UCD-SNMP) library and header files
  Required for support of SNMP agents. *Optional.*

- Open SSL library and header files. Required for SNMP support. *Optional.*

10.2.3 Required software considerations

ZABBIX may work on previous versions of **Apache**, **MySQL**, **PostgreSQL** as well.
If using **PostgreSQL**, version 7.0.2 or later is strongly recommended.

PostgreSQL vs MySQL

Regarding the choice between **PostgreSQL** and **MySQL**, **MySQL** is recommended for several reasons:

- recent benchmarks using ZABBIX clearly show that **PostgreSQL** (7.1.x) is at least 10 times slower than **MySQL** (3.23.29)

Note: These results are predictable. ZABBIX server processes use simple SQL statements like single row INSERT, UPDATE, and simple SELECT operators. In such environment, use of advanced SQL engine (like **PostgreSQL**) is overkill.

- no need to constantly run resource-hungry command "*vacuum*" for **MySQL**
- **MySQL** is used as a primary development platform.

MySQL MyISAM table type is not a best choice because of lack of row-level locking. So, InnoDB is recommended. InnoDB uses row level locking. Therefore, most of SQL statements can be executed simultaneously without delay. As a result, performance of InnoDB is better than MyISAM. For more information about **MySQL** see http://www.mysql.com.

If you do use **PostgreSQL**, **zabbix_suckerd** will periodically (defined in *HousekeepingFrequency*) execute command "*vacuum analyze*".

For more information about **PostgreSQL** see www.postgresql.org.

10.2.4 Time synchronisation

It is very important to have precise system date on server with ZABBIX running. **timed** is one of most popular daemons that synchronises the host's time with the time of other machines.

10.3 Structure of ZABBIX distribution

ZABBIX distribution:

- *doc*

The directory contains this Manual in different formats

- *src*

The directory contains sources for all ZABBIX processes except frontends.

- *src/zabbix_sucker*

The directory contains Makefile and sources for **zabbix_suckerd.**

- *src/zabbix_agent*

The directory contains Makefile and sources for **zabbix_agent** and **zabbix_agentd.**

- *src/zabbix_trapper*

The directory contains Makefile and sources for **zabbix_trapperd.**

- *src/zabbix_sender*

The directory contains Makefile and sources for **zabbix_sender.**

- *include*

The directory contains include ZABBIX files.

- *misc*

- *misc/init.d*

The directory contains start-up scripts for different platforms.

- *misc/pinger*

The directory contains scripts for ICMP pinging, **pinger.pl**.

- *frontends*

- *frontends/php*

The directory contains sources for PHP frontend.

- *create*

The directory contains SQL script for initial database creation.

- *create/mysql*

MySQL database schema.

- *create/postgresql*

PostgreSQL database schema.

- *create/data*

Data for initial database creation.

- *upgrades*

The directory contains upgrade procedures for different versions of ZABBIX.

10.4 Installation procedure

Basic installation of ZABBIX usually takes no more than 15 minutes.

10.4.1 Server side

Create the ZABBIX superuser account

This is the user the server will run as. For production use you should create a dedicated unprivileged account ("zabbix" is commonly used). Running ZABBIX as "*root*," "*bin*," or any other account with special rights is a security risk. Do not do it!

Note: ZABBIX daemon processes (**zabbix_suckerd** and **zabbix_trapperd**) are protected from being run under *root* account.

Untar ZABBIX sources

Use command:

```
gunzip zabbix.tar.gz; tar -xvf zabbix.tar
```

Create the ZABBIX database.

ZABBIX comes with SQL scripts used to create the required database schema and also to insert a default configuration. There are separate scripts for **MySQL** and **PostgreSQL**.

For **MySQL**:

```
mysql -u<username> -p<password>
>create database zabbix;
>quit;
cd create/mysql
cat schema.sql |mysql -u<username> -p<password> zabbix
cd ../data
cat data.sql |mysql -u<username> -p<password> zabbix
```

For **PostgreSQL**:

```
psql -U <username>
create database zabbix;
> \q
```

```
cd create/postgresql
cat schema.sql|psql -U <username> zabbix
cd ../data
cat data.sql|psql -U <username> zabbix
```

Configure and compile the source code for your system

The sources must be compiled for both the server (monitoring machine) as well as the clients (monitored machines).

To configure the source for the server, you must specify which database will be used.

```
./configure—with-mysql –with-net-snmp # for MySQL
```

or

```
./configure—with-pgsql –with-net-snmp # for PostgreSQL make
```

Note: Use flag *—enable-static* to statically link libraries. If you plan to distribute compiled binaries among different servers, you must use this flag to make these binaries work without required libraries.
Flag *–with-ucd-snmp* can be used instead of *–with-net-snmp*. If no SNMP support required, both *–with-net-snmp and –with-ucd-snmp* may be skipped.

However, if you want to compile client binaries only, run:

```
./configure
```

Parameter *—enable-static* may be used to force static linkage.

Make everything:

```
make
```

Copy created binaries from *bin/* to */opt/zabbix/bin* or any other directory
Other common directories are */usr/local/bin* and */usr/local/zabbix/bin*.

Configure /etc/services

The step is not real requirement. However, it is recommended. On the client (monitored) machines, add the following lines to */etc/services*:

zabbix_agent 10000/tcp
zabbix_trap 10001/tcp

Configure /etc/inetd.conf

If you are going to use a non-daemon versions of ZABBIX trapper, add the following line to */etc/inetd.conf*:

zabbix_trap stream tcp nowait.3600 zabbix /opt/zabbix/bin/zabbix_trapper

If you plan to use **zabbix_agent** instead of the recommended **zabbix_agentd**, the following line must be added:

zabbix_agent stream tcp nowait.3600 zabbix /opt/zabbix/bin/zabbix_agent

Restart inetd

```
killall -HUP inetd
```

Configure WEB interface

As user *zabbix* (or whatever you decided to name your dedicated ZABBIX account), do the following:

Change these values in *frontends/php/include/db.inc*:

```
$DB_TYPE ="POSTGRESQL"; /* Or "MYSQL" for MySQL */
$DB_SERVER ="localhost";
$DB_DATABASE ="zabbix";
$DB_USER ="";
$DB_PWD =""
```

Copy the **PHP** source files to a place where your web server can get to it. Perhaps */home/zabbix/html* or */home/zabbix/public_html* or */var/www/html/zabbix*, etc.

For example:

```
mkdir /home/zabbix/html

cp -R frontends/php/* /home/zabbix/html/
```

Configure /etc/zabbix/zabbix_agent.conf

You need to configure this file for every host having **zabbix_agent** installed. The file should contain IP address of ZABBIX server. Connections from other hosts will be denied. You may take *misc/conf/zabbix_agent.conf* as example.

Configure /etc/zabbix/zabbix_agentd.conf

You need to configure this file for every host with **zabbix_agentd** installed. The file should contain the IP address of the ZABBIX server. Connectionsfrom other hosts will be denied. You may take *misc/conf/zabbix_agentd.conf* as example.

Configure /etc/zabbix/zabbix_suckerd.conf

For small installations (up to ten monitored hosts), default parameters are sufficient. However, you should change default parameters to maximize performance from ZABBIX. See section [Performance tuning] for more details.

You may take *misc/conf/zabbix_suckerd.conf* as example.

Configure /etc/zabbix/zabbix_trapperd.conf

For small installations (up to ten monitored hosts), default parameters are sufficient. However, you should change default parameters to maximize performance from ZABBIX. See section [Performance tuning] for more details.

You may take *misc/conf/zabbix_trapperd.conf* as example.

Run server processes

Run **zabbix_suckerd** and **zabbix_trapperd** on server side.

```
cd bin

./zabbix_suckerd

./zabbix_trapperd
```

Run agents

Run **zabbix_agentd** where necessary.

```
cd bin

./zabbix_agentd
```

10.4.2 Client side

Create the ZABBIX account.

This is the user the agent will run as. For production use you should create a dedicated unprivileged account ("zabbix" is commonly used). ZABBIX agents have protection against running under *root* account.

Untar ZABBIX sources.

Use command:

```
gunzip zabbix.tar.gz; tar xvf zabbix.tar
```

Configure and compile the source code for your system.

The sources must be compiled for the client only.

To configure the source for the client:

```
./configure
```

Note: Use flag —*enable-static* to statically link libraries. If you plan to distribute compiled binaries among different servers, you must use this flag to make these binaries work without required libraries.

Build agent:

```
make
```

Copy created binaries from *bin/* to */opt/zabbix/bin* or any other directory
Other common directories are */usr/local/bin* or */usr/local/zabbix/bin*.

Configure /etc/services

The step is not real requirement. However, it is recommended.

On the client (monitored) machines, add the following lines to */etc/services*:

zabbix_agent 10000/tcp
zabbix_trap 10001/tcp

Configure /etc/inetd.conf

If you plan to use **zabbix_agent** instead of the recommended **zabbix_agentd**, the following line must be added:

zabbix_agent stream tcp nowait.3600 zabbix /opt/zabbix/bin/zabbix_agent

Restart inetd

```
killall -HUP inetd
```

Configure /etc/zabbix/zabbix_agent.conf

You need to configure this file for every host having **zabbix_agent** installed. The file should contain IP address of ZABBIX server. Connections from other hosts will be denied. Note, that no end of line character should present in the file.
You may take *misc/conf/zabbix_agent.conf* as example.

Configure /etc/zabbix/zabbix_agentd.conf

You need to configure this file for every host with zabbix_agentd installed. The file should contain IP address of ZABBIX server. Connections from other hosts will be denied. You may take *misc/conf/zabbix_agentd.conf* as example.

Run zabbix_agentd on all monitored machines

```
/opt/zabbix/bin/zabbix_agentd
```

Not that you should not run **zabbix_agentd** if you have chosen to use **zabbix_agent**!

11 Configuration Guide

11.1 Initial Configuration

Once your ZABBIX installation is complete, you will need to configure ZABBIX. Point your web browser to the location you installed the **PHP** scripts to. (For example, if you copied the PHP files to */home/zabbix/public_html* or */home/zabbix/html*, point your browser to http://localhost/~zabbix.

Once there, click on the CONFIG link. You will be prompted for a username and password. ZABBIX comes pre-configured with an administrative account: Enter the username 'Admin' and leave the password field empty.
Add the configuration of your SMTP server of choice. Next, click on the "USERS" link. Add yourself to list of ZABBIX users. Don't forget to add "read/write"
and "add" default permissions to the user. After you create your account, click the "Media" link beside your account listing, and add your email address.
Next, delete the default Admin user or specify password. Click on the 'HOSTS' link to add new hosts to be monitored. For each host you add, be sure that you've installed the agent software on the host. If you have not yet done this,
set the status to "Not monitored". When a host is added, a list of all possible parameters for the host will be automatically added.
After you've added your desired hosts, click on the "ITEMS" link to modify the details of the monitored parameters. You may disable or delete monitored parameters or change how often the parameters are checked.
Next, click on the 'TRIGGERS' link to change the triggers relating to monitored parameters. You can change threshold values, disable or delete triggers, and set up actions (email notification).
When you set up an email notification, you may modify the subject and message body as you wish. You can use macros in both email subject and email body. For example, the subject for a warning message could look like:

Processor load on www.zabbix.org is *{www.zabbix.org:system[procload].last(0)}*

In this case, you will receive message with subject like "Processor load on www.zabbix.org is 0.85"

Click on the "MAPS" link to set up a network map. A map will help you quickly identify any problems with your monitored machines.
Click on the "GRAPHS" link to set up a graph. A graph provides quick historical statistics on one or more monitored statistics.
Finally, do not forget to set up IT Services to get high level view of servers and applications.
Once you have an initial configuration created, run zabbix_suckerd on the server machine:

```
/opt/zabbix/bin/zabbix_suckerd
```

and run **zabbix_agentd** on all monitored machines

```
/opt/zabbix/bin/zabbix_agentd
```

You should not run **zabbix_agentd** if you have chosen to use **zabbix_agent**!

11.2 Graphs

User-defined graphs allow the creation of complex graphs. These graphs can be easily accessed via the menu item “Graphs”.

11.3 Media

Media is a delivery channel for ZABBIX alerts. None, one or more media types can be assigned to user.

1. EMAIL – Email notification
2. SCRIPT – Custom script. ZABBIX passes three parameters to the script: Recipient, Subject and Message.

11.4 Actions

Action may be defined for the following events:
1. Trigger changes status from FALSE to TRUE
2. Trigger changes status from TRUE to FALSE

Note: Status change FALSE->UNKNOWN->TRUE is treated as FALSE->TRUE, and TRUE->UNKNOWN->FALSE as TRUE->FALSE.

Each action has a scope and severity.

Scope	Severity	Description
This trigger	- (ignored)	Action is performed for this trigger only
All triggers of this host	X	Global action. Action performed for all triggers of all hosts related to this trigger if trigger severity is equal or more than severity of this action.
All triggers	X	Global action. Action performed for all triggers if trigger severity is equal or more than severity of this action.

11.5 Triggers

Trigger is defined as a logical expression.
See section 14.3 for details about syntax of trigger expressions.

Expression is recalculated every time ZABBIX server receives new value, if this value is part of this expression. The expression may have the following values:

Value	Description
TRUE	If trigger is in TRUE state, something wrong is happened.
FALSE	This is normal state of trigger.

Value	Description
UNKNOWN	In this case, ZABBIX cannot evaluate expression. This may happen because of several reasons: [to be finished]

11.6 Screens

ZABBIX screens allow grouping of various information for quick access and display on one screen. Easy-to-use screen builder makes creation of the screens easy and intuitive.

The following elements can be used for screen building:

- simple graphs
- user-defined graphs

Number of elements in each screen is unlimited.

11.7 IT Services

IT Services are intended for those who want to get a high-level (business) view of monitored infrastructures. In many cases, we are not interested in low-level details, like lack of disk space, high processor load, etc. What we are interested is availability of service provided by our IT department. We can also be interested in identifying weak places of IT infrastructure, SLA of various IT services, structure of existing IT infrastructure, and many other information of higher level.

ZABBIX IT Services provides answers to all mentioned questions.

IT Services is hierarchy representation of monitored data.

A very simple IT Service structure may look like:

```
IT Service
|
|-Workstations
| |
| |-Workstation1
| |
| |-Workstation2
|
|-Servers
```

Each node of the structure has attribute status. The status is calculated and propagated to upper levels according to selected algorithm. Triggers create lowest level of the IT Services.
[To be finished...]

11.8 User permissions

All ZABBIX users access the ZABBIX application through the Web-based front end. Each ZABBIX user is assigned a unique user identity and a password. All user passwords are encrypted and stored on the ZABBIX database. Users can not use their user id and password to log directly into the UNIX server unless they have also been set up accordingly to UNIX. Communication between the Web Server and the user's browser can be protected using SSL.

Access permissions on screen within the menu may be set for each user. By default, no permissions are granted on a screen when user is registered to the ZABBIX.

Note that the user is automatically disconnected after 30 minutes of inactivity.

[To be finished...]

11.9 Development Environment

Debian 3.0 and RedHat 8.0 Linux (all on Intel hardware) are used as primary development platform for ZABBIX.

Two servers are used for test purposes:

1. Debain Linux 2.1, Intel PII/350Mhz, 192MB, IDE
2. RedHat 8.0, Intel P4/1.6Mhz, 256MB, IDE

It means that if you have difficulties choosing between Linux and other OS, go for Debian (http://www.debian.org) or RedHat (http://www.redhat.com) - you will get better support.

ZABBIX 1.0beta4 was developed and tested on FreeBSD 4.4. Thus, FreeBSD (http://www.freebsd.org) is also safe choice.

[Add information about versions of test HP-UX, AIX, Solaris boxes]

11.10 Platform specific notes

ZABBIX operates on all supported UNIX platforms. However use of some platforms introduces issues that need to be considered.

11.10.1 FreeBSD

Default FreeBSD settings (at least, FreeBSD 4.4) cannot handle the amount of TCP connections ZABBIX is able to generate. To solve this issue, define *NoTimeWait* parameter in both *zabbix_suckerd.conf* and *zabbix_trapperd.conf*. This will greatly decrease number of sockets in TIME_WAIT state. As an alternative solution, kernel parameters can be tuned using *sysctl* mechanism.

12 Upgrading

The upgrade procedure is quite simple. New binaries and frontend should be installed according to latest installation instructions.
In order to update database structure, the following steps should be performed.

Important: The upgrade process can take from 0 seconds (if no patches required) to several hours. Note that before applying database patches, all ZABBIX processes must be stopped. For production installations a database backup is required!

12.1 Database upgrade

Go to the *upgrades/dbpatches* directory. In this directory are subdirectories named according to a version upgrade (e.g. *1.0beta3_to_1.0beta4*). Enter the directory corresponding to your upgrade (if you are upgrading through multiple versions, you will need to apply the upgrades one at a time). Depending on which database you use:

```
cd mysql; cat patch.sql |mysql zabbix -u<username> -p<password>
```

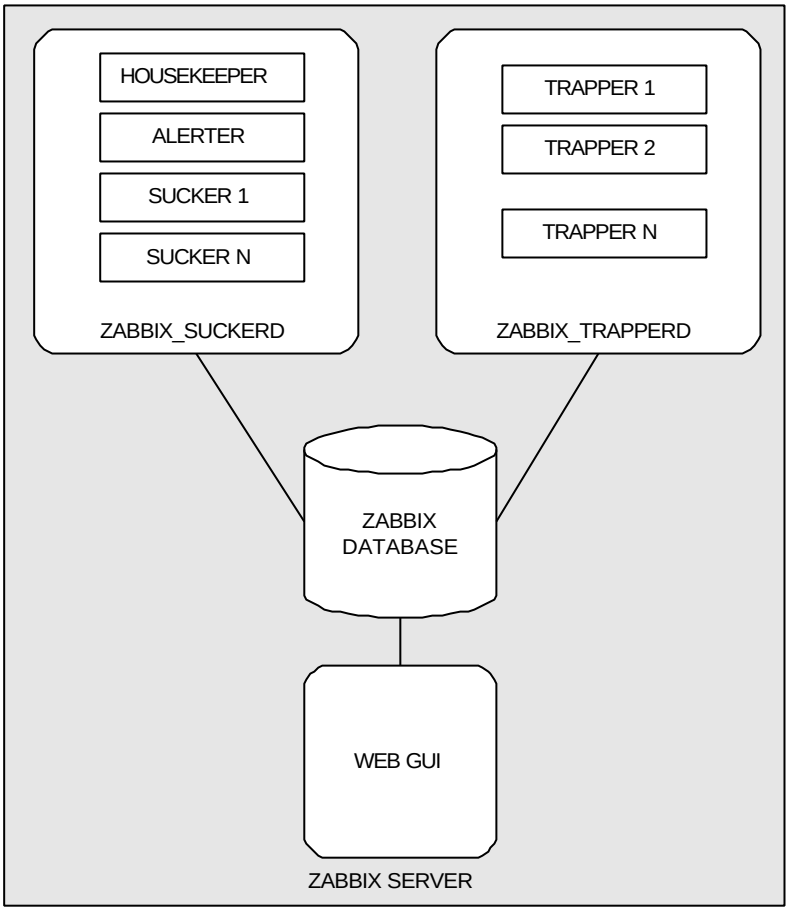
or

```
cd postgresql; cat patch.sql|psql -U <username> zabbix
```

Do not forget to upgrade PHP front-end files.

Finally, read version specific notes below for any extra procedures and useful information.

13 ZABBIX Architecture



ZABBIX consist of four major components:

1. ZABBIX database
2. **zabbix_suckerd.** This is main process of ZABBIX application. The process performs the following functions:
   - collects information from ZABBIX native agents, SNMP agents and does simple checks
   - periodically executes housekeeping procedure
   - sends alerts (both generated by **zabbix_suckerd** and **zanbbix_trapperd**)
3. **zabbix_trapperd**. This binary processes information sent by **zabbix_sender.**
4. WEB-based graphical user interface.

13.1 Overview

ZABBIX is structured in a client-server architecture. ZABBIX is capable of running in either standalone or **inetd** execution mode.

Note: The standalone method is the recommended approach.

The ZABBIX server processes can be run in a polling or listening configuration or both. These methods differ in the approach that they utilise and affect what processes must be run. In the Server Polling Method the ZABBIX Server polls established ZABBIX clients (preferred.) In the Server Listen Method the ZABBIX server awaits connections from active ZABBIX clients.

ZABBIX server processes consist of a standalone process used for the Server Polling Mode (**zabbix_suckerd**) and a choice of standalone or **inetd** processes for the Server Listen Method (**zabbix_trapperd** or **zabbix_trapper**).

ZABBIX agents (clients) consist of a choice of standalone or **inetd** processes for the Server Polling Method and a standalone process for the Server Listen Method (There is no **inetd** process for the Server Listen Method because when using that method the client must initiate the connection.)

If the server is used in the Server Polling Mode then the ZABBIX listening client is used (**zabbix_agentd** or **zabbix_agent**), this client awaits connections from the server. If the server is used in the Server Listen Mode then the ZABBIX active client is used (**zabbix_sender**), this client initiates connections to the server.

ZABBIX is written in **C**. WEB frontend is written in **PHP**.

13.2 Server Approach Modes

ZABBIX supports two methods of server role activity. The server may be used in a polling mode (Server Polling Method) or the server may be using in a listening mode (Server Listen Method) or both.

These methods are not mutually exclusive and can be used in conjunction.

The standard method utilised is the Server Polling Method in most instances. In this method the server is setup and configured to be aware of clients and then the server polls the clients at defined intervals.

The additional Server Active Method can also be used. This is generally used in a complimentary fashion to have clients send information to the server on a predetermined basis as defined by the administrator. For example, the Server Listen Method can be used to have a client machine send information to the server when a backup process has begun and again when it has ended. In this manner the client dictates the interval of the information and sends information when specified events occur rather than the server requesting information at specified intervals.

13.2.5 Server Polling Method (Server connects to Client)

In this method the ZABBIX server, **zabbix_suckerd**, initiates connections to a client agent placed on the monitored host (**zabbix_agentd** or **zabbix_agent**) on a periodic basis. The server requests specific information (processor load, free memory, available inodes, etc, whatever is being monitored.)
The client provides the server with the requested information and the server, in turn, stores the received values in the database.

Note: In this case the traditional sense of client is turned around. The client is actually the server. That is to say that the connection is made from the main ZABBIX server (**zabbix_suckerd**) to the ZABBIX client (**zabbix_agentd**), and therefore the ZABBIX client is acting like a traditional server and listening for connections and responding; and the ZABBIX server is actually acting as a traditional client and initiating a connection.

13.2.2 Server Listen Method (Client connects to Server)

In this method the ZABBIX client (**zabbix_sender**) initiates connections to a ZABBIX server (**zabbix_trapperd** or **zabbix_trapper**) on an event driven basis.

13.3 Theory of operations

[to be finished]

13.4 Data exchange protocol

zabbix_suckerd zabbix_agent(d)

Server ->[command]->Agent
Agent->[result]->Server

[command] has the following structure:

[key|host:port]

[|host:port] are used if we want to use ZABBIX agent as a proxy. Skipper otherwise.

If received result is equal to 'ZBX_NOTSUPPORTED' then the agent does not support processing of required parameter. 'ZBX_ERROR' indicates that agent was unable to return requested parameter.

zabbix_sender zabbix_trapper(d)

Sender ->[server:key:value]->Trapper
Trapper->[OK]->Sender

13.5 Database

Database use plays a very important role in the ZABBIX application. The entire history of received parameter values is stored in the database. In addition, the database is the sole source of configuration parameters for the application. ZABBIX performance depends highly on the efficiency and speed of the database used by ZABBIX.

13.6 ZABBIX Processes

ZABBIX supports several different processes that perform specific functions depending on the mode of execution that is utilised (**inetd** vs. standalone.)

13.6.1 Standalone Mode Processes

zabbix_suckerd

The **zabbix_suckerd** process resides on the server and periodically connects to agents (either ZABBIX native or SNMP) to get the values of parameters being monitored. After receiving the values, the process recalculates the status of ZABBIX triggers. Also, the process sends alerts to users if required.

If **zabbix_suckerd** is unable to get a value from an agent because of network problems, the value will be checked only after DELAY_ON_NETWORK_ERROR (defined in *include/common.h*, default value is 60 seconds). After that the host status will be changed to UNREACHABLE.

If an agent does not support requested parameter, **zabbix_suckerd** will change status of the parameter to NOTSUPPORTED and will not try to get its value anymore.

By default, **zabbix_suckerd** forks five copies of itself. This means that ZABBIX will be able to get four (five minus process used for housekeeping) monitored parameters simultaneously. To modify the number of forked processes, change the parameter *StartSuckers* in */etc/zabbix/zabbix_suckerd.conf*. Note that each forked process requires one connection to the database. Make sure that the database is able to provide this number of connections.

One **zabbix_suckerd** process is used for housekeeping purposes only. The housekeeping process periodically (hourly, by default) deletes outdated information from the tables history, alarms, alerts and sessions.

The housekeeping process will delete all records from the alarms and alerts table older than the value defined in ZABBIX configuration.
Parameter *HousekeepingFrequency* from *zabbix_suckerd.conf* defines how often housekeeping procedure must be executed.

No command line switches are supported by **zabbix_suckerd**.

The **zabbix_suckerd** process is run as a daemon under a non-privileged user account, usually *zabbix*.

Syslog or a file is used to store debug information for the process.

zabbix_agentd

The **zabbix_agentd** process resides on the host being monitored. Its purpose is to provide request information to **zabbix_suckerd**.

When executed, the **zabbix_agentd** process forks itself (five copies by default). Load is balanced between each copy of the process.

When a connection to **zabbix_agentd** is made, the agent determines if the connection is coming from an authorised server. The *zabbix_agentd.conf* file must exist and contain the IP address (or list of IP addresses) of ZABBIX server. Connections from other IP addresses are rejected.

No command-line switches supported by **zabbix_agentd**.

The process is run as a daemon under a non-priveledged user account, usually *zabbix*.

Syslog or a file is used to store debug information for the process.

List of supported parameters can be extended by using configuration file parameter *UserParameter*.

ZABBIXW32.exe (WIN32 agent)

About

ZabbixW32 is ZABBIX agent for Win32 systems. It will work on Windows NT 4.0, Windows 2000, and Windows XP. ZabbixW32 doesn't work on other Windows platforms.

Installation

Installation is very simple and includes 3 steps:

1. Unpack ZabbixW32.exe
2. Create configuration file *c:/zabbix_agentd.conf* (it has the same syntax as UNIX agent).
3. Run this command to install ZABBIX as a service.

```
ZabbixW32.exe install
```

If you wish to use configuration file other that *c:\zabbix_agentd.conf*, you should use the following command for service installation:

```
ZabbixW32.exe --config <your_configuration_file> install
```

Full path to configuration file should be specified.

Now you can use Control Panel to start agent's service or run

```
ZabbixW32.exe start
```

Note: Windows NT 4.0 note. ZabbixW32 uses PDH (Performance Data Helper) API to gather various system information, so PDH.DLL is needed. This DLL is not supplied with Windows NT 4.0, so you need to download and install it by yourself. Microsoft Knowledge Base article number 284996 describes this in detail and contains a download link. You can find this article at

http://support.microsoft.com/default.aspx?scid=kb;en-us;284996

Command line syntax

Usage:

```
zabbixw32 [options] [command]
```

Where possible commands are:

```
check-config    : Check configuration file and exit
standalone      : Run in standalone mode
start           : Start Zabbix Win32 Agent service
stop            : Stop Zabbix Win32 Agent service
install         : Install Zabbix Win32 Agent as service
remove          : Remove previously installed Zabbix Win32 Agent service
install-events  : Install Zabbix Win32 Agent as event source for Event Log
                  This is done automatically when service is being installed
remove-events   : Remove Zabbix Win32 Agent event source
                  This is done automatically when service is being removed
help            : Display help information
version         : Display version information
```

And possible options are:

```
--config <file> : Specify alternate configuration file
                  (default is C:\zabbix_agentd.conf)
```

The file contains configuration parameters for **zabbixx32**. Supported parameters:

Parameter	Mandatory	Default value	Description
Alias	No	-	Sets the alias for parameter. It can be useful to substitute long and complex parameter name with a smaller and simpler one. For example, if you wish to retrieve paging file usage in percents from the server, you may use parameter "perf_counter[\Paging File(_Total)\% Usage]", or you may define an alias by adding the following line to configuration file: Alias = pg_usage:perf_counter[\Paging File(_Total)\% Usage] After that you can use parameter

Parameter	Mandatory	Default value	Description
			name "pg_usage" to retrieve the same information. You can specify as many "Alias" records as you wish. Please note that aliases cannot be used for parameters defined in "PerfCounter" configuration file records.
DebugLevel	No	-	The parameter has no effect.
ListenPort	No	10000	Port number to listen
LogFile	No	-	Name of log file. If not set, **syslog** is used.
LogUnresolved Symbols	No	-	Controls logging of unresolved symbols during agent startup. Values can be strings "yes" or "no" (without quotes).
MaxCollectorPr ocessingTime	No	100	Sets maximum acceptable processing time of one data sample by collector thread (in milliseconds). If processing time will exceed specified value, warning message will be written to the log file.
NoTimeWait	No	-	The parameter has no effect.
PerfCounter	No	-	<parameter_name>,"<perf_counter_path>",<period> Defines new parameter <parameter_name> which is an average value for system performance counter <perf_counter_path> for the specified time period <period> (in seconds). For example, if you wish to receive average number of processor interrupts per second for last minute, you can define new parameter "interrupts" as following: PerfCounter = interrupts,"\Processor(0)\Interrupts /sec",60 Please note double quotes around performance counter path. Samples for calculating average value will be taken every second.
PidFile	No	-	The parameter has no effect.
Server	Yes	-	Comma-delimited list of IP addresses of ZABBIX servers.

Parameter	Mandatory	Default value	Description
			Connections from other IP addresses will be rejected.
StartAgents	No	-	The parameter has no effect.
UserParameter	No	-	User-defined parameter to monitor. There can be several user-defined parameters. Value has form <key>,<shell command> Example: *UserParameter=test,echo 1*

zabbix_trapperd

The process collects information passed by active agents (**zabbix_sender**).

By default, **zabbix_trapperd** forks ten copies of itself. The parameter TRAPPERD_FORKS in *include/common.h* can be changed to modify the number of forked processes.

No command-line switches are supported by **zabbix_trapperd**.

The process is run as daemon under non-privileged user account, usually *zabbix*.

Syslog or a file is used to store debug information for the process.

13.6.2 Inetd Mode Processes

zabbix_agent

The **zabbix_agent** process resides on the host being monitored. Its purpose it to provide requested information to **zabbix_suckerd**. The **zabbix_agent** process is designed to be used with **inetd** daemon.

Execution time of the process is limited. If the agent is not able to get the requested information within *Timeout* seconds defined in */etc/zabbix/zabbix_agent.conf*, it kills itself.

When a connection to **zabbix_agent** is made, the agent reads */etc/zabbix/zabbix_agent.conf* in order to determine if the connection is coming from an authorised server. The *zabbix_agent.conf* file must exist and contain the IP address of ZABBIX server. Connections from other IP addresses are rejected.

Use of **zabbix_agentd** (see below) instead of **zabbix_agent** is strongly recommended. The **zabbix_agentd** process does not require extra *fork()* and

exec() calls for every connection, no frequent parsing of configuration file is required as well.

zabbix_trapper

The process provides trapping support for ZABBIX. It constantly waits for connections from **zabbix_sender** agents. It is designed to be used with **inetd** daemon.

If possible, it is recommended that **zabbix_trapperd** is used to provide trapping support. The **zabbix_trapperd** process does not require extra *fork()* and *exec()* calls for every connection.

13.6.3 Command line processes

zabbix_sender

The process, when executed, sends information to ZABBIX trapper (**inetd** or standalone).

zabbix_sender <zabbix_server> <port> <host:key> <value>

zabbix_server Name or IP address of ZABBIX server to connect.

port Port number to connect to ZABBIX. (Port number of **zabbix_trapper** process).

host:key Host name and key for value.

value Value for parameter “host:key”. Positive float number or a character string.

EXAMPLE:

zabbix_sender oracle.company.com 10001 zabbix.company.com:procload 2.34

In this example, **zabbix_sender** will send information about processor load on host *oracle.company.com* to ZABBIX server located at *zabbix.company.com*. Value of the processor load is 2.34.

If no parameters are given, **zabbix_sender** expects list of values from standard input i.e. stdin. Parameters should be given in the following format:

zabbix_server1 port1 host1:key1 value1

zabbix_server2 port2 host2:key2 value2

....

zabbix_serverN portN hostN:keyN valueN

In this case, **zabbix_sender** will be executed only once, therefore no unnecessary forks will be required.

13.7 Extending ZABBIX agents

Using configuration parameter *UserParameter* can easily extend ZABBIX agents. For example, number of emails waiting for delivery may be defined as:

UserParameter: mailq,echo /var/spool/mail/ | wc -w*

See *misc/conf/zabbix_agentd.conf* for already predefined (but commented) parameters.
The used command must return one line. This line is sent back to the ZABBIX server unhanged.

13.8 Monitoring of MS Windows servers

MS Windows based servers can be monitored by using:

- native ZABBIX agent for WIN32 platforms (Windows NT4.0, Windows 2000, Windows XP)
- SNMP agent
- **zabbix_sender** compiled under CygWin (http://www.cygwin.com)

14 ZABBIX Configuration

14.1 Configuration Methods and Execution Modes

Depending on the configuration method and execution mode that is being utilised (polling vs. listening and **inetd** vs. standalone respectively) the ZABBIX server and ZABBIX client have several different processes and names.

14.1.1 Polling mode server

Server must be standalone mode, client can be either standalone or **inetd** mode.

Standalone:

server - **zabbix_suckerd**
client - **zabbix_agentd**

Inetd:

client - **zabbix_agent**

14.1.2 Listen mode server

Server can be standalone or **inetd** mode, client must be standalone command line.

Standalone:

server - **zabbix_trapperd**
client - **zabbix_sender** (command line)

Inetd:

server - **zabbix_trapper**
client - **zabbix_sender** (command line)

14.2 Configuration files

All ZABBIX processes can be configured by changing appropriate configuration files. After a change was made, a restart of the process id required.

All ZABBIX configuration files are stored in the directory */etc/zabbix/*.
Configuration files must have read permissions.

/etc/zabbix/zabbix_suckerd.conf

The file contains configuration parameters for **zabbix_suckerd**. The file must exist and it should have read permissions for user *zabbix*. Supported parameters:

Parameter	Mandatory	Default value	Description
AlertScriptsPath	No	/home/zabbix/bin	Location of scripts for user-defined media types.
DBHost	No	localhost	DB host name.
DBName	Yes	-	Database name. Usually "*zabbix*".
DBSocket	No	-	DB socket name. Used for non-TCP connection to **MySQL** database. Example: */tmp/mysql.sock*
DBPassword	No	NULL	User's password. If password is not used, then this parameter must be commented.
DBUser	No	NULL	User name for connecting to the database
DebugLevel	No	3	Debug level (0 - none, 1 - critical, 2 - error, 3 - warnings, 4 - debug)
DisableHousekeeping	No	0	If set to 1, housekeeping procedure will be disabled
HousekeepingFrequency	No	1	The parameter defines how often the daemon must perform housekeeping procedure (in hours). If **PostgreSQL** is used set the value to 24 as it will perform command *VACUUM*.
FpingLocation	No	*/usr/sbin/fping*	Location of ICMP pinger.
LogFile	No	-	Name of log file. If not set, **syslog** is used.
NoTimeWait	No	-	Experimental parameter. If set, no sockets in TIME_WAIT state will exist. Works on Free BSD.
PidFile	No	*/tmp/zabbix_suckerd.pid*	Name of file to store PID.
PingerFrequency	No	30	**zabbix_suckerd** pings servers once per PingerFrequency seconds.
SenderFrequency	No	30	The parameter defines how often the daemon must try to send alerts (in seconds)
StartSuckers	No	5	Number of **zabbix_suckerd** to start (4-255)

Parameter	Mandatory	Default value	Description
Timeout	No	5	Do not spend more than *Timeout* seconds on retrieving requested value (1-255)

Note: Example of the configuration file can be found at *misc/conf/zabbix_suckerd.conf*

/etc/zabbix/zabbix_agentd.conf

The file contains configuration parameters for **zabbix_agentd**. The file must exist and it should have read permissions for user *zabbix*. Supported parameters:

Parameter	Mandatory	Default value	Description
DBHost	No	localhost	DB host name.
DBUser	No	NULL	User name for connecting to the database
DebugLevel	No	3	Debug level (0 - none, 1 - critical, 2 - error, 3 - warnings, 4 - debug)
Enable Proxy	No	0	0 – request forwarding is prohibited 1 – agent will be able to forward requests
ListenIP	No	-	IP address to bind agent to. Useful if the host has multiple interfaces.
ListenPort	No	10000	Port number to listen
LogFile	No	-	Name of log file. If not set, **syslog** is used.
NoTimeWait	No	-	Experimental parameter. If set, no sockets in TIME_WAIT state will exist. Works on FreeBSD.
PidFile	No	*/tmp/zabbix_suckerd.pid*	Name of file to store PID.
Server	Yes	-	Comma-delimited list of IP addresses of ZABBIX servers. Connections from other IP addresses will be rejected.
StartAgents	No	5	Number of agents to start
UserParameter	No	-	User-defined parameter to monitor. There can be several user-defined parameters. Value has form <key>,<shell command>

Parameter	Mandatory	Default value	Description
			Example: *UserParameter=users,who |wc -l*

Note: Example of the configuration file can be found at *misc/conf/zabbix_agentd.conf*

/etc/zabbix/zabbix_agent.conf

The file contains configuration parameters for **zabbix_agent**. The file must exist and it should have read permissions for user *zabbix*. Supported parameters:

Parameter	Mandatory	Default value	Description
Enable Proxy	No	0	0 – request forwarding is prohibited 1 – agent will be able to forward requests
Server	Yes	-	Comma-delimited list of IP addresses of ZABBIX servers. Connections from other IP addresses will be rejected.
Timeout	No	3	Do not spend more that Timeout seconds on getting requested value (1-255).
UserParameter	No	-	User-defined parameter to monitor. There can be several user-defined parameters. Value has form <key>,<shell command> Example: *UserParameter=users,who |wc -l*

Note: Example of the configuration file can be found at *misc/conf/zabbix_agent.conf*

/etc/zabbix/zabbix_trapper.conf

The file contains configuration parameters for **zabbix_trapper**. The file must exist and it should have read permissions for user *zabbix*.
Supported parameters:

Parameter	Mandatory	Default value	Description
DBHost	No	localhost	DB host name.

Parameter	Mandatory	Default value	Description
DBName	Yes	-	Database name. Usually "*zabbix*".
DBPassword	No	NULL	User's password. If password is not used, then this parameter must be commented.
DBSocket	No	-	DB socket name. Used for non-TCP connection to **MySQL** database. Example: */tmp/mysql.sock*
DBUser	No	NULL	User name for connecting to the database
DebugLevel	No	3	Debug level (0 - none, 1 - critical, 2 - error, 3 - warnings, 4 - debug)
LogFile	No	-	Name of log file. If not set, **syslog** is used.
NoTimeWait	No	-	Experimental parameter. If set, no sockets in TIME_WAIT state will exist. Works on FreeBSD.
PidFile	No	*/tmp/zabbix_suckerd.pid*	Name of file to store PID.
Timeout	No	5	Do not spend more than *Timeout* seconds on retrieving requested value (1-255)

Note: Example of the configuration file can be found at *misc/conf/zabbix_trapper.conf*

/etc/zabbix/zabbix_trapperd.conf

The file contains configuration parameters for **zabbix_trapperd**. The file must exist and it should have read permissions for user *zabbix*.
Supported parameters:

Parameter	Mandatory	Default value	Description
DBConnectOnEach	No	0	If 1, **zabbix_trapperd** will connect to the database on each value received.
DBHost	No	localhost	DB host name.
DBName	Yes	-	Database name. Usually "*zabbix*".
DBPassword	No	NULL	User's password. If password is not used, then this parameter must be commented.

Parameter	Mandatory	Default value	Description
DBSocket	No	-	DB socket name. Used for non-TCP connection to **MySQL** database. Example: */tmp/mysql.sock*
DBUser	No	NULL	User name for connecting to the database
DebugLevel	No	3	Debug level (0 - none, 1 - critical, 2 - error, 3 - warnings, 4 - debug)
ListenPort	No	10001	
LogFile	No	-	Name of log file. If not set, **syslog** is used.
NoTimeWait	No	-	Experimental parameter. If set, no sockets in TIME_WAIT state will exist. Works on FreeBSD.
PidFile	No	*/tmp/zabbix_suckerd.pid*	Name of file to store PID.
StartTrappers	No	5	
Timeout	No	5	Do not spend more than *Timeout* seconds on retrieving requested value (1-255)

Note: Example of the configuration file can be found at *misc/conf/zabbix_trapperd.conf*

14.3 Expression for triggers

The expressions used in triggers are very flexible. You can use them to create complex logical tests regarding monitored statistics.

The following operators are supported for triggers:

Operator	Definition
-	Arithmetical minus
+	Arithmetical plus
/	Division
*	Multiplication
>	More than
<	Less than
=	Is equal. The operator is defined as A=B <=> (A>B-0.000001) and (A<B+0.000001)

|	Logical OR
&	Logical AND

The following functions are supported:

Function	Definition
abschange	Returns absolute difference between last and previous value.
avg	Average value for period of time. Parameter defines length of the period in seconds.
delta	Same as max(<period>)-min(<period>)
change	Returns difference between last and previous value.
count	Number of successfully retrieved values for period of time. Parameter defines length of the period in seconds.
diff	Returns: 1 – if last and previous values differs 0 – otherwise
last	Last (most recent) value. Parameter is ignored.
max	Maximal value for period of time. Parameter defines length of the period in seconds.
min	Minimal value for period of time. Parameter defines length of the period in seconds.
nodata	Returns: 1 – if no data received during period of time. Parameter defines length of the period in seconds. The length should not be less than 30 seconds. Can be used for item type TRAPPER only!
prev	Returns previous value. Parameter is ignored.
str	Find string in last (most recent) value. Parameter defines string to find. Returns: 1 – found 0 – otherwise Can be used for items having type STRING only.
sum	Sum of values for period of time. Parameter defines length of the period in seconds.

Note: Note that all above functions (except **diff** and **str**) cannot be used for non-numeric parameters!

A simple useful expression might look like:

{<server>:<key>.<function>(<parameter>)}<operator><const>

Parameter must be given even for those functions, which ignore it. Example: *last(0)*

Example 1. Processor load is too high on zabbix.sf.net.

{zabbix.sf.net:system[procload].last(0)}>5

'*zabbix.sf.net:system[procload]*' gives a short name of the monitored parameter. It specifies that the server is 'sourceforge.net' and the key being monitored is '*system[procload]*'. By using the function *last()*', we are referring to the most recent value. Finally, '*>5*' means that the trigger is true whenever the most recent processor load measurement from *zabbix.sf.net* is greater than 5.

Example 2. zabbix.sf.net is overloaded.

({zabbix.sf.net:system[procload].last(0)}>5)|({zabbix.sf.net:system[procload].min(600)}>2)

The expression is true when either the current processor load is more than 5 or the processor load was more than 2 during last 10 minutes.

Example 3. /etc/passwd has been changed.

Use of function *diff*:

({zabbix.sf.net:cksum[/etc/passwd].diff(0)})>0

The expression is true when the previous value of checksum of */etc/passwd* differs from the most recent one.

Similar expressions could be useful to monitor changes in important files, such as */etc/passwd*, */etc/inetd.conf*, */kernel*, etc.

Example 4. Someone downloads big file from the Internet.

Use of function *min*:

({zabbix.sf.net:netloadin1[eth0].min(300)})>100000

The expression is true when number of received bytes on *eth0* is more than 100000 within last 5 minutes.

Example 5. Both clustered SMTP servers are down.

Note use of two *different* hosts in one expression:

({smtp1.zabbix.org:check_service[smtp].last(0)}=0)&({smtp2.zabbix.org:check_service[smtp].last(0)}=0)

The expression is true when both SMTP server is down on both *smtp1.zabbix.org* and *smtp2.zabbix.org*.

Example 6. Zabbix agent needs to be upgraded.

Use of function *str*:

{zabbix.zabbix.org:version[zabbix_agent].str(beta8)}=0

The expression is true if ZABBIX agent has version *beta8* (presumably *1.0beta8*).

Example 7. Server is unreachable.

{zabbix.zabbix.org:status.last(0)}=2

Note:	Note 'status' is special parameter which is calculated if and only if corresponding host has at least one parameter for monitoring. See description of 'status' for more details.

Example 8. No heart beats within last 3 minutes.

Use of function *nodata*:

{zabbix.zabbix.org:tick.nodata(180)}=1

'tick' must have type 'ZABBIX trapper''. In order to make this trigger work, item 'tick' must be defined. The host should periodically send data for this parameter using **zabbix_sender**. If no data is received within 180 seconds, the trigger value becomes TRUE.

14.4 Trigger dependencies

[to be finished]

14.5 Monitored Parameters

14.5.1 Host statuses

Monitored

This is normal host status. ZABBIX monitors all hosts having this status. Values received by **zabbix_trapperd** will be accepted and stored in the database.

Not monitored

ZABBIX will not monitor hosts having this status. Values received by **zabbix_trapperd** will be ignored.

Unreachable

A host becomes unreachable if and only if a network error (host unreachable, connection refused, unknown host, etc) occurred during retrieval of monitored parameter. Once a host became unreachable ZABBIX will not try to request any values within next 60 seconds. Then, if a parameter is retrieved successfully, ZABBIX will change host status to 'Monitored'. Otherwise, the host will not be checked within next 60 seconds. And so on.

14.5.2 Types of parameters

ZABBIX agent
Simple check
SNMPv1 agent
SNMPv2 agent
Zabbix trapper
Internal

14.5.3 Flexible and non-flexible parameters

Flexible parameter is parameter which accepts argument. For example, *diskfree[*]* is flexible parameter. '*' is any string that will be passed as argument of the parameter. *diskfree[/]*, *diskfree[/opt]* - correct definitions.

String between [] may contain the following characters:

0-9a-zA-Z.:,()_/[space]

14.5.4 List of supported parameters

SIMPLE CHECKS

Simple checks does not require any agent running on a host being monitored. ZABBIX server will make direct connections to get status of the service. ZABBIX will use host name or IP address defined in the host details.

Key	Description	Return value	Comments
icmpping	Checks if server accessible by ICMP ping	0 – ICMP ping fails 1 – ICMP ping successful	One of **zabbix_suckerd** processes performs ICMP pings once per *PingerFrequency* seconds.
ftp	Checks if FTP server is running and accepting connections	0 – FTP server is down 1 – FTP server is running	
http	Checks if HTTP (WEB) server is running and accepting connections	0 – HTTP server is down 1 – HTTP server is running	
imap	Checks if IMAP server is running and accepting connections	0 – IMAP server is down 1 – IMAP server is running	
nntp	Checks if NNTP server is running and accepting connections	0 – NNTP server is down 1 – NNTP server is running	
pop	Checks if POP server is running and accepting connections	0 – POP server is down 1 – POP server is running	
smtp	Checks if SMTP server is running and accepting connections	0 – SMTP server is down 1 – SMTP server is running	
ssh	Checks if SSH server is running and accepting connections	0 – SSH server is down 1 – SSH server is running	
ftp_perf	Checks if FTP server is running and accepting connections	0 – FTP server is down Otherwise, number of milliseconds spent connecting to FTP server	
http_perf	Checks if HTTP (WEB) server is running and accepting connections	0 – HTTP server is down Otherwise, number of milliseconds spent connecting to HTTP server	
imap_perf	Checks if IMAP server is running and accepting connections	0 – IMAP server is down Otherwise, number of milliseconds spent connecting to IMAP server	
nntp_perf	Checks if NNTP server is running and accepting connections	0 – NNTP server is down Otherwise, number of milliseconds spent connecting to NNTP server	
pop_perf	Checks if POP server is running and accepting connections	0 – POP server is down Otherwise, number of milliseconds spent connecting to POP server	
smtp_perf	Checks if SMTP	0 – SMTP server is down	

Key	Description	Return value	Comments
	server is running and accepting connections	Otherwise, number of milliseconds spent connecting to SMTP server	
ssh_perf	Checks if SSH server is running and accepting connections	0 – SSH server is down Otherwise, number of milliseconds spent connecting to SSH server	

INTERNAL CHECKS

Internal checks allow to control internals of ZABBIX. Internal checks are calculated by **zabbix_suckerd.**

Key	Description	Return value	Comments
zabbix[history]	Number of values stored in tbale HISTORY	Integer value	Do not use if PostgreSQL or MySQL InnoDB is used!
zabbix[items]	Number of items in ZABBIX database	Integer value	
zabbix[items_unsupported]	Number of unsupported items in ZABBIX database	Integer value	
zabbix[queue]	Number of items in the queue	Integer value	
zabbix[triggers]	Number of triggers in ZABBIX database	Integer value	

ZABBIX AGENT

Key	Description	Return value	Comments
check_port[ip,port]	Check, if it is possible to make TCP connection to port number <port>.	0 – can connect 1 – cannot connect	IP address is optional. If IP address is missing, 127.0.0.1 is used. Example: check_port[80] can be used to test availability of WEB server
check_service[ftp,ip,port]	Check if FTP server is running and accepting connections.	0 – FTP server is down 1 – FTP server is running	IP address and port number is optional. If IP address is missing 127.0.0.1 is used. If port number is missing, '21' is used. IP address cannot be specified without port number. Example: check_service[ftp,45] can be used to test availability of FTP server on TCP port 45.
check_service[http,ip,port]	Check if HTTP (WEB) server is running and accepting connections.	0 – HTTP server is down 1 – HTTP server is running	IP address and port number is optional. If IP address is missing 127.0.0.1 is used. If

Key	Description	Return value	Comments
			port number is missing, '80' is used. IP address cannot be specified without port number. Example: check_service[http] can be used to test availability of HTTP server on TCP port 80.
check_service[imap,ip,port]	Check if IMAP server is running and accepting connections.	0 – IMAP server is down 1 – IMAP server is running	IP address and port number is optional. If IP address is missing 127.0.0.1 is used. If port number is missing, '143' is used. IP address cannot be specified without port number.
check_service[nntp,ip,port]	Check if NNTP server is running and accepting connections.	0 – NNTP server is down 1 – NNTP server is running	IP address and port number is optional. If IP address is missing 127.0.0.1 is used. If port number is missing, '119' is used. IP address cannot be specified without port number.
check_service[pop,ip,port]	Check if POP3 server is running and accepting connections.	0 – POP3 server is down 1 – POP3 server is running	IP address and port number is optional. If IP address is missing 127.0.0.1 is used. If port number is missing, '110' is used. IP address cannot be specified without port number.
check_service[smtp,ip,port]	Check if SMTP (email) server is running and accepting connections.	0 – SMTP server is down 1 – SMTP server is running	IP address and port number is optional. If IP address is missing 127.0.0.1 is used. If port number is missing, '25' is used. IP address cannot be specified without port number.
check_service[ssh,ip,port]	Check if SSH server is running and accepting connections.	0 – SSH server is down 1 – SSH server is running	IP address and port number is optional. If IP address is missing 127.0.0.1 is used. If port number is missing, '22' is used. IP address cannot be specified without port number.
check_service_perf[ftp,ip,port]	Check performance of FTP server.	0 – FTP server is down <sec> – number of seconds spent on connection to the FTP server	IP address and port number is optional. If IP address is missing 127.0.0.1 is used. If port number is missing, '21' is used. IP address cannot be specified without port number. Example: check_service[ftp] can be used to test

Key	Description	Return value	Comments
			performance of FTP server. Return value: 0.0234
check_service_perf[http,ip,port]	Check performance of HTTP (WEB) server.	0 – HTTP server is down <sec> – number of seconds spent on connection to the HTTP server	IP address and port number is optional. If IP address is missing 127.0.0.1 is used. If port number is missing, '80' is used. IP address cannot be specified without port number.
check_service_perf[imap,ip,port]	Check performance of IMAP server.	0 – IMAP server is down <sec> – number of seconds spent on connection to the IMAP server	IP address and port number is optional. If IP address is missing 127.0.0.1 is used. If port number is missing, '143' is used. IP address cannot be specified without port number.
check_service_perf[nntp,ip,port]	Check performance of NNTP server.	0 – NNTP server is down <sec> – number of seconds spent on connection to the NNTP server	IP address and port number is optional. If IP address is missing 127.0.0.1 is used. If port number is missing, '119' is used. IP address cannot be specified without port number.
check_service_perf[pop,ip,port]	Check performance of POP server.	0 – POP server is down <sec> – number of seconds spent on connection to the POP server	IP address and port number is optional. If IP address is missing 127.0.0.1 is used. If port number is missing, '110' is used. IP address cannot be specified without port number.
check_service_perf[smtp,ip,port]	Check performance of SMTP (email) server.	0 – SMTP server is down <sec> – number of seconds spent on connection to the SMTP server	IP address and port number is optional. If IP address is missing 127.0.0.1 is used. If port number is missing, '25' is used. IP address cannot be specified without port number.
check_service_perf[ssh,ip,port]	Check performance of SSH server.	0 – SSH server is down <sec> – number of seconds spent on connection to the SSH server	IP address and port number is optional. If IP address is missing 127.0.0.1 is used. If port number is missing, '22' is used. IP address cannot be specified without port number.
cksum[*]	Calculate check sum of a given file.	Check sum of the file calculate by standard algorithm used by UNIX utility **cksum**	Example: cksum[/etc/passwd]
diskfree[*]	Calculate free (unused) disk space for a given volume.	Unused disk space in Kb	In case of mounted volume, unused disk space for local file system is returned.

Key	Description	Return value	Comments
			Example: diskfree[/tmp]
disktotal[*]	Calculate total disk space for a given volume.	Disk space in Kb	In case of mounted volume, total disk space for local file system is returned. Example: disktotal[/]
diskused[*]	Calculate used disk space for a given volume.	Disk space in Kb	In case of mounted volume, used disk space for local file system is returned. Example: disktotal[/]
disk_read_ops1[*]	Average number of disk reads within last minute for a disk device.	Number of disk reads	Example: disk_read_ops1[hda0]
disk_read_ops5[*]	Average number of disk reads within last 5 minutes for a disk device.	Number of disk reads	Example: disk_read_ops5[hda1]
disk_read_ops15[*]	Average number of disk reads within last 15 minutes for a disk device.	Number of disk reads	Example: disk_read_ops15[sda0]
disk_write_ops1[*]	Average number of disk writes within last minute for a disk device.	Number of disk writes	Example: disk_write_ops1[hda0]
disk_write_ops5[*]	Average number of disk writes within last 5 minutes for a disk device.	Number of disk writes	Example: disk_write_ops5[hda1]
disk_write_ops15[*]	Average number of disk writes within last 15 minutes for a disk device.	Number of disk writes	Example: disk_write_ops15[sda0]
disk_read_blks1[*]	Average number of disk read blocks within last minute for a disk device.	Number of read blocks	Example: disk_read_blks1[hda0]
disk_read_blks5[*]	Average number of disk read blocks within last 5 minutes for a disk device.	Number of read blocks	Example: disk_read_blks5[hda1]
disk_read_blks15[*]	Average number of disk read blocks within last 15 minutes for a disk device.	Number of read blocks	Example: disk_read_blks15[sda0]
disk_write_blks1[*]	Average number of disk written blocks within last minute for a disk device.	Number of written blocks	Example: disk_write_blks1[hda0]
disk_write_blks5[*]	Average number of disk written blocks	Number of written blocks	Example:

Key	Description	Return value	Comments
	within last 5 minutes for a disk device.		disk_write_blks5[hda1]
disk_write_blks15[*]	Average number of disk written blocks within last 15 minutes for a disk device.	Number of written blocks	Example: disk_write_blks15[sda0]
filesize[*]	Size of a given file.	Size in bytes.	File must have read permissions for user 'zabbix'. Example: filesize[/var/log/syslog]
inodefree[*]	Number of unused inodes for a given volume.	Number of unused inodes	Example: inodefree[/]
inodetotal[*]	Total number of inodes for a given volume.	Total number of inodes	Example: inodetotal[/var]
kern[maxfiles]	Maximum number of opened file supported by OS.	Number of files	
kern[maxproc]	Maximum number of processes supported by OS.	Number of processes	
memory[buffers]	Amount of memory used for kernel buffers.	Amount of memory in bytes	
memory[cached]	Amount of cached memory.	Amount of memory in bytes	
memory[free]	Amount of unused physical memory.	Amount of memory in bytes	
memory[shared]	Amount of used physical memory.	Amount of memory in bytes	
memory[total]	Total amount of physical memory.	Amount of memory in bytes	
netloadin1[*]	Average number of bytes received by an interface. Average for last minute.	Number of bytes	Example: netloadin1[eth0]
netloadin5[*]	Average number of bytes received by an interface. Average for last 5 minute.	Number of bytes	Example: netloadin5[ppp0]
netloadin15[*]	Average number of bytes received by an interface. Average for last 15 minute.	Number of bytes	Example: netloadin15[eth0]
netloadout1[*]	Average number of bytes sent by an interface. Average for last minute.	Number of bytes	Example: netloadout1[eth0]
netloadout5[*]	Average number of bytes sent by an interface. Average for last 5 minute.	Number of bytes	Example: netloadout5[ppp0]
netloadout15[*]	Average number of bytes sent by an	Number of bytes	Example:

Key	Description	Return value	Comments
	interface. Average for last 15 minute.		netloadout15[eth0]
ping	Always return 1.	1	Can be used as a TCP ping.
proc_cnt[*]	Number of processes called "*" running.	Number of processes	Example: proc_cnt[inetd]
status	Host status.	0 – normal 2 – unreachable	This parameter is calculated internally by **zabbix_suckerd.** ZABBIX does not connect to an agent in order to calculate value of the 'status'.
swap[free]	Free swap space.	Number of bytes	
system[hostname]	Return host name.	String value	Example of returned value: www.sf.net
swap[total]	Total swap space.	Number of bytes	
system[proccount]	Number of started processes	Number of processes	
system[procload]	Load average for last 1 minute	Load average	Note that this is not percentage.
system[procload5]	Load average for last 5 minute	Load average	Note that this is not percentage.
system[procrunning]	Number of running processes	Number of processes	
system[uname]	Returns detailed host information	String value	Example of returned value: *FreeBSD localhost 4.4-RELEASE FreeBSD 4.4-RELEASE #0: Tue Sep 18 11:57:08 PDT 2001 murray@builder.FreeBSD.org:/usr/src/sys/compile/GENERIC i386*
system[uptime]	System's uptime in seconds	Number of seconds	
system[users]	Number of users connected	Number of users	Command **who** is used on agent side.
tcp_count	Number of established TCP connections	Number of TCP connections	
version[zabbix_agent]	Version of **zabbix_agent(d)** running on monitored host	String value	Example of returned value: 1.0beta10
version[history]	Returns number of records stored in table HISTORY	Number of records	Internal check. No connections to any agent required.
version[items]	Returns number of records stored in table ITEMS	Number of records	Internal check. No connections to any agent required.
version[items_unsupported]	Returns number of unsupported items	Number of unsupported items	Internal check. No connections to any agent required.
version[queue]	Returns length of the queue of ITEMS	Length of the queue	This value is good indicator of the

Key	Description	Return value	Comments
	ready for immediate request of latest information.		performance of ZABBIX server. Ideally this value should be close to zero.

14.5.5 WIN32-SPECIFIC PARAMETERS

This section contains description of parameter supported by ZABBIX WIN32 agent only.

Key	Description	Return value	Comments
agent[avg_collector_time]	Average time spent by collector thread on each sample processing for last minute.	Time in milliseconds	
agent[max_collector_time]	Maximum time spent by collector thread on each sample processing for last minute.	Time in milliseconds	
agent[accepted_requests]	Total number of requests accepted by agent for processing.	Number of requests	
agent[rejected_requests]	Total number of requests rejected by agent for processing.	Number of requests	
agent[timed_out_requests]	Total number of requests timed out in processing.	Number of requests	
agent[accept_errors]	Total number of accept() system call errors.	Number of system calls	
agent[processed_requests]	Total number of requests successfully processed by agent.	Number of requests	
agent[failed_requests]	Total number of requests with errors in processing.	Number of requests	These requests generated ZBX_ERROR return code
agent[unsupported_requests]	Total number of requests for unsupported parameters.	Number of requests	These requests generated ZBX_UNSUPPORTED return code
cpu_util	Average CPU(s) utilisation for last minute.	Processor load in percents	
cpu_util5	Average CPU(s) utilisation for last 5 minute.	Processor load in percents	
cpu_util15	Average CPU(s) utilisation for last 15 minute.	Processor load in percents	
cpu_util[*]	Average CPU(s) utilisation for last minute, where parameter is zero-based CPU number.	Processor load in percents	Example: cpu_util[0]

Key	Description	Return value	Comments
cpu_util5[*]	Average CPU(s) utilisation for last 5 minutes, where parameter is zero-based CPU number.	Processor load in percents	Example: cpu_util5[1]
cpu_util15[*]	Average CPU(s) utilisation for last 15 minutes, where parameter is zero-based CPU number.	Processor load in percents	Example: cpu_util15[0]
md5_hash[*]	MD5 hash of specified file.	String value	Agent will return ZBX_UNSUPPORTED if the file is larger than 64MB. Example: md5_hash[c:\autoexec.bat]
perf_counter[*]	Value of any performance counter, where parameter is the counter path.	Value of the counter	Performance Monitor can be used to obtain list of available counters. Note that this parameter will return correct value only for counters that require just one sample (like "\System\Threads"). It will not work as expected for counters that require more that one sample – like CPU utilisation.
device_state[*]	State of service. Parameter is service name.	0 – running 1 – paused 2 – start pending 3 – pause pending 4 – continue pending 5 – stop pending 6 – stopped 7 – unknown 255 – SCM communication error	Parameter must be real service name as it seen in service properties under "Name:", not service display name!

proc_info[<process>:<attribute>:<type>] Different information about specific process(es).

<process> - process name (same as in *proc_cnt[]* parameter)
<attribute> - requested process attribute. The following attributes are currenty supported:
vmsize - Size of process virtual memory in Kbytes
wkset - Size of process working set (amount of physical memory used by process) in Kbytes
pf - Number of page faults
ktime - Process kernel time in milliseconds
utime - Process user time in milliseconds
io_read_b - Number of bytes read by process during I/O operations
io_read_op - Number of read operation performed by process

io_write_b - Number of bytes written by process during I/O operations
io_write_op - Number of write operation performed by process
io_other_b - Number of bytes transferred by process during operations other than read and write operations
io_other_op - Number of I/O operations performed by process, other than read and write operations
gdiobj - Number of GDI objects used by process
userobj - Number of USER objects used by process
<type> - representation type (meaningful when more than one process with the same name exists). Valid values are:
min - minimal value among all processes named <*process*>
max - maximal value among all processes named <*process*>
avg - average value for all processes named <*process*>
sum - sum of values for all processes named <*process*>

Examples:

1. In order to get the amount of physical memory taken by all Internet Explorer processes, use the following parameter:
   proc_info[iexplore.exe:wkset:sum]
2. In order to get the average number of page faults for Internet Explorer processes, use the following parameter:
   proc_info[iexplore.exe:pf:avg]

Note:	All io_xxx,gdiobj and userobj attributes available only on Windows 2000 and later versions of Windows, not on Windows NT 4.0.

14.5.6 SNMP parameters

Example:

Community: *public*
Oid: *1.2.3.45.6.7.8.0*
Key: <Unique string to be used as reference to triggers>

Note that OID can be given in either numeric or string form. However, in some cases, string OID must be converted to numeric representation. Utility **snmpget** may be used for this purpose:

```
snmpget -On localhost public enterprises.ucdavis.memory.memTotalSwap.0
```

Monitoring of SNMP parameters is possible if either *–with-net-snmp* or *–with-ucd-snmp* flag was specified while configuring ZABBIX sources.

14.6 Supported Parameters by Platform

Parameter/system	Windows (NT4.0, W2000,XP)	Linux (Debian Potato)	FreeBSD 4.3 (i386)	Solaris 5.8 (Ultra-60)	HP-UX 11.00 (9000/800)	AIX 4.3 (Power PC)
agent[accept_errors]	X	-	-	-	-	-
agent[avg_accepted_requests]	X	-	-	-	-	-
agent[avg_rejected_requests]	X	-	-	-	-	-
agent[avg_collector_time]	X	-	-	-	-	-
agent[failed_requests]	X	-	-	-	-	-
agent[max_collector_time]	X	-	-	-	-	-
agent[processed_requests]	X	-	-	-	-	-
agent[timed_out_requests]	X	-	-	-	-	-
agent[unsupported_requests]	X	-	-	-	-	-
check_port[*]	-	X	X	X	X	X
check_service[ftp]	-	X	X	X	X	X
check_service[http]	-	X	X	X	X	X
check_service[imap]	-	X	X	X	X	X
check_service[nntp]	-	X	X	X	X	X
check_service[pop]	-	X	X	X	X	X
check_service[smtp]	-	X	X	X	X	X
check_service[ssh]	-	X	X	X	X	X
check_service_perf[ftp]	-	X	X	X	X	X
check_service_perf[http]	-	X	X	X	X	X
check_service_perf[imap]	-	X	X	X	X	X
check_service_perf[nntp]	-	X	X	X	X	X
check_service_perf[pop]	-	X	X	X	X	X
check_service_perf[smtp]	-	X	X	X	X	X
check_service_perf[ssh]	-	X	X	X	X	X
cksum[*]	X	X	X	X	X	X
cpu_util	X	-	-	-	-	-
cpu_util5	X	-	-	-	-	-
cpu_util15	X	-	-	-	-	-
cpu_util[*]	X	-	-	-	-	-
cpu_util5[*]	X	-	-	-	-	-
cpu_util15[*]	X	-	-	-	-	-
diskfree[*]	X	X	X	X	X	X
disktotal[*]	X	X	X	X	X	X
disk_read_blks1[*]	-	X	-	-	-	-

Parameter/system	Windows (NT4.0, W2000,XP)	Linux (Debian Potato)	FreeBSD 4.3 (i386)	Solaris 5.8 (Ultra-60)	HP-UX 11.00 (9000/800)	AIX 4.3 (Power PC)
disk_read_blks5[*]	-	X	-	-	-	-
disk_read_blks15[*]	-	X	-	-	-	-
disk_read_ops1[*]	-	X	-	-	-	-
disk_read_ops5[*]	-	X	-	-	-	-
disk_read_ops15[*]	-	X	-	-	-	-
disk_write_blks1[*]	-	X	-	-	-	-
disk_write_blks5[*]	-	X	-	-	-	-
disk_write_blks15[*]	-	X	-	-	-	-
disk_write_ops1[*]	-	X	-	-	-	-
disk_write_ops5[*]	-	X	-	-	-	-
disk_write_ops15[*]	-	X	-	-	-	-
filesize[*]	X	X	X	X	X	X
Inodefree[*]	-	X	X	X	X	X
inodetotal[*]	-	X	X	X	X	X
io[disk_io]	-	X	-	-	-	-
io[disk_rblk]	-	X	-	-	-	-
io[disk_rio]	-	X	-	-	-	-
io[disk_wblk]	-	X	-	-	-	-
io[disk_wio]	-	X	-	-	-	-
kern[maxfiles]	-	-	X	-	-	-
kern[maxproc]	-	-	X	-	-	-
md5_hash[*]	X	-	-	-	-	-
memory[buffers]	-	X	-	-	-	-
memory[cached]	X XP only	X	-	-	-	-
memory[free]	X	X	X	X	-	X
memory[shared]	-	X	X	-	-	-
memory[total]	X	X	X	X	X	X
netloadin1[*]	-	X	-	-	-	-
netloadin15[*]	-	X	-	-	-	-
netloadin5[*]	-	X	-	-	-	-
netloadout1[*]	-	X	-	-	-	-
netloadout15[*]	-	X	-	-	-	-
netloadout5[*]	-	X	-	-	-	-
perf_counter[*]	X	-	-	-	-	-
ping	X	X	X	X	X	X
proc_cnt[*]	X	X	-	X	-	-
proc_info[*]	X	-	-	-	-	-

Parameter/system	Windows (NT4.0, W2000,XP)	Linux (Debian Potato)	FreeBSD 4.3 (i386)	Solaris 5.8 (Ultra-60)	HP-UX 11.00 (9000/800)	AIX 4.3 (Power PC)
service_state[*]	X	-	-	-	-	-
swap[free]	X	X	-	X	X	-
swap[total]	X	X	-	X	X	-
swap[in]	-	X	-	-	-	-
swap[out]	-	X	-	-	-	-
system[hostname]	X	X	X	X	X	X
system[interrupts]	-	X	-	-	-	-
system[switches]	-	X	-	-	-	-
system[procload]	X	X	X	X	X	-
system[procload5]	X	X	X	X	X	-
system[procload15]	X	X	X	X	X	-
system[proccount]	X	X	-	-	-	-
system[procrunning]	-	X	-	-	-	-
system[uname]	X	X	X	X	X	X
system[uptime]	X	X	X	-	-	-
system[users]	-	X	X	X	X	X
tcp_count	-	X	X	-	-	X
version[zabbix_agent]	X	X	X	X	X	X

Parameters netload*, disk_read* and disk_write* are supported by **zabbix_agentd** only. **Inetd** version of the agent does not support them.

15 ZABBIX Cookbook

15.1 General recipes

15.1.1 Monitoring of server's availability

Two methods (or combination of both methods) may be used in order to monitor availability of a server.

ICMP ping (pinger.pl)

[to be finished...]

Key 'status'

[to be finished...]

15.1.2 Use of both email and SMS notification for a single user

[to be finished...]

15.2 Monitoring of specific applications

15.2.1 MySQL

Configuration file misc/conf/zabbix_agentd.conf contains
list of parameters that can be used for monitoring of MySQL.

Set of parameter for monitoring MySQL server (v3.23.42 and later)

Change -u<username> and add -p<password> if required

#UserParameter=mysql[ping],mysqladmin -uroot ping|grep alive|wc -l

#UserParameter=mysql[uptime],mysqladmin -uroot status|cut f2 -d":"|cut -f1 -d"T"

#UserParameter=mysql[threads],mysqladmin -uroot status|cut f3 -d":"|cut -f1 -d"Q"

#UserParameter=mysql[questions],mysqladmin -uroot status|cut f4 -d":"|cut -f1 -d"S"

#UserParameter=mysql[slowqueries],mysqladmin -uroot status|cut f5 -d":"|cut -f1 -d"O"

#UserParameter=mysql[qps],mysqladmin -uroot status|cut -f9 d":"

#UserParameter=version[mysql],mysql -V

mysql[ping] Check, if MySQL is alive

Result: 0 - not started

1 - alive

mysql[uptime] Number of seconds MySQL is running

mysql[threads] Number of MySQL threads

mysql[questions] Number of processed queries

mysql[slowqueries] Number of slow queries

mysql[qps] Queries per second

mysql[version] Version of MySQL

Example: mysql Ver 11.16 Distrib 3.23.49, for pc-linux-gnu (i686)

15.2.2 IMAP servers

[to be finished...]

15.3.1 FTP servers

[to be finished...]

15.4.1 POP3 servers

[to be finished...]

15.5.1 NNTP servers

[to be finished...]

15.5.2 WIN32

[to be finished...]

15.5.3 Novell

[to be finished...]

15.5.4 Tuxedo

Tuxedo command line utility **tmadmin** can be used in definition of a *UserParameter* in order to return per server/service/queue performance counters and availability of Tuxedo resources.

15.5.5 Informix

Standard Informix utility **onstat** can be used for monitoring of virtually every aspect of Informix database.

15.5.6 PostgreSQL

[to be finished...]

16 Getting Maximum Performance from ZABBIX

16.1 Real world configuration

Server with ZABBIX 1.0beta9 installed (RedHat Linux 8.0, kernel 2.4.18-14, MySQL/MyISAM 3.23.54a-4, Pentium IV 1.5Ghz, 256Mb, IDE) is able to collect more than 200 parameters per second from servers being monitored (assuming no network delays). How many servers can be monitored by ZABBIX on the hardware, you may ask? It depends on number of monitored parameters and how often ZABBIX should acquire these parameters. Suppose, each server you monitor has ten parameters to watch for. You want to update these parameters once in 30 seconds. Doing simple calculation, we see that ZABBIX is able to handle 600 servers (or 6000 checks). In case if these parameters need to be updated once in a minute, the hardware configuration will be able to handle 600x2=1200 servers. These calculations made in assumption that all monitored values are retrieved as soon as required (latency is 0). If this is not a requirement, then number of monitored servers can be increased even up to 5x-10x times.

16.2 Performance tuning

Performance of ZABBIX can be significantly improved by tuning:

Hardware

- Use fastest processor available
- SCSI is better than IDE (performance of IDE disks may be significantly improved by using utility **hdparm**)
- Use fast Ethernet adapter
- Having more memory is always better

Operating System

- Use latest (stable!) version of OS
- Exclude unnecessary functionality from kernel
- Tune kernel parameters

ZABBIX configuration parameters

Many parameters may be tuned to get optimal performance.

zabbix_suckerd

StartSuckers General rule - keep value of this parameter as low as possible. Every additional instance of **zabbix_suckerd** adds known overhead, in the same time, parallelism is increased. Optimal number of instances is achieved when queue, on average, contains minimum number of parameters (ideally, 0 at any given moment). This value can be monitored by using internal check *zabbix[queue]*.

DebugLevel Optimal value is '3'.

DBSocket **MySQL** only. It is recommended to use *DBSocket* for connection to the database. That is fastest and most secure way.

zabbix_trapperd

DBConnectOnEach Should not be set if you want to get maximum performance.

Database Engine

- use fastest database engine, i.e. **MySQL**
- use stable release of a database engine
- rebuild **MySQL** or **PostgreSQL** from sources to get maximum performance
- follow performance tuning instructions taken from **MySQL** or **PostgreSQL** documentation
- for **MySQL**, use InnoDB table structure

> **Note:** ZABBIX works at least 1.5 times faster (comparing to MyISAM) if InnoDB is used. This is because of increased parallelism. However, InnoDB requires more CPU power.

General advices

- monitor required parameters only
- tune 'Update interval' for all items. Keeping small update interval may be good for nice graphs, however, this may over load ZABBIXtune parameters for default templates
- tune housekeeping parameters
- do not monitor parameters wich return same information. Example: why use *system[procload]*,*system[procload5]* and *system[procload15]* if *system[procload]* contains all.

- avoid use of triggers with long period given as function argument. For example, *max(3600)* will be calculated significantly slower than *max(60)*.

17 ZABBIX Frontends

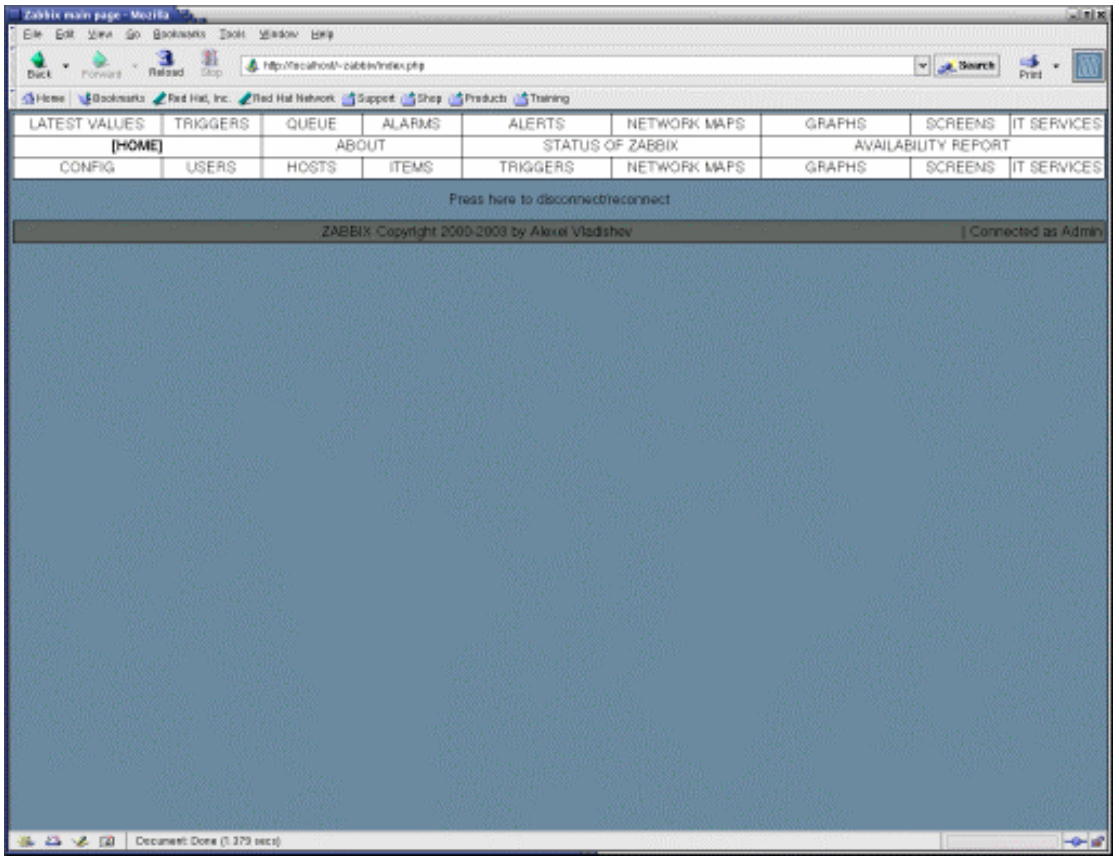
Currently, the only frontend available is the WEB front-end written in **PHP**.

17.1 PHP-based Frontend

The front-end provides a convenient and platform-independent method for accessing ZABBIX. Information provided by the frontend can be either graphical or textual. While graphical representations usually provide the easiest method to understand trends, the text representation of monitored parameter is intended to provide an easy way to export ZABBIX data to other analytical tools.

Menu

Most of the pages contain the menu that consists of three rows. The first and second rows contain end-user functionality. The third row is for ZABBIX administration.



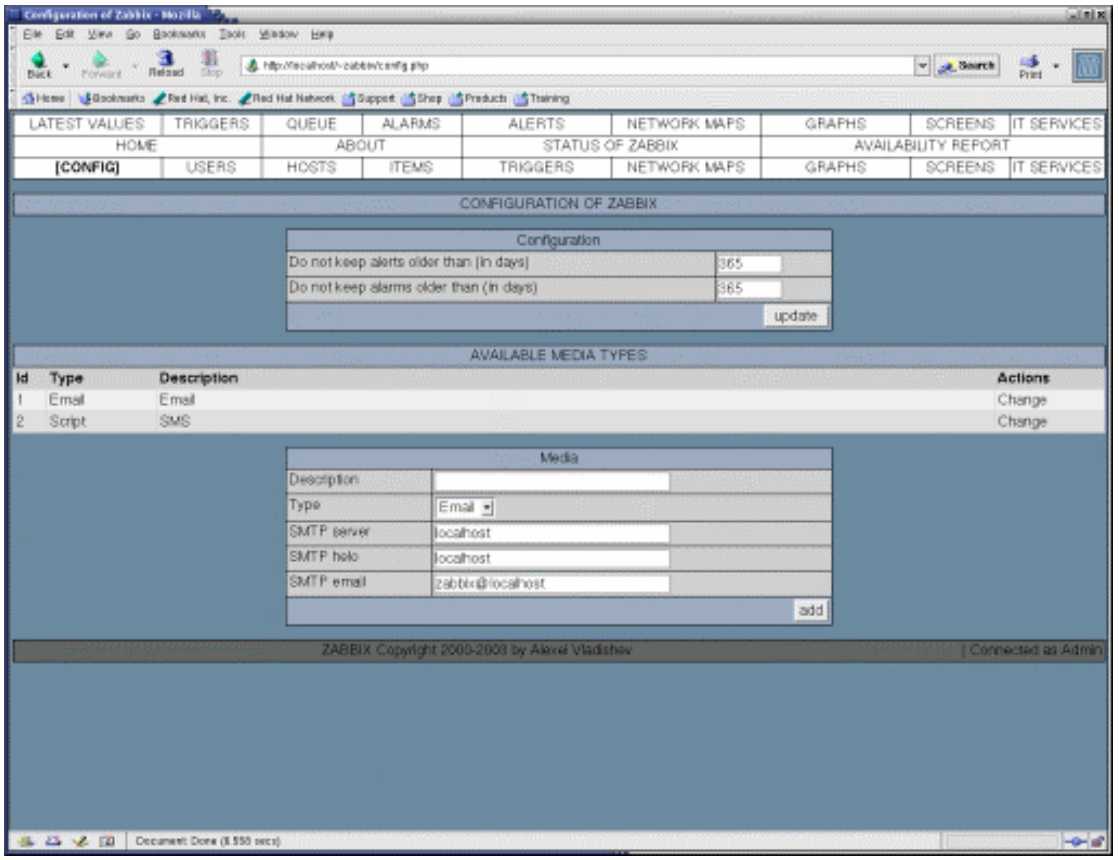
Main menu

Possible actions:

Action	Result
Press a menu item	Transfer to corresponding screen
Press **[Enter]** button	If the name and password is correct, an user will be logged in.
When connected, press 'disconnect'	User will be disconnected.

CONFIG

The menu item is used to set global ZABBIX parameters.



Main menu -> CONFIG (Media type: EMAIL)

Description

Name of the media type. For example: *Email*

SMTP server

IP address or name of SMTP server. ZABBIX sends alert messages via this server.

Example: *mail.zabbix.org*

Value for SMTP HELO authentication

Usually set to domain name. If the field left blank, ZABBIX will not use HELO authentication.

Example: *zabbix.org*

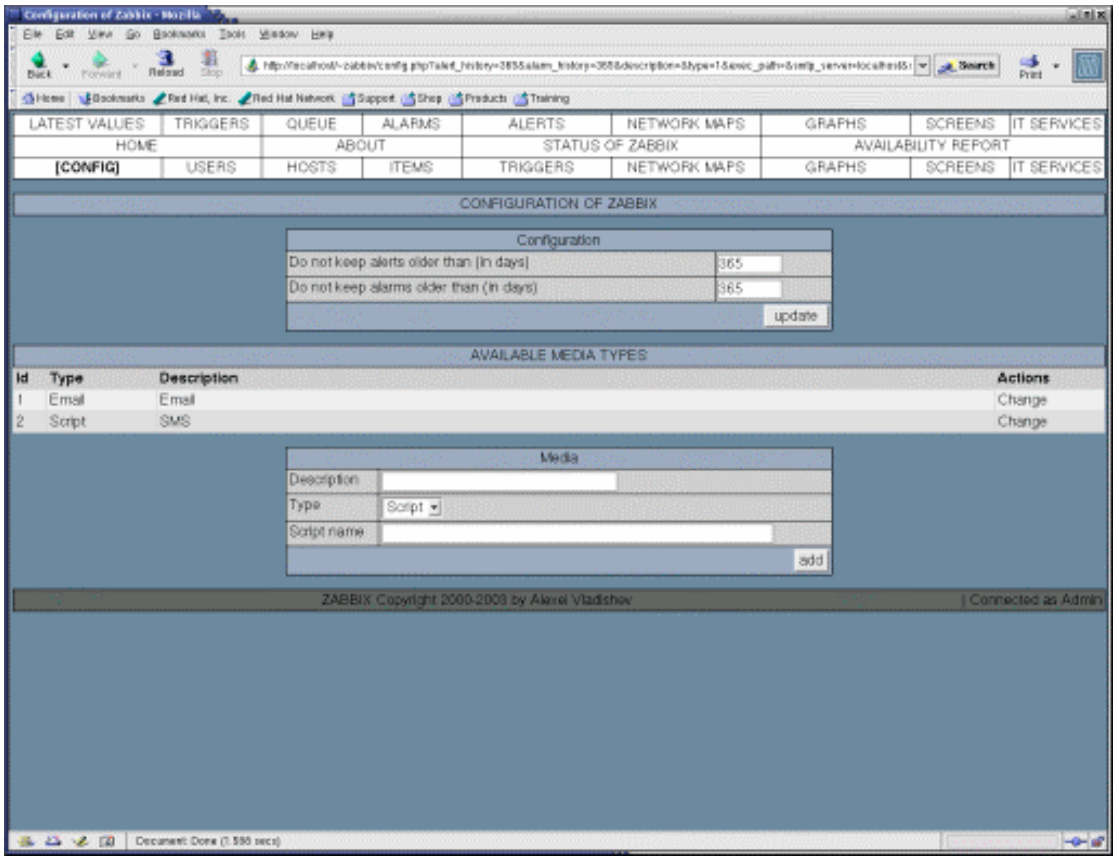
ZABBIX email address to send alerts from

ZABBIX uses this address as source address for all emails.

Example: zabbix@zabbix.org

Possible actions:

Action	Result
Press **[update]** button	Configuration parameters will be updated



Main menu -> CONFIG (Media type: SCRIPT)

Description

Name of the media type. For example: *SMS*

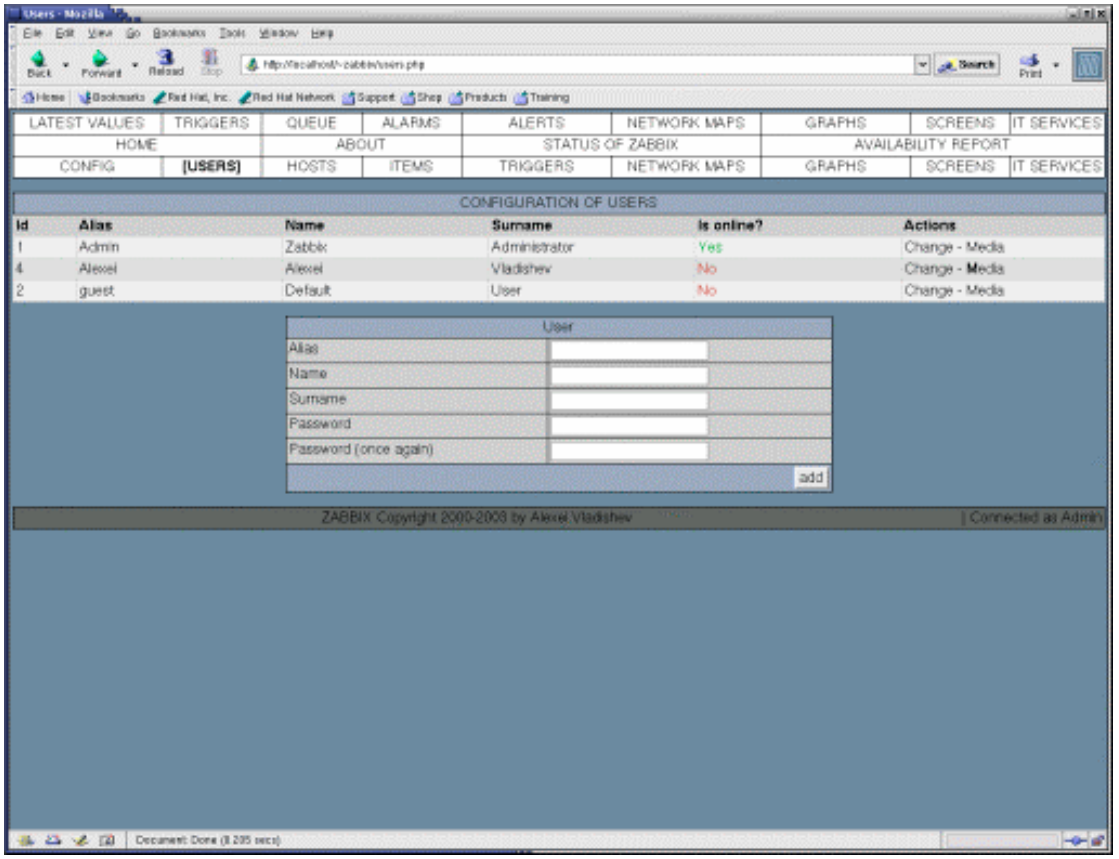
Script name

Name of the script. ZABBIX will create absolute name of the script by concatenating AlertScriptsPath and this name.
For example: *send_sms.sh*

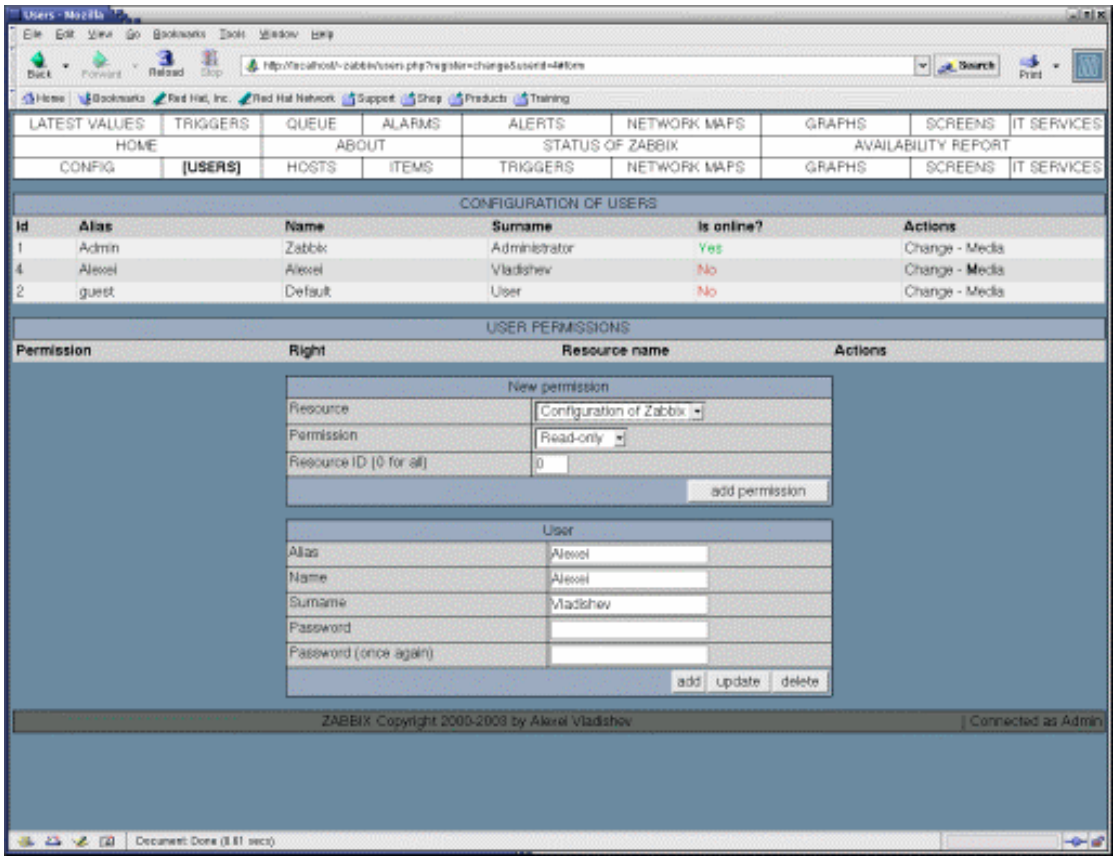
Note: ZABBIX will provide three command line parameters to the script: recipient, subject and message.

USERS

The menu item is used to configure ZABBIX users and theirs access rights.



Main menu -> USERS



Main menu -> USERS (User selected)

USER DETAILS

Alias

Short name of the user.

Example: *alexei*

Name

Name of the user.

Example: *Alexei*

Surname

Surname of the user.

Example: *Vladishev*

Password

Password. The field can be left blank.

Example: ********

PERMISSIONS

Resource

ZABBIX resource to apply selected permission. Possible values:

Permission	Affects
Configuration of Zabbix	Global configuration parameters. Screen "CONFIG".
Default permission	
Graph	
Host	
IT Service	
Item	
Network map	
Trigger's comment	
User	

Example: *"Configuration of Zabbix"*

Permission

It specifies permission for the selected resource. Possible permissions:

Permission	Description
Read-only	Allow read permissions.
Read-write	Allow read and update permissions.
Hide	Restrict access.
Add	Allow add permissions.

Example: *Read-Only*

Resource ID (0 for all)

ID of the selected resources. If 0 is specified, permission will be applied to all resources of given type.

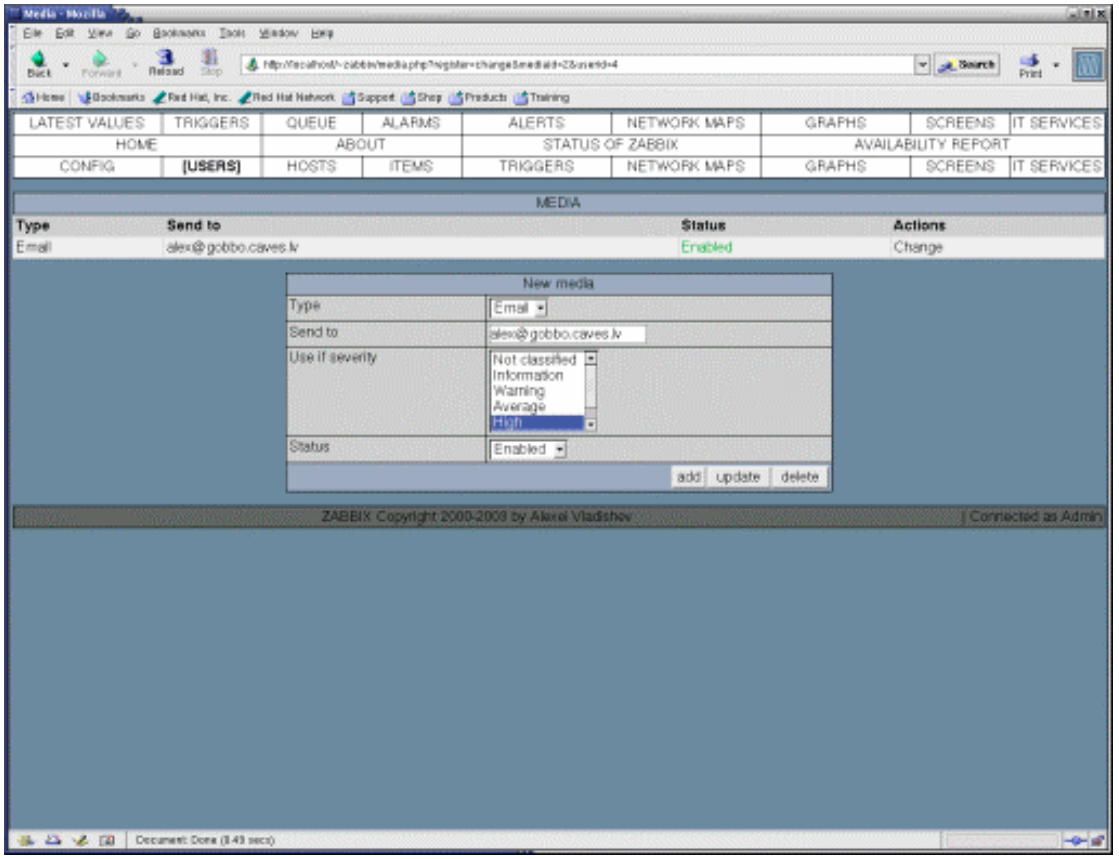
Example: *1234*

Possible actions:

Action	Result
Press **[update]** button ...not finished...	Configuration parameters will be updated

MEDIA

The screen is used to configure notification channels for a selected user.



Type

Type of media (delivery channel). Currently, *E-mail* is the only supported type of media.

Example: *E-mail*

Send to

Address to send messages to.

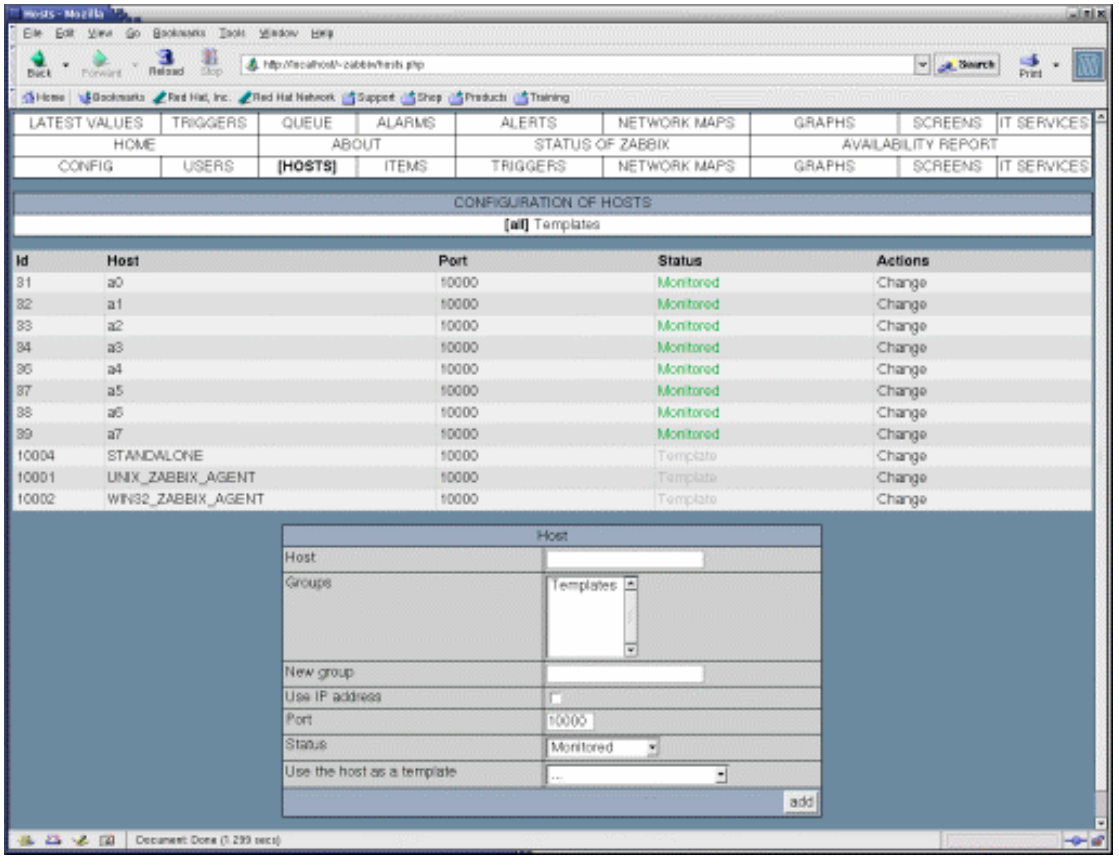
Example: alex@gobbo.caves.lv

Use if severity

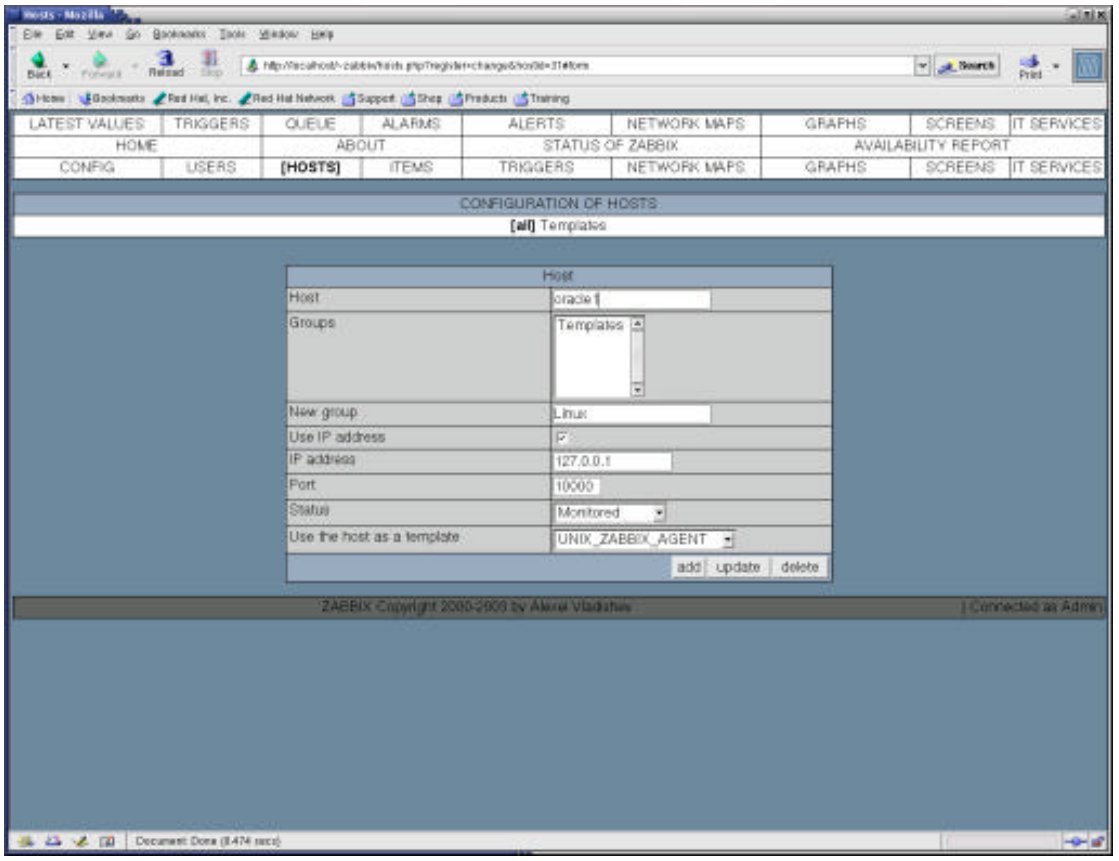
Select severity for this media. If a trigger's severity is in the list, then the Media will be used for delivery of this alert.

HOSTS

The menu item is used to configure hosts for monitoring.



Main menu -> HOSTS



Main menu -> HOSTS (Host selected)

Host

Host name. Full or short DNS name of the host is usually used.

Example: *zabbix.sourceforge.net*

Groups

Host groups. Select zero or several host groups.

New group

New host group.

Example: *Database servers*

Use IP address

Select this checkbox if you want to connect to the host using IP address.

IP address

Host IP address.
`

Example: *127.0.0.1*

Port

This parameter specifies port number of ZABBIX agent on this host.

Example: *10000*

Status

Host status. Possible values:

Host status	Description
Monitored	ZABBIX will monitor this host
Not monitored	ZABBIX will not monitor this host.
Template	Template. ZABBIX will not monitor this host.

Example: *Monitored*

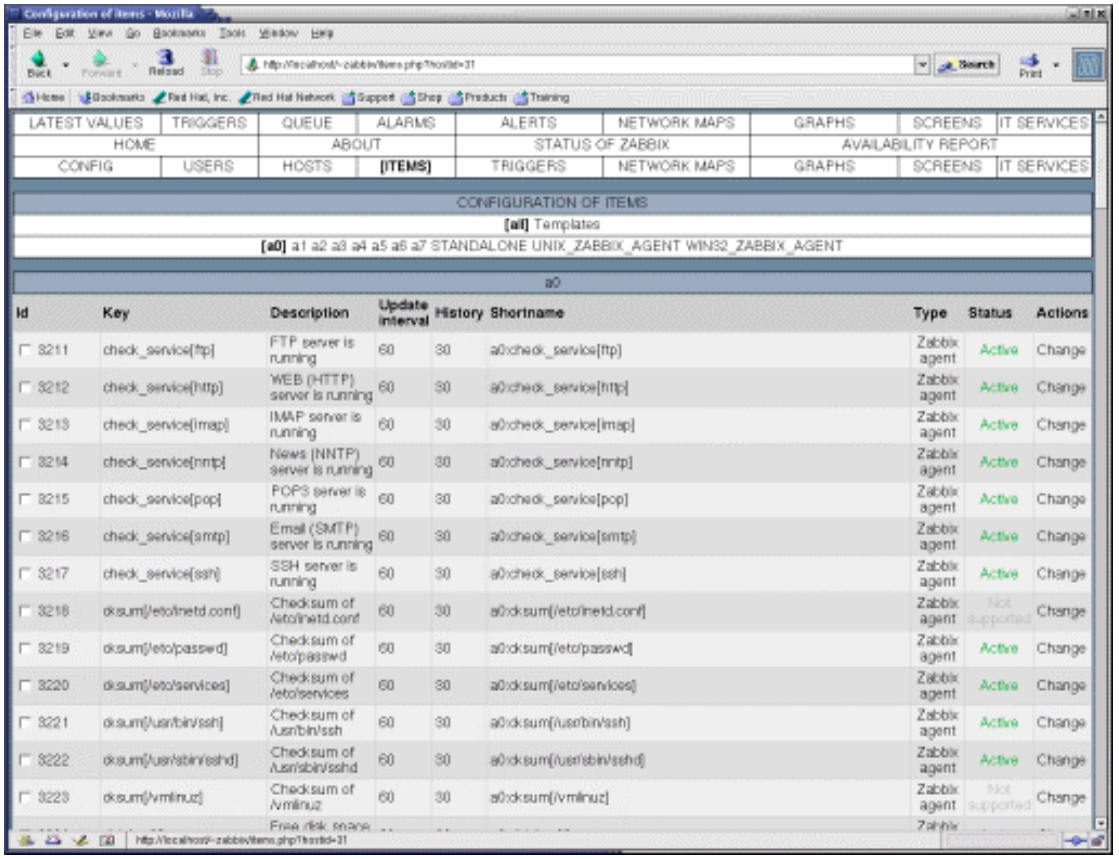
Use the host as a template

Add items and triggers of selected template host. This will greatly simplify addition of hosts having similar configuration. Pre-defined templates:

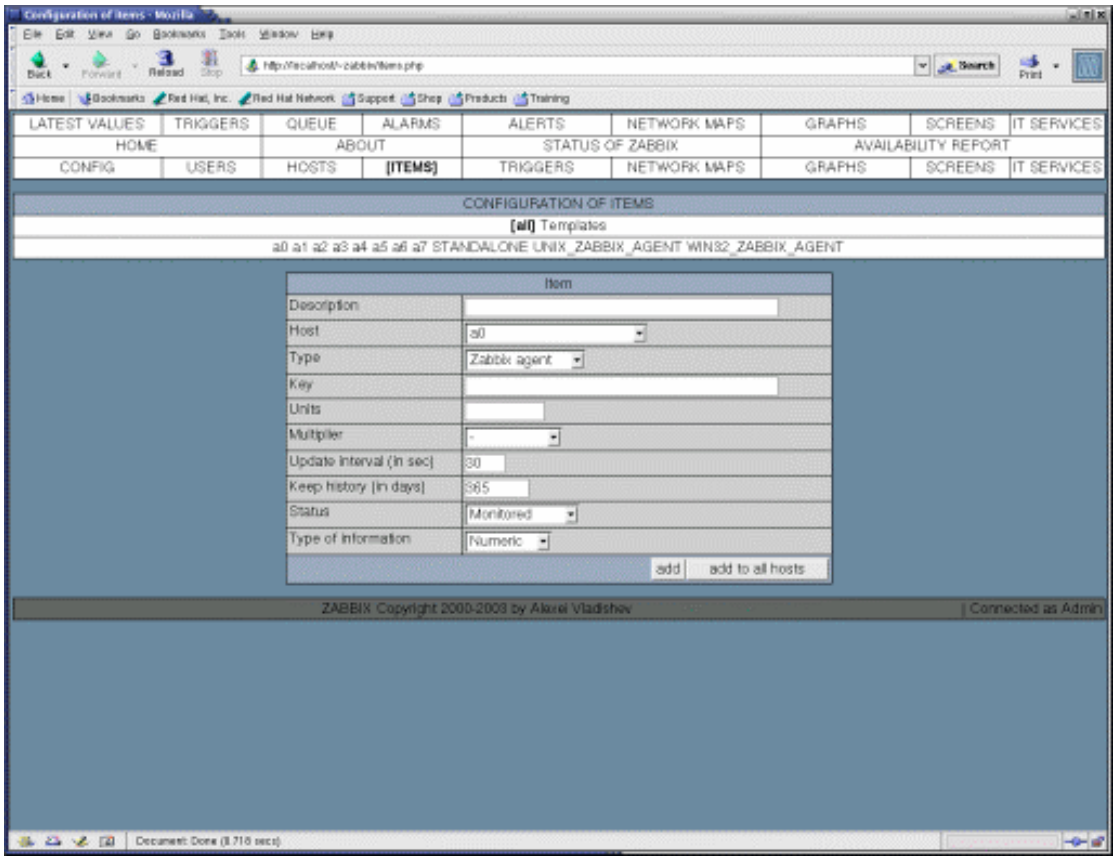
Template	Description
STANDALONE	Host without any agents running. ZABBIX will perform basic service checks.
UNIX_ZABBIX_AGENT	Host having **zabbix_agentd** running. All parameters supported by **zabbix_agentd** will be performed.
WIN32_ZABBIX_AGENT	Host having WIN32 **zabbix_agentd** running. All parameters supported by WIN32 **zabbix_agentd** will be performed. Used for monitoring of Windows-based servers and workstations.

ITEMS

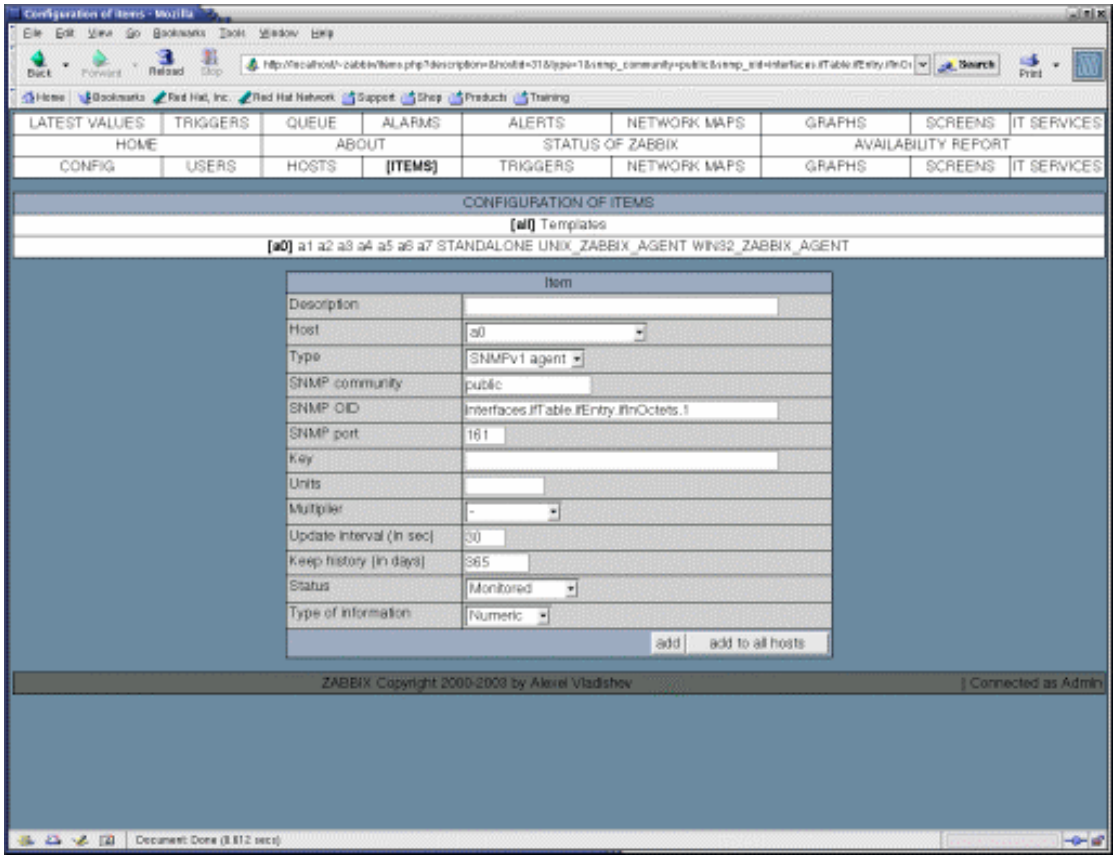
The menu item is used to configure parameters for monitoring.



Description



Description (Item type: ZABBIX agent)



Description (Item type: SNMPv1 agent)

Description of parameter for monitoring.

Example: *Processor load on host %s*

Host

Host name this item belongs to.

Type

Item type. Possible values:

Item type	Description
ZABBIX agent	This item type assumes ZABBIX agent running on the selected host.
Simple check	No agent required. Used to check availability of simple services (HTTP, SSH, SMTP, etc)
SNMPv1 agent	This item type assume SNMPv1 agent running on device (host, router, printer, etc) being monitored.
SNMPv2 agent	Same as above. SNMPv2 agent is required.
ZABBIX trapper	**zabbox_trapperd** must be running in order to process received values usually sent by **zabbix_sender.**

SNMP community (for SNMP only)

SNMP community name. *Public* is usually used.

Example: *public*.

SNMP OID (for SNMP only)

SNMP Object ID.

Example: *interfaces.ifTable.ifEntry.ifnOctets.1*.

Key

[Unfinished…]

Update interval (in sec)

This parameter defines how often **zabbix_suckerd** must retrieve values. This parameter does not affect item type 'ZABBIX trapper'. The update interval is also used to calculate refresh rate for graphs generated by PHP front-end.

Type of information

Can be "Numeric" or "String". Numeric values are stored in table HISTORY. String values are stored in table HISTORY_STR. PHP front-end cannot generate graphs based on "String" values.

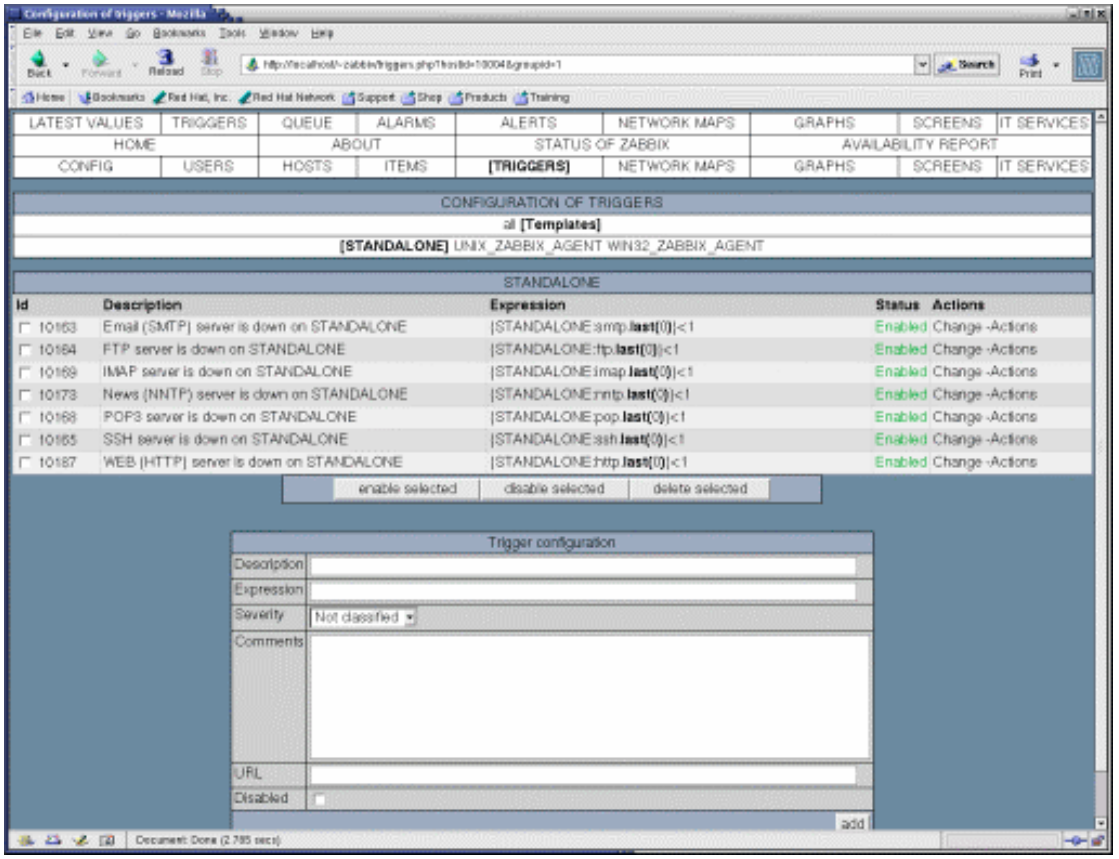
Allowed hosts (for trapper items only)

This parameter defines list of IP addresses which may send values. If left empty, all hosts are allowed.

[Unfinished…]

TRIGGERS

The menu item is used to configure triggers.



Description
Trigger description.

> **Note:** Macro %s will be substituted by host name. This is required to allow the trigger to be used for a template host.

Example: *Processor load on %s is too high*.

Expression

Logical expression for the trigger.

Severity

Possible values: [to be finished]

Comments

For informational purposes only. Usually used to provide instructions for problem resolution.

URL

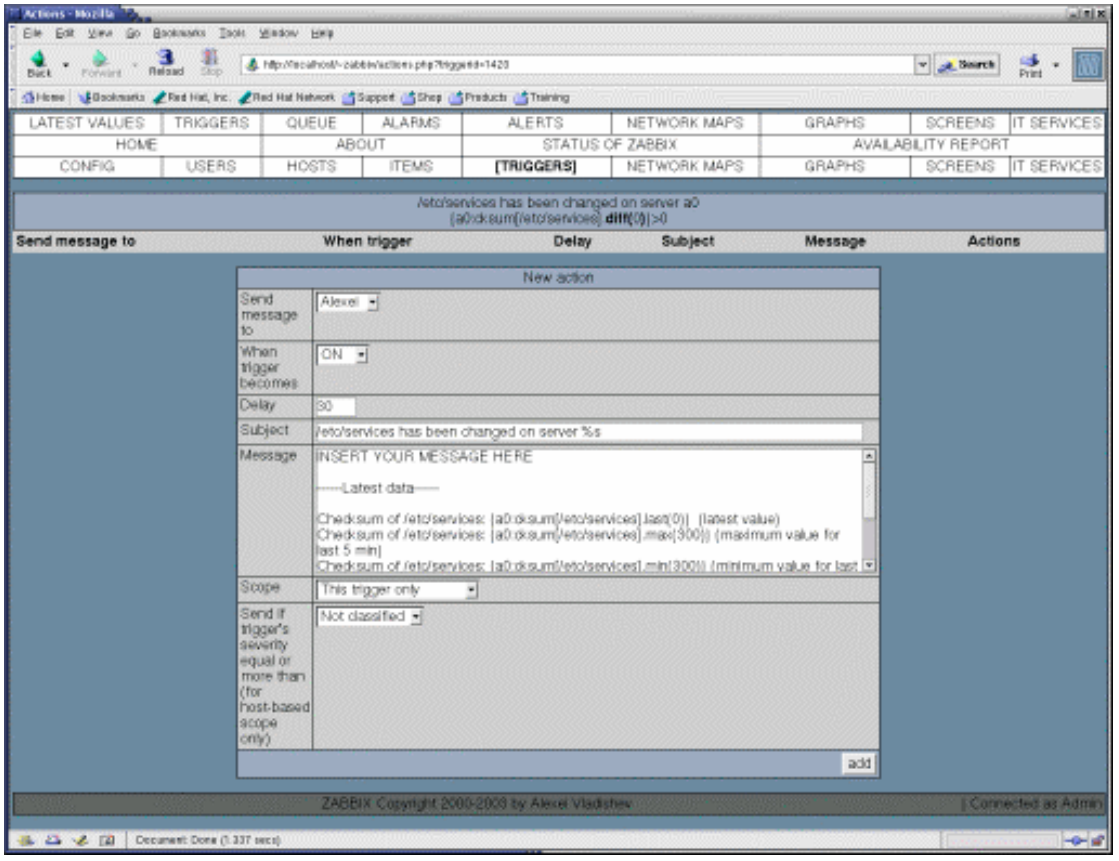
For informational purposes only. If defined, the link will be shown in screen "Status of Triggers". Usually used to provide additional information for the trigger.

Example: if trigger defines availability of WEB server, the URL may point to the server.

Disabled

If disabled, the trigger does not generate alarms and therefore no actions performed.

[... Not finished ...]



Actions (Configuration of trigger actions)

Send message to

ZABBIX user to send this message to.

When trigger becomes

ON – Action will be applied if trigger value changes value from FALSE to TRUE (or FALSE->UNKNOWN->TRUE)

OFF – Action will be applied if trigger value changes value from TRUE to FALSE (or TRUE->UNKNOWN->FALSE)

Delay

ZABBIX will not send more than one message within this period of time.

Subject

Subject of the message. Macros will be substituted.

Message

The message itself. Macros will be substituted.

Scope

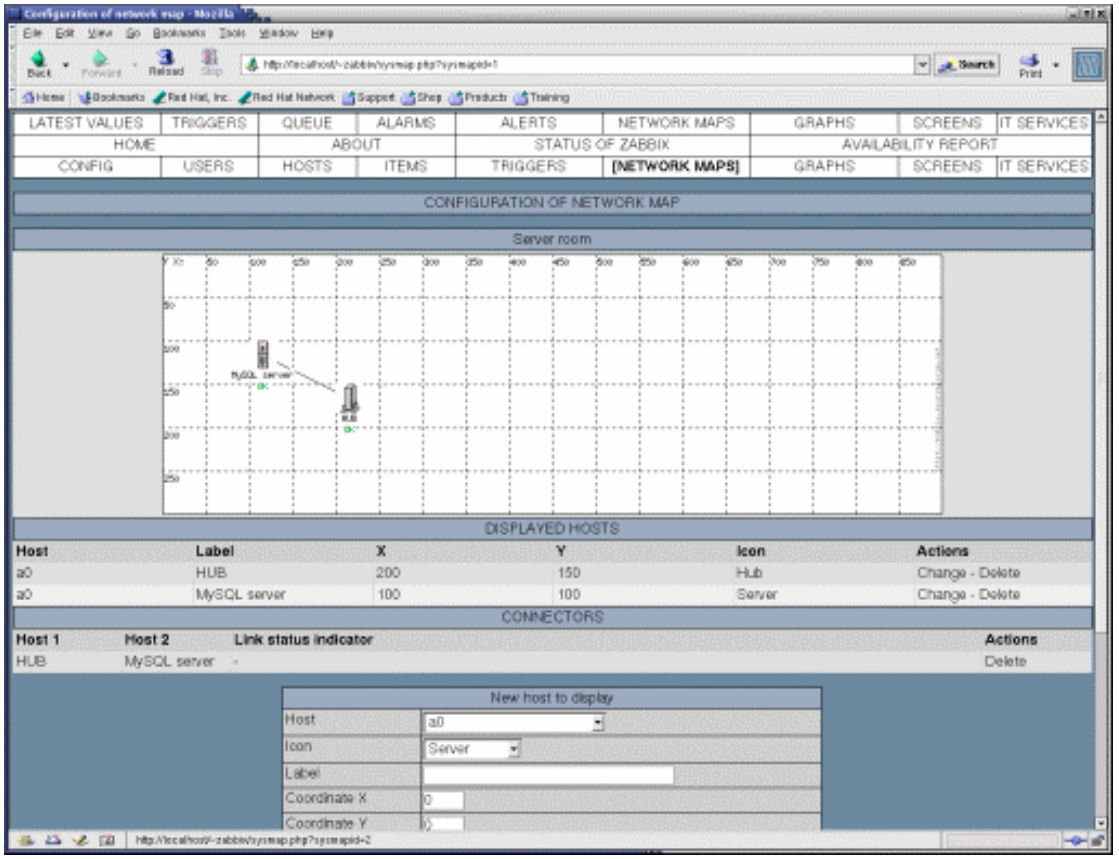
Scope	Description
This trigger only	Action is performed for this trigger only
All triggers of this host	Global action. Action performed for all triggers of all hosts related to this trigger if trigger severity is equal or more than severity of this action.
All triggers	Global action. Action performed for all triggers if trigger severity is equal or more than severity of this action.

Send of trigger's severity is equal or more than

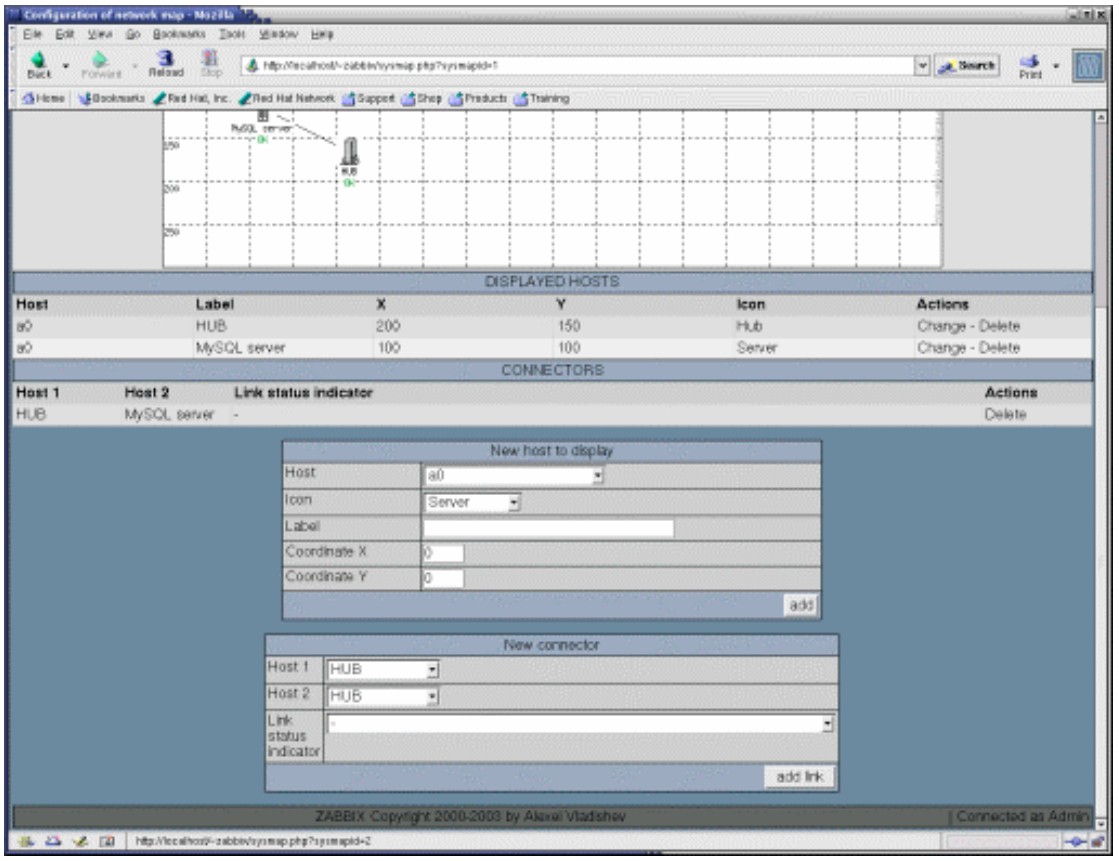
For scope "All triggers of this host" and "All triggers" only.

NETWORK MAPS

The menu item is used to configure network maps.



Screen1



Screen2

Name

Name of the network map.

Example: *Server room*.

Width

Width of the network map in pixels.

Example: *800*.

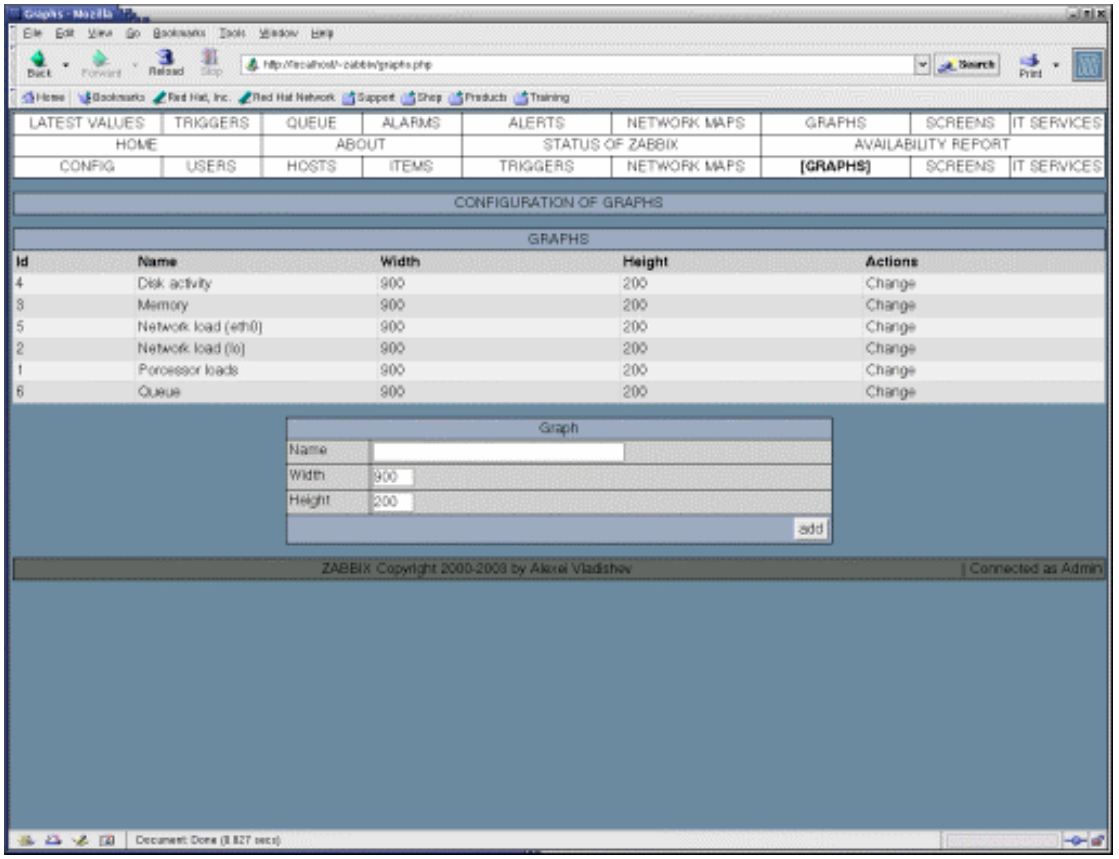
Height

Height of the network map in pixels.

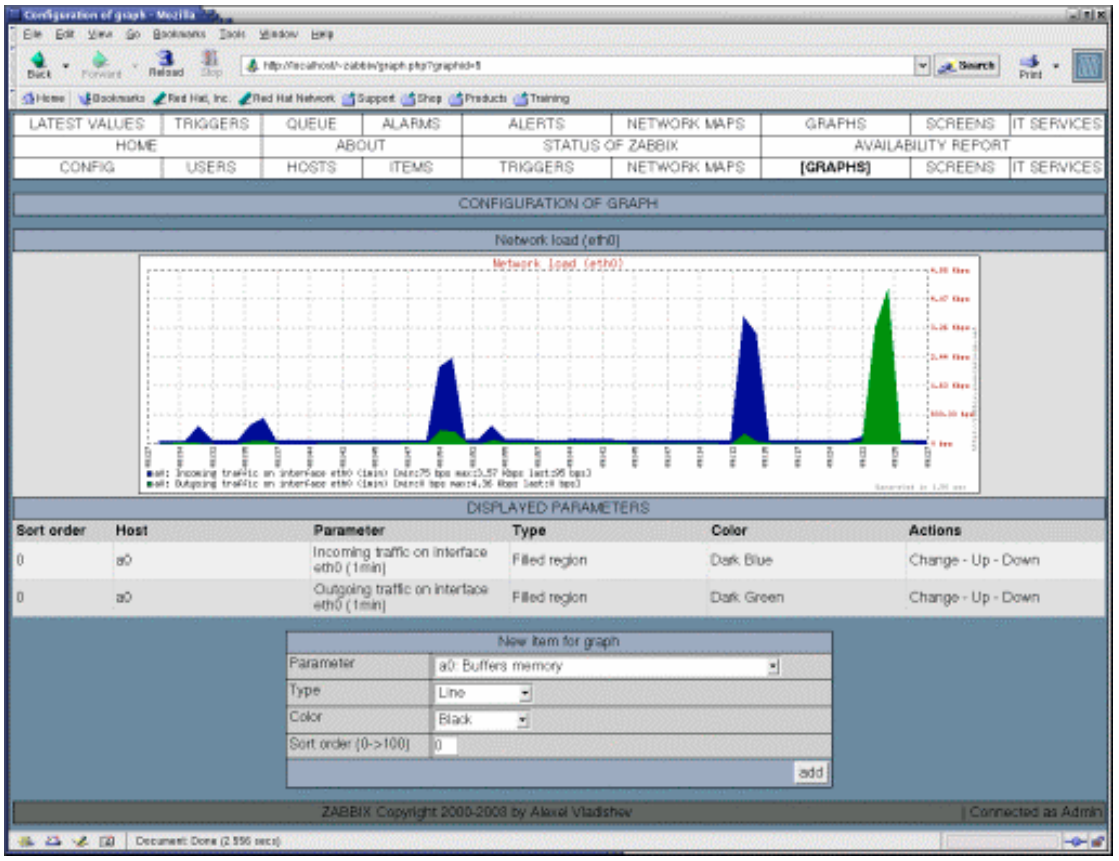
Example: *600*.

GRAPHS

The menu item is used to configure user-defined (complex) graphs.



List of graphs



Configuration of graph

Name

Name of the graph.

Example: *Network load on WEB server*.

Width

Width of the graph in pixels.

Example: *800*.

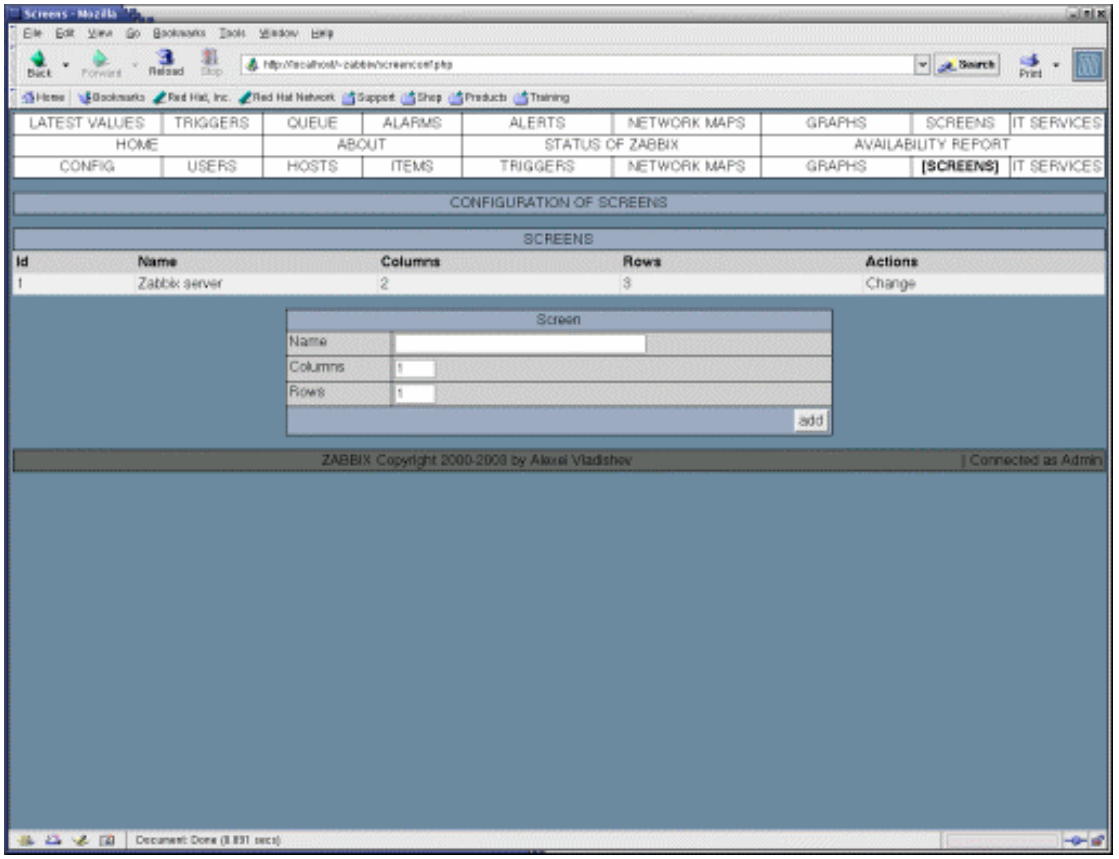
Height

Height of the graph in pixels.

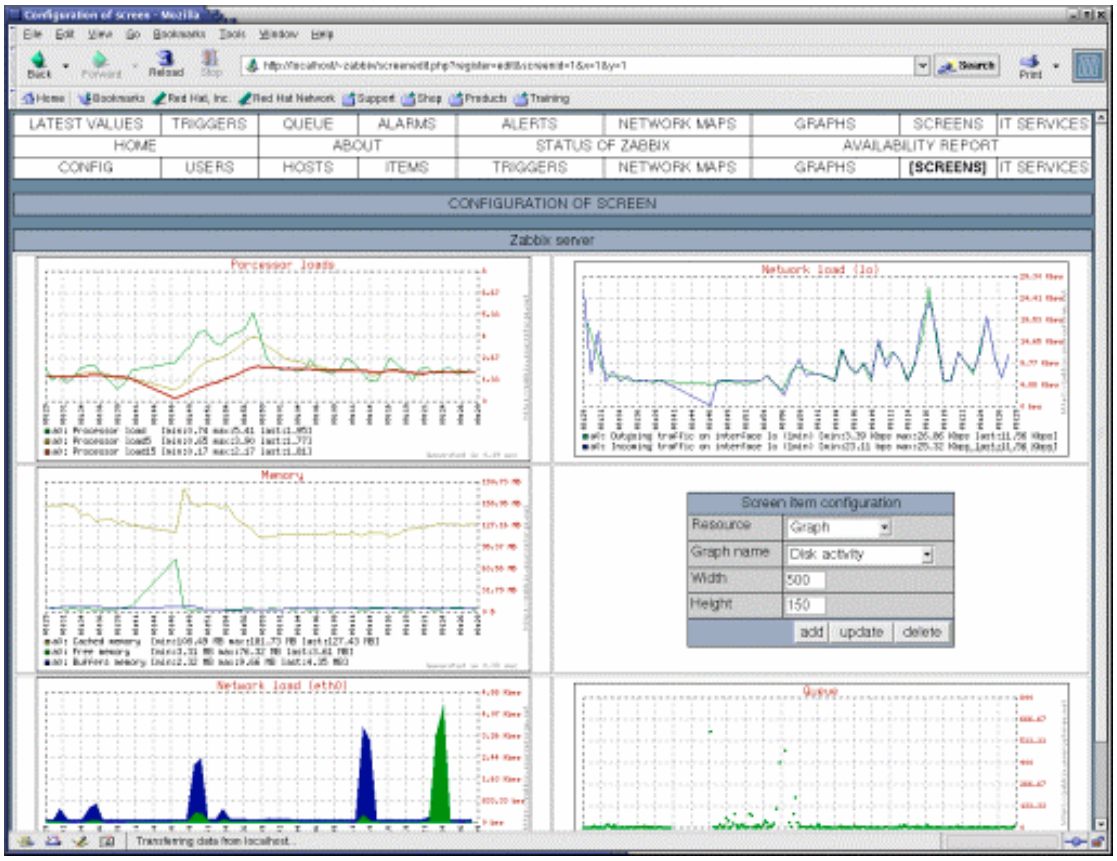
Example: *600*.

SCREENS

The menu item is used to configure screens.



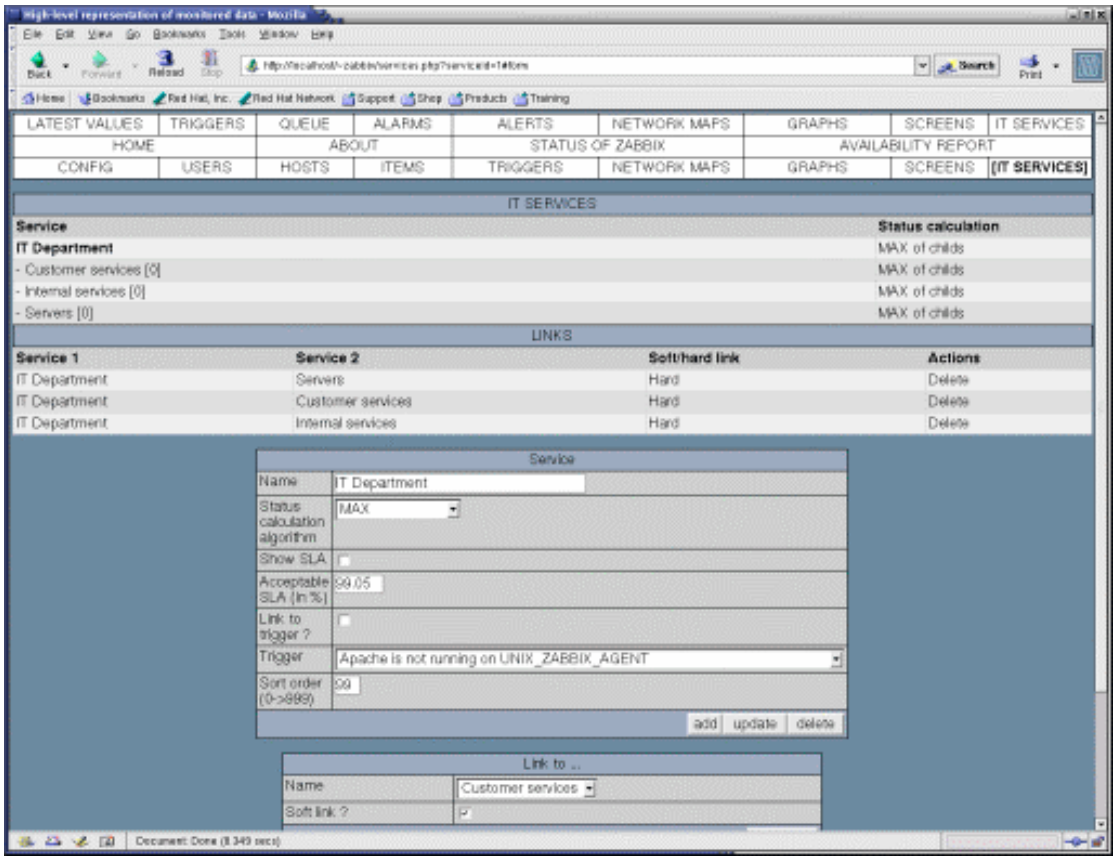
List of screens



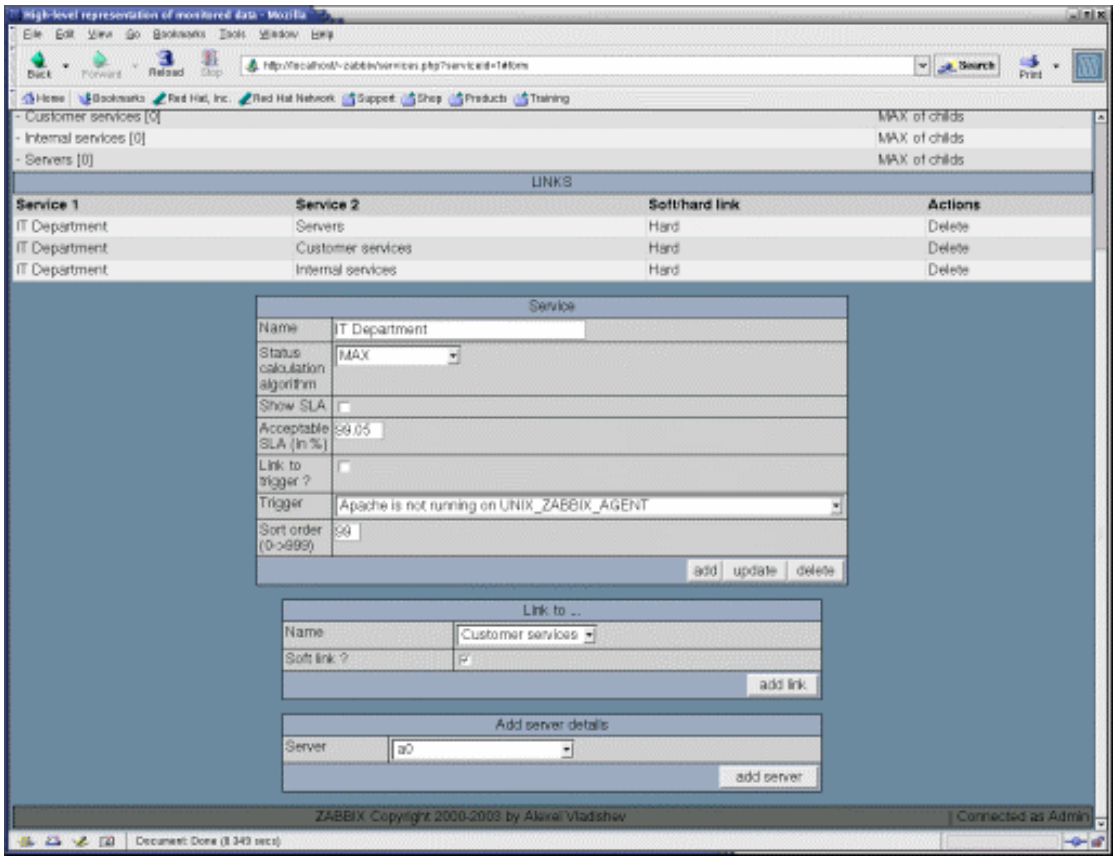
Configuration of screen

IT SERVICES

The menu item is used to configure structure of high-level IT service tree.



Screen1



Screen2

New service

Name

Name of the service.

Examples: “London office”, “Oracle server”, “Routers”.

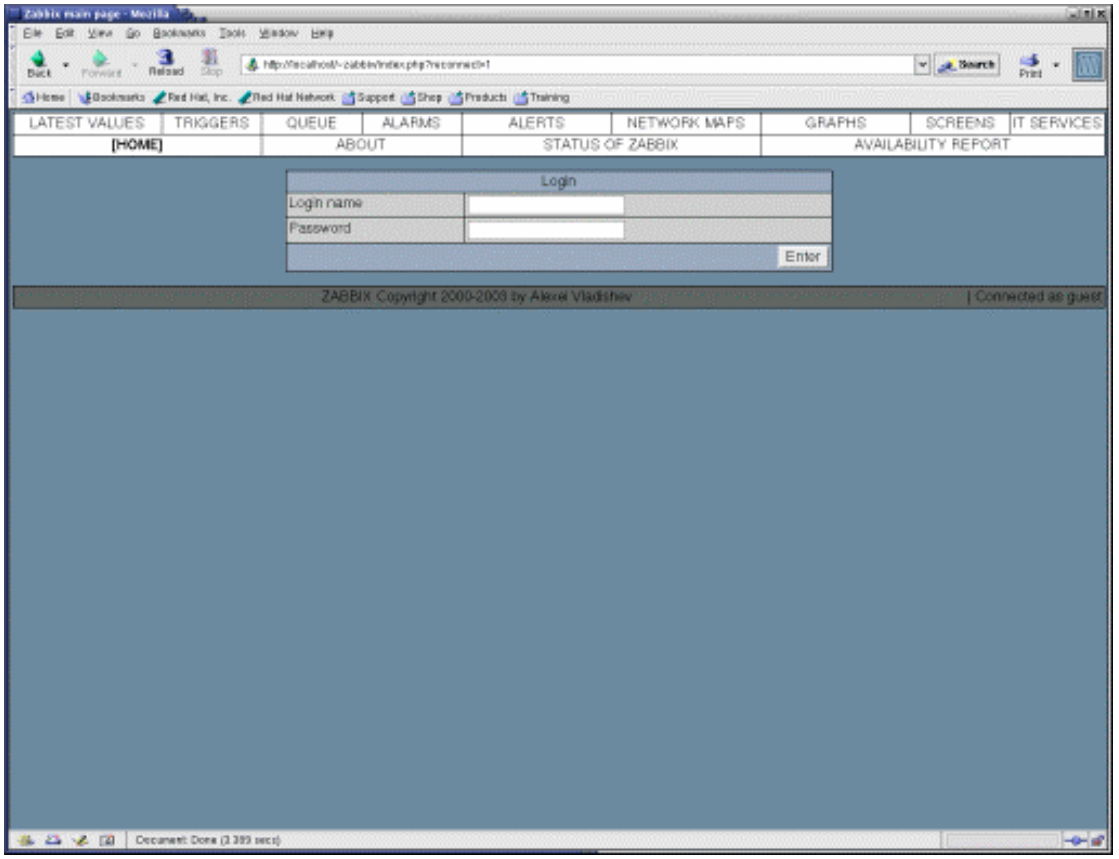
Link to trigger

Status of the trigger will be propagated to the service.

[to be finished]

HOME

This is login screen.



Login name

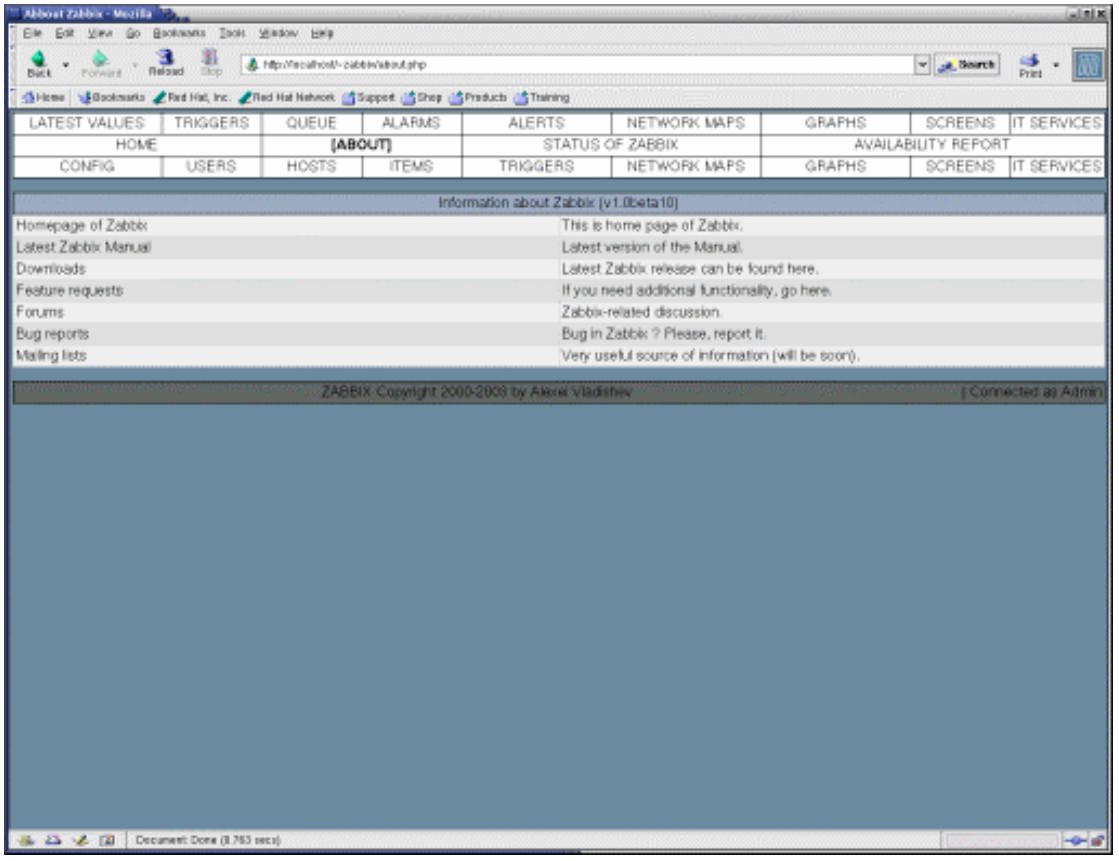
Login name. When ZABBIX is first installed, user 'Admin' with empty password exists.

Password

User's password. Can be empty.
Note that ZABBIX logs users out after 30 minutes of inactivity.

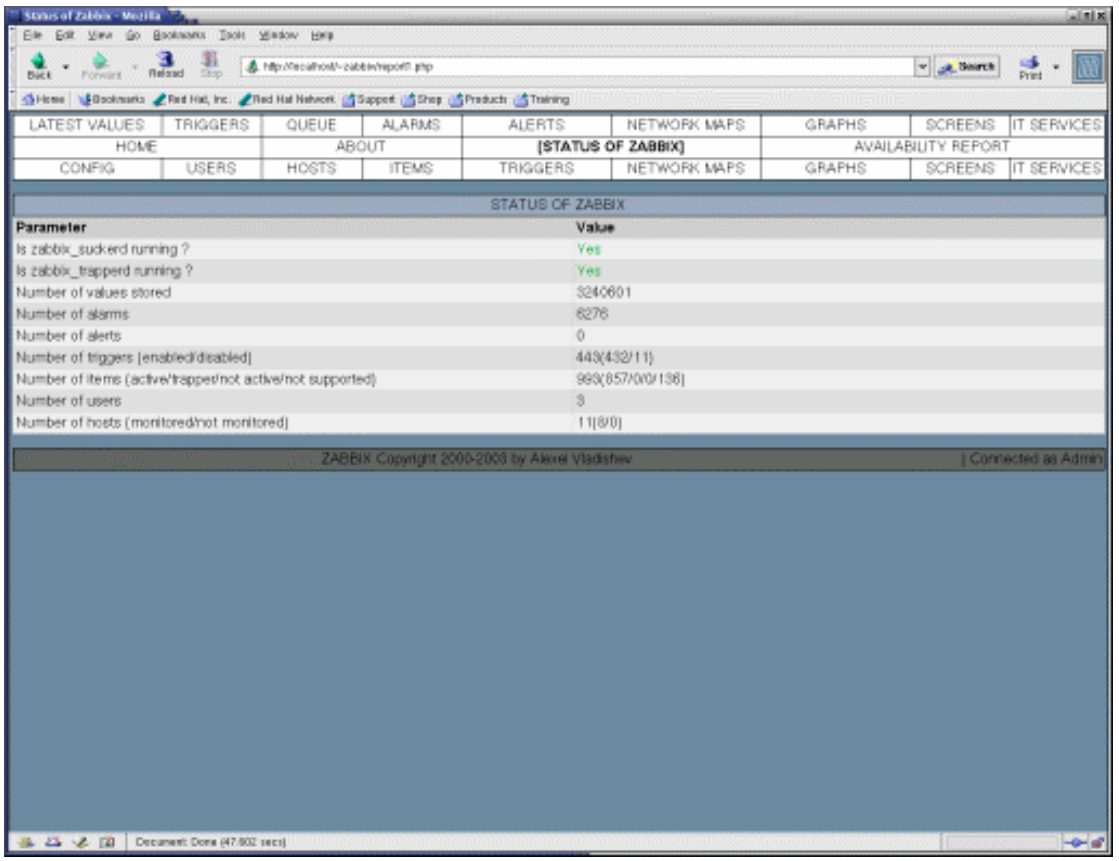
ABOUT

This screen provides pointers to ZABBIX related information. The screen also shows the version of ZABBIX software currently running.



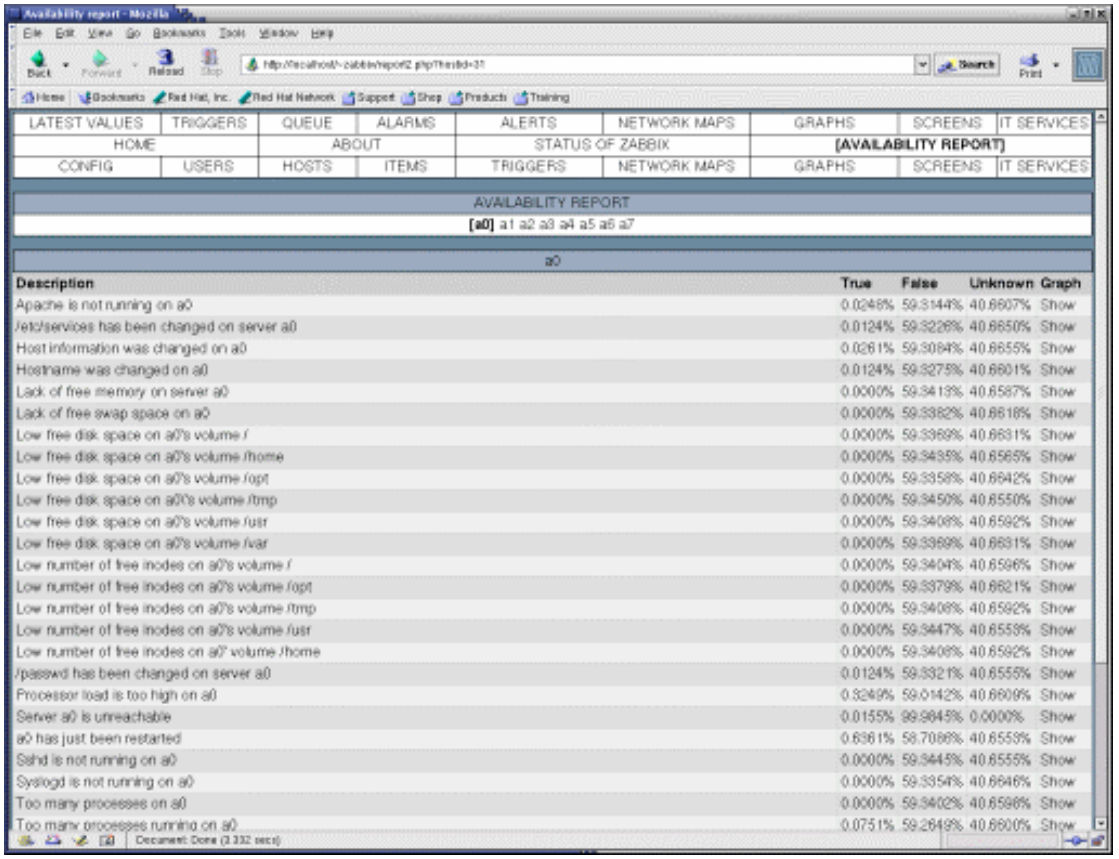
STATUS OF ZABBIX

This report gives general information about the ZABBIX server.



AVAILABILITY REPORT

This report calculates amount of time a trigger has been in true and false states. For example, suppose you want to know what percentage of the time your web server is down. Assuming you have been monitoring the web server and have the “WEB server is down” trigger enabled, the availability report will display what percentage of the time the trigger was in the true state (web server down) and in the false state (web server running).



LATEST VALUES	TRIGGERS	QUEUE	ALARMS	ALERTS	NETWORK MAPS	GRAPHS	SCREENS	IT SERVICES
HOME		ABOUT		STATUS OF ZABBIX		[AVAILABILITY REPORT]		
CONFIG	USERS	HOSTS	ITEMS	TRIGGERS	NETWORK MAPS	GRAPHS	SCREENS	IT SERVICES

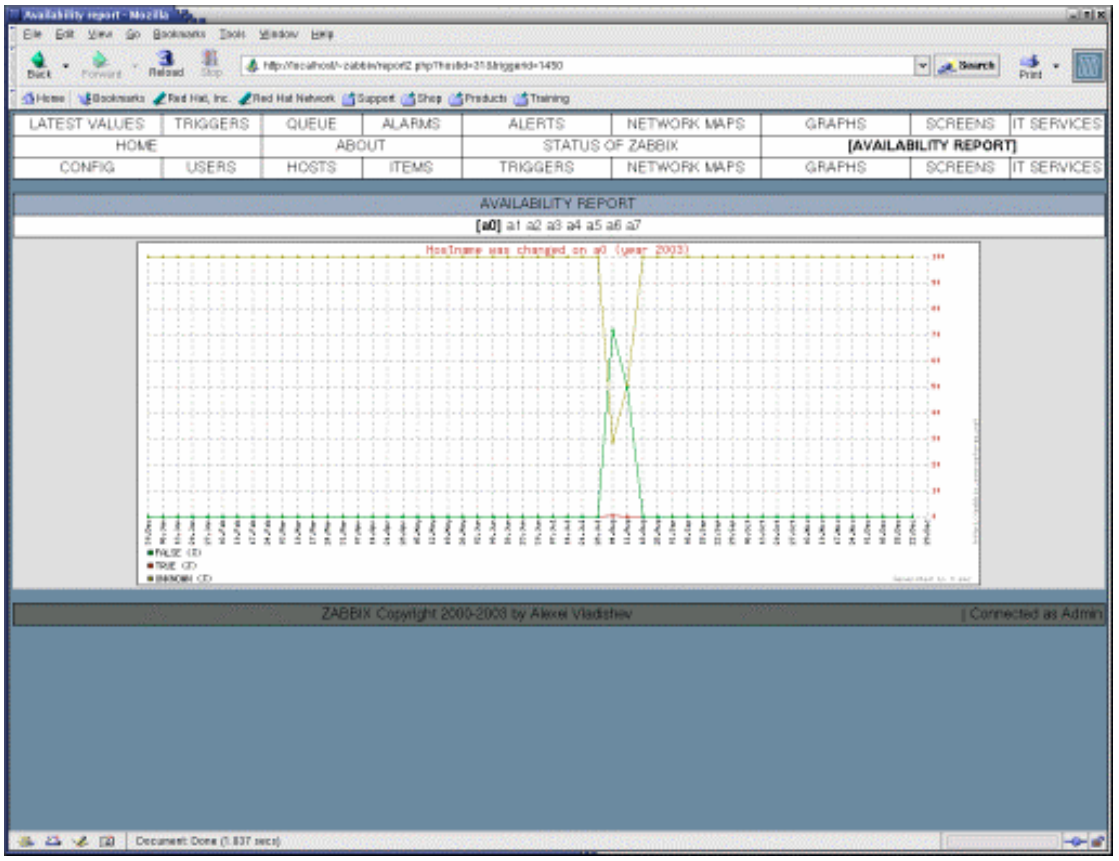
AVAILABILITY REPORT

[a0] a1 a2 a3 a4 a5 a6 a7

a0

Description	True	False	Unknown	Graph
Apache is not running on a0	0.0248%	59.3144%	40.6607%	Show
/etc/services has been changed on server a0	0.0124%	59.3228%	40.6650%	Show
Host information was changed on a0	0.0261%	59.3084%	40.6655%	Show
Hostname was changed on a0	0.0124%	59.3275%	40.6601%	Show
Lack of free memory on server a0	0.0000%	59.3413%	40.6587%	Show
Lack of free swap space on a0	0.0000%	59.3382%	40.6618%	Show
Low free disk space on a0's volume /	0.0000%	59.3369%	40.6631%	Show
Low free disk space on a0's volume /home	0.0000%	59.3435%	40.6565%	Show
Low free disk space on a0's volume /opt	0.0000%	59.3358%	40.6642%	Show
Low free disk space on a0's volume /tmp	0.0000%	59.3450%	40.6550%	Show
Low free disk space on a0's volume /usr	0.0000%	59.3408%	40.6592%	Show
Low free disk space on a0's volume /var	0.0000%	59.3369%	40.6631%	Show
Low number of free inodes on a0's volume /	0.0000%	59.3404%	40.6596%	Show
Low number of free inodes on a0's volume /opt	0.0000%	59.3379%	40.6621%	Show
Low number of free inodes on a0's volume /tmp	0.0000%	59.3408%	40.6592%	Show
Low number of free inodes on a0's volume /usr	0.0000%	59.3447%	40.6553%	Show
Low number of free inodes on a0' volume /home	0.0000%	59.3408%	40.6592%	Show
/passwd has been changed on server a0	0.0124%	59.3321%	40.6555%	Show
Processor load is too high on a0	0.3249%	59.0142%	40.6609%	Show
Server a0 is unreachable	0.0155%	99.9845%	0.0000%	Show
a0 has just been restarted	0.6361%	58.7086%	40.6553%	Show
Sshd is not running on a0	0.0000%	59.3445%	40.6555%	Show
Syslogd is not running on a0	0.0000%	59.3354%	40.6646%	Show
Too many processes on a0	0.0000%	59.3402%	40.6598%	Show
Too many processes running on a0	0.0751%	59.2649%	40.6600%	Show

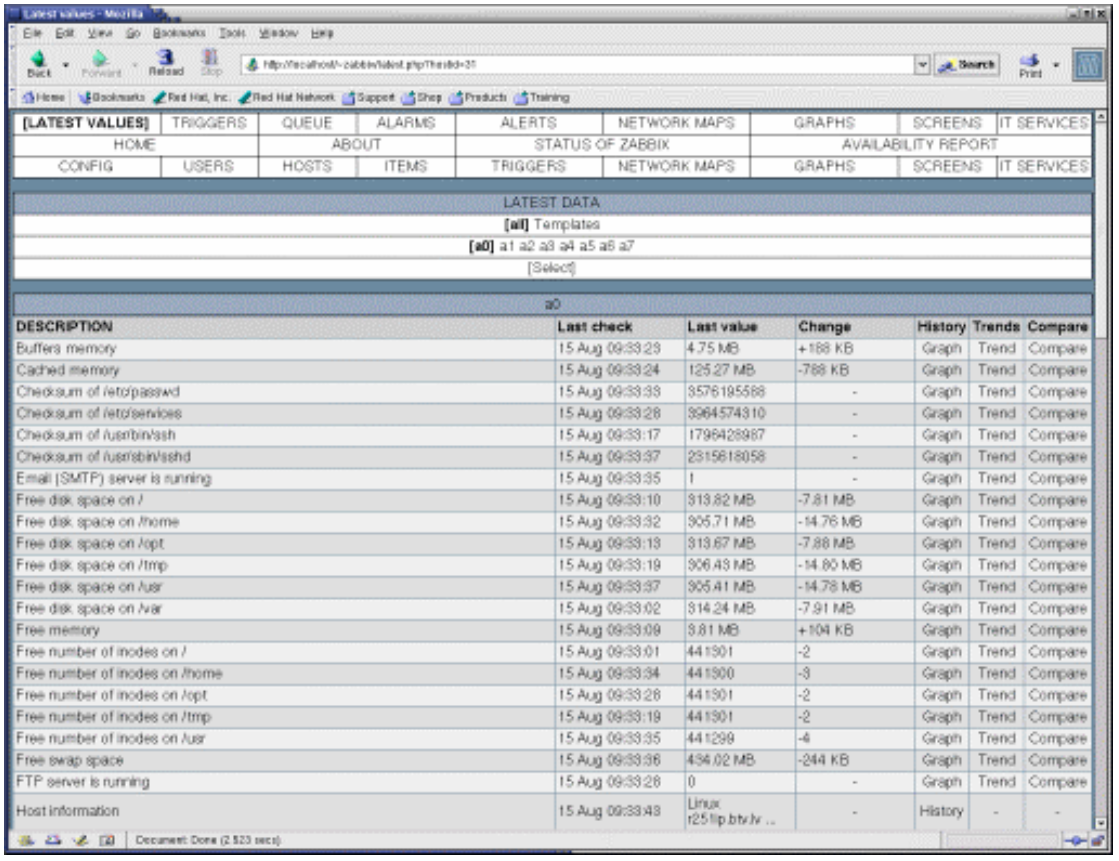
Screen1


LATEST VALUES
TRIGGERS
QUEUE
ALARMS
ALERTS
NETWORK MAPS
GRAPHS
SCREENS
IT SERVICES
HOME
ABOUT
STATUS OF ZABBIX
[AVAILABILITY REPORT]
CONFIG
USERS
HOSTS
ITEMS
TRIGGERS
NETWORK MAPS
GRAPHS
SCREENS
IT SERVICES
AVAILABILITY REPORT
[a0] a1 a2 a3 a4 a5 a6 a7
ZABBIX Copyright 2000-2003 by Alexei Vladishev
[Connected as Admin]

Screen2

LATEST VALUES

The screen shows latest values of monitored parameters. The screen opens access to historical data, graphs and trends of collected information.

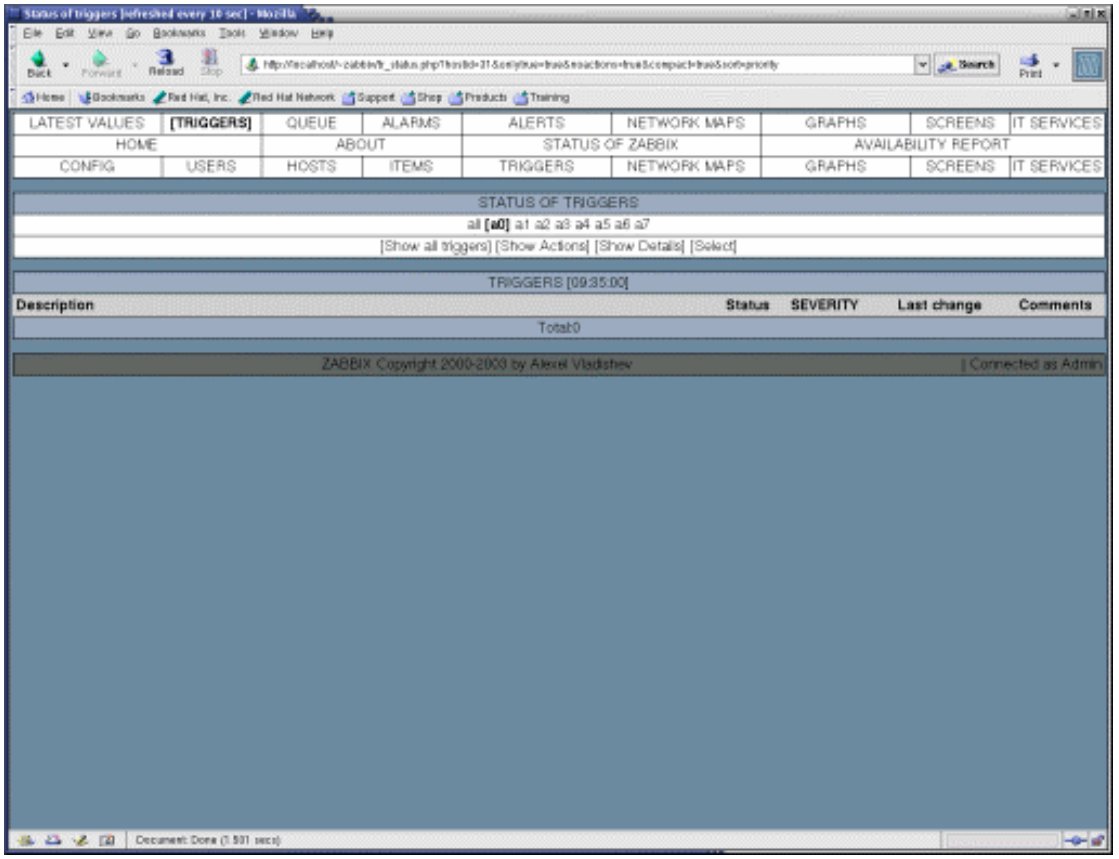


Possible actions:

Action	Result
Click on a host group	Displayed hosts will be limited to members of the host group
Click on a host	Monitored parameter of the host will only be displayed
Click on **[Select]**	Input box for selecting parameters by description will be displayed
Click on **[Graph]**	Graph with most recent values of the parameter will be displayed
Click on **[Trend]**	Trends of the parameter will be displayed
Click on **[Compare]**	Graph with most recent and week average values of the parameter will be displayed

STATUS OF TRIGGERS

The screen shows latest status of triggers. Be default, this screen shows all triggers in TRUE state.

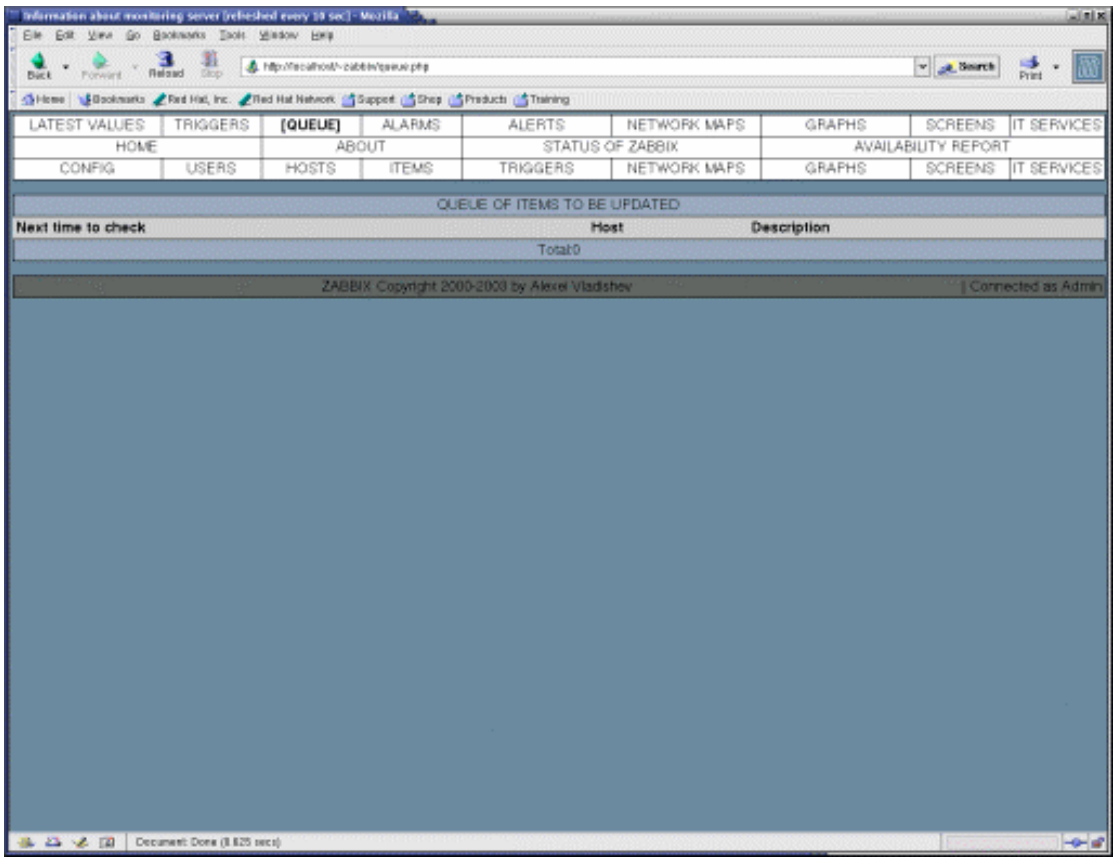


Possible actions:

Action	Result
Click on a host	Triggers of the selected host will only be displayed
Click on **[Show all triggers]**	All triggers regardless of status will be displayed
Click on **[Show actions]**	
Click on **[Show details]**	Trigger expressions will be displayed
Click on **[Select]**	Input box for selecting triggers by description will be displayed
Click on **[Add]** (comment)	Screen for modifying trigger comment (detailed description) will be displayed

QUEUE

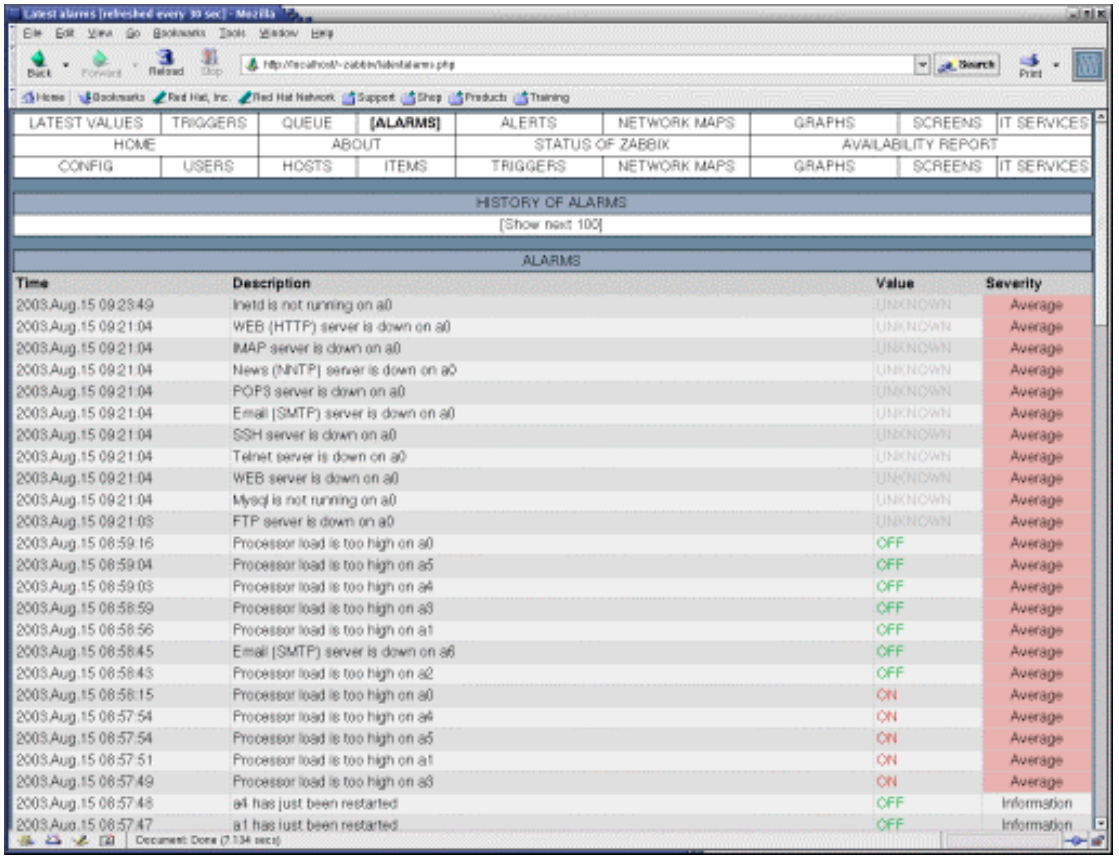
The screen shows list of parameters that must be immediately retrieved.



Length of the queue is indication of ZABBIX performance. Ideally no items should be in the queue at any moment. If length of the queue is large, then performance tuning of ZABBIX is required.

ALARMS

The screen shows list of alarms.

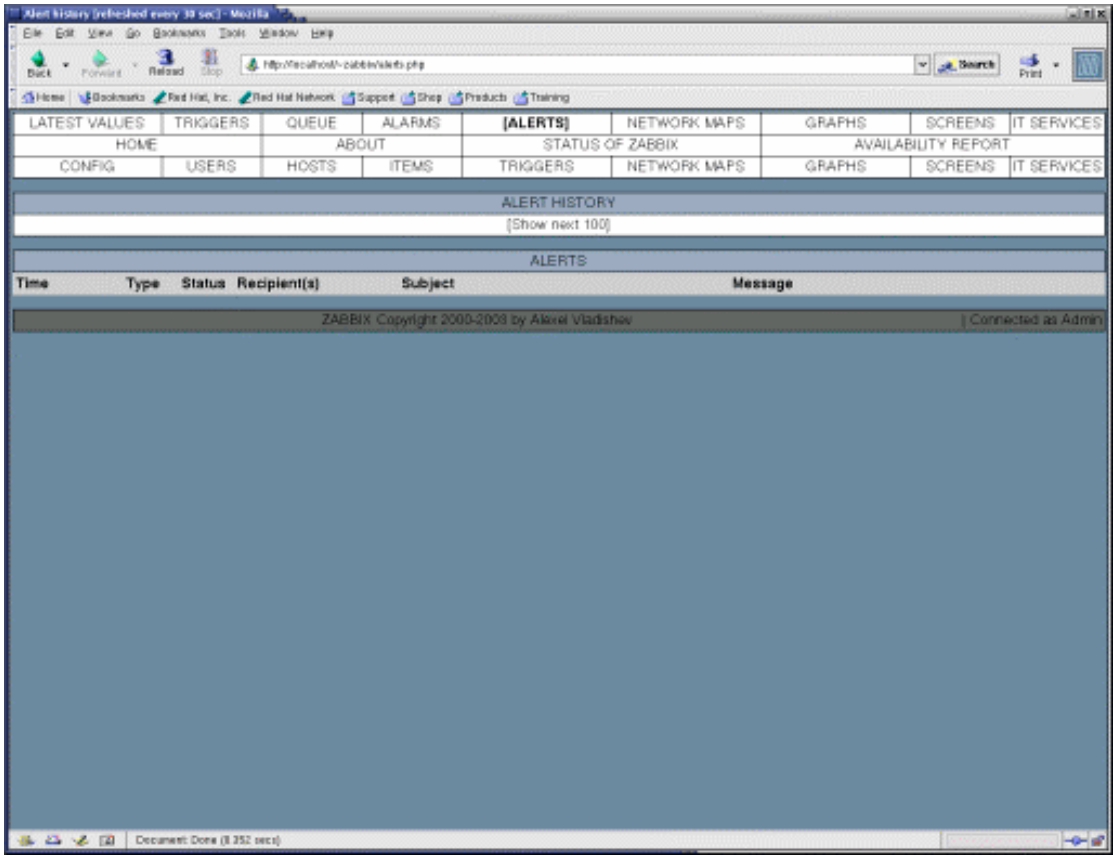


Possible actions:

Action	Result
Click on a **[Select next 100]**	Next 100 alarms will be displayed
Click on a **[Select prev 100]**	Previous 100 alarms will be displayed

ALERT HISTORY

The screen shows list of alerts.

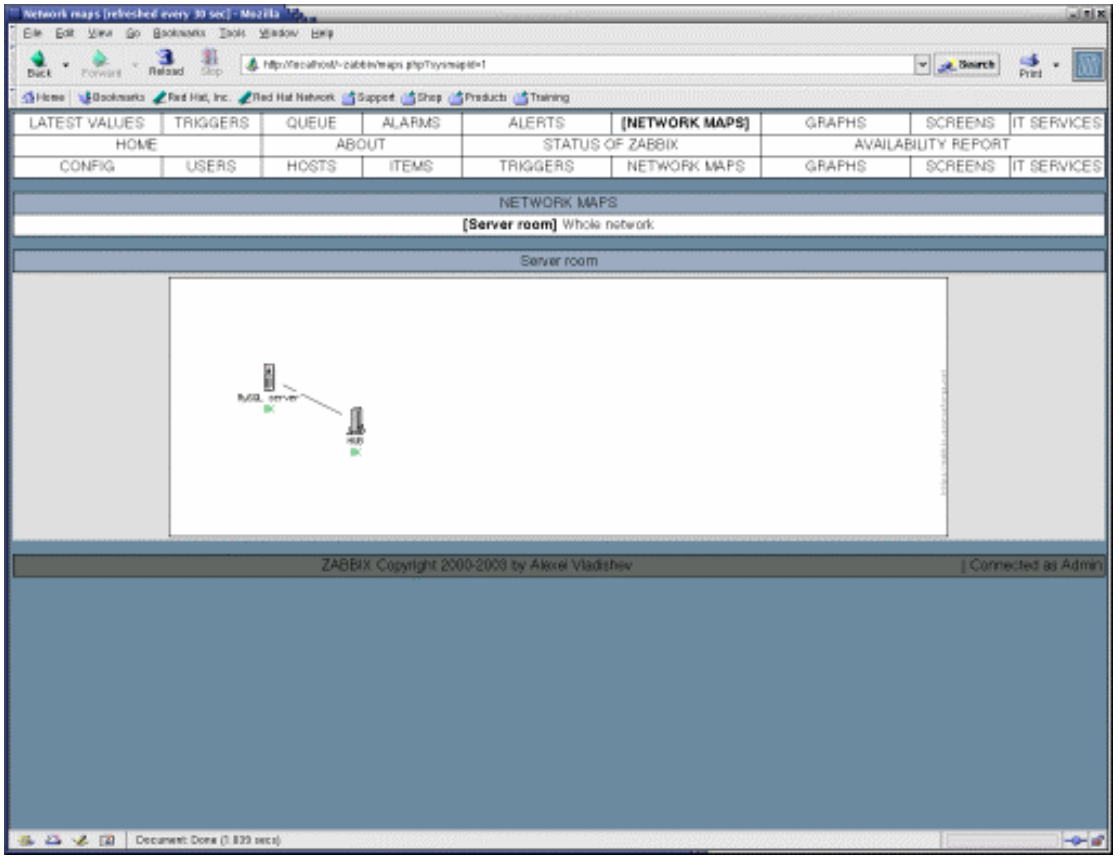


Possible actions:

Action	Result
Click on a **[Select next 100]**	Next 100 alerts will be displayed
Click on a **[Select prev 100]**	Previous 100 alerts will be displayed

NETWORK MAPS

The screen gives access to defined network maps.



Status of each host (icon) is indicated by:

- **OK** (no problems for this host)
- **Problems** (more than one trigger in TRUE state for this host)
- **Trigger description** (exactly one trigger in TRUE state)

Status of each connector is indicated by:

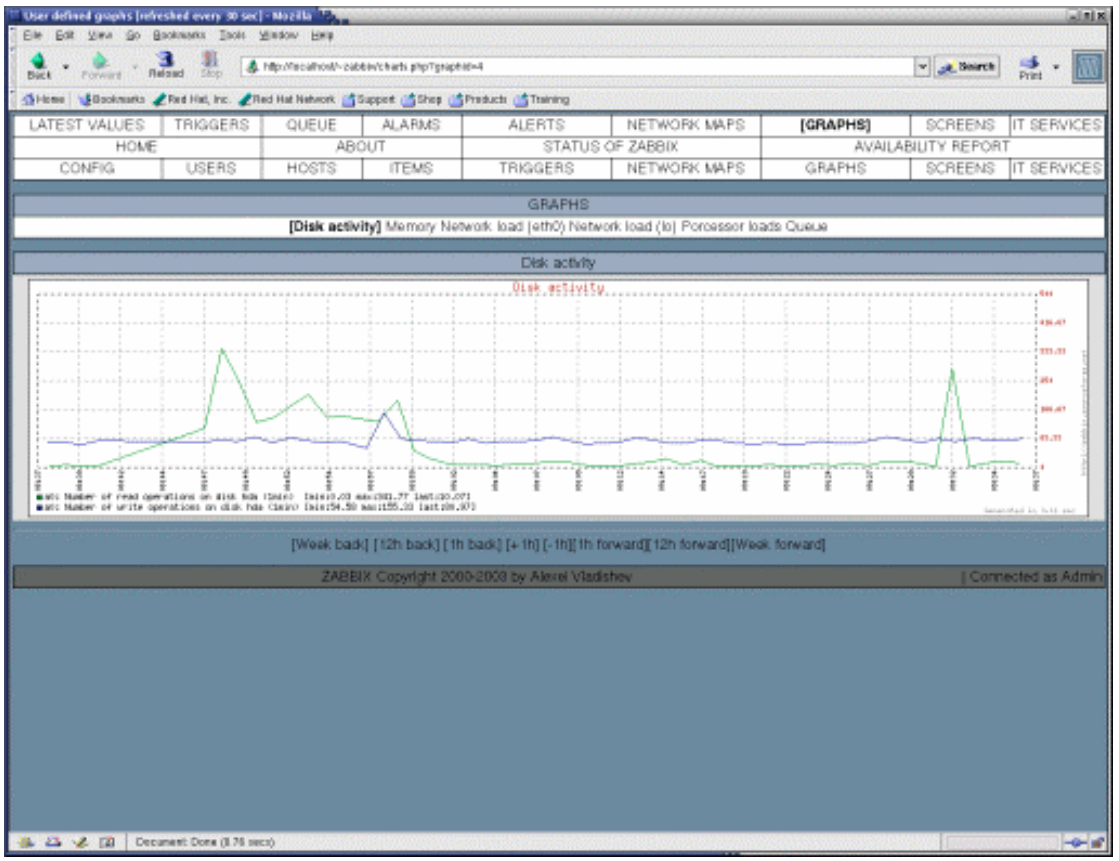
- **black** line. No problems or this connector is not linked to a trigger.
- **red** line. Corresponding trigger in TRUE state.

Possible actions:

Action	Result
Click on a map name	The map will be displayed
Click on a map name (in map header)	The map will be displayed in full screen mode.
Click on host icon	Status of Triggers screen will be displayed with the host selected

GRAPHS

The screen shows list of user-defined graphs.



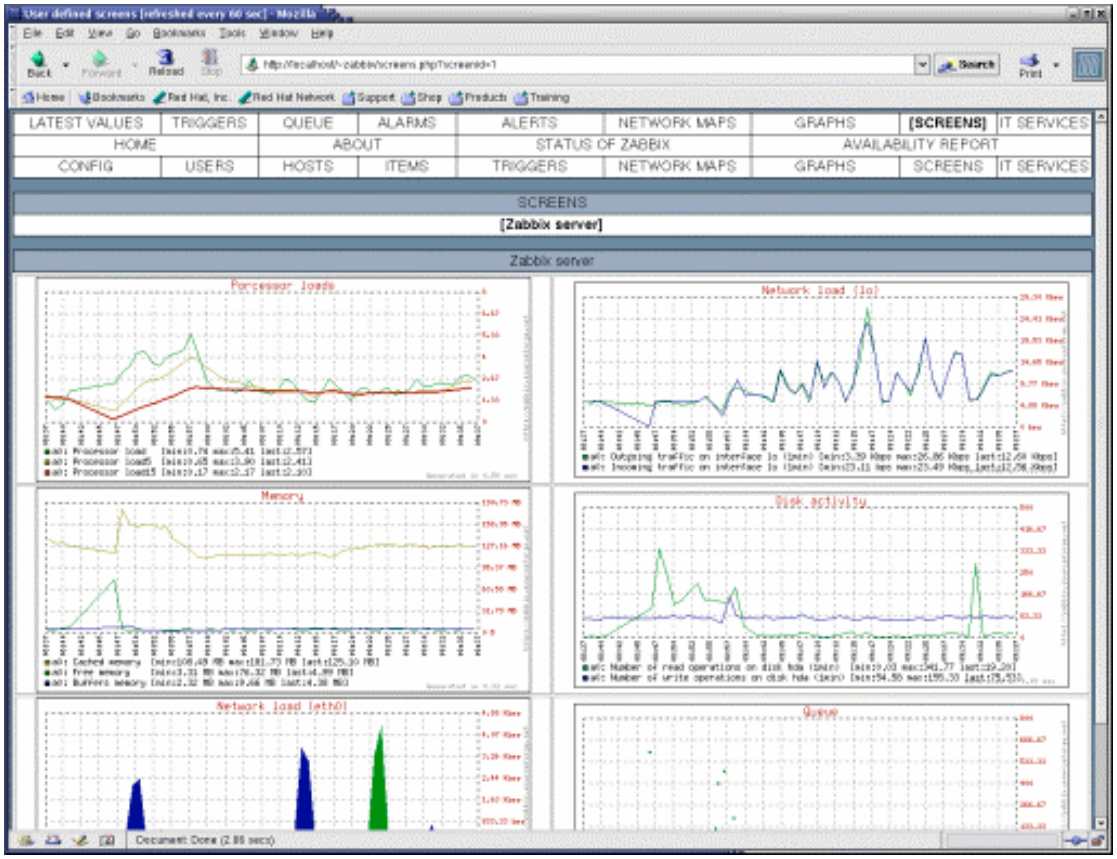
Note: Width of the graph is automatically resized to fit screen size.

Possible actions:

Action	Result
Click on a graph name	The graph will be displayed
Click on a graph name (in graph header)	The graph will be displayed in full screen mode.

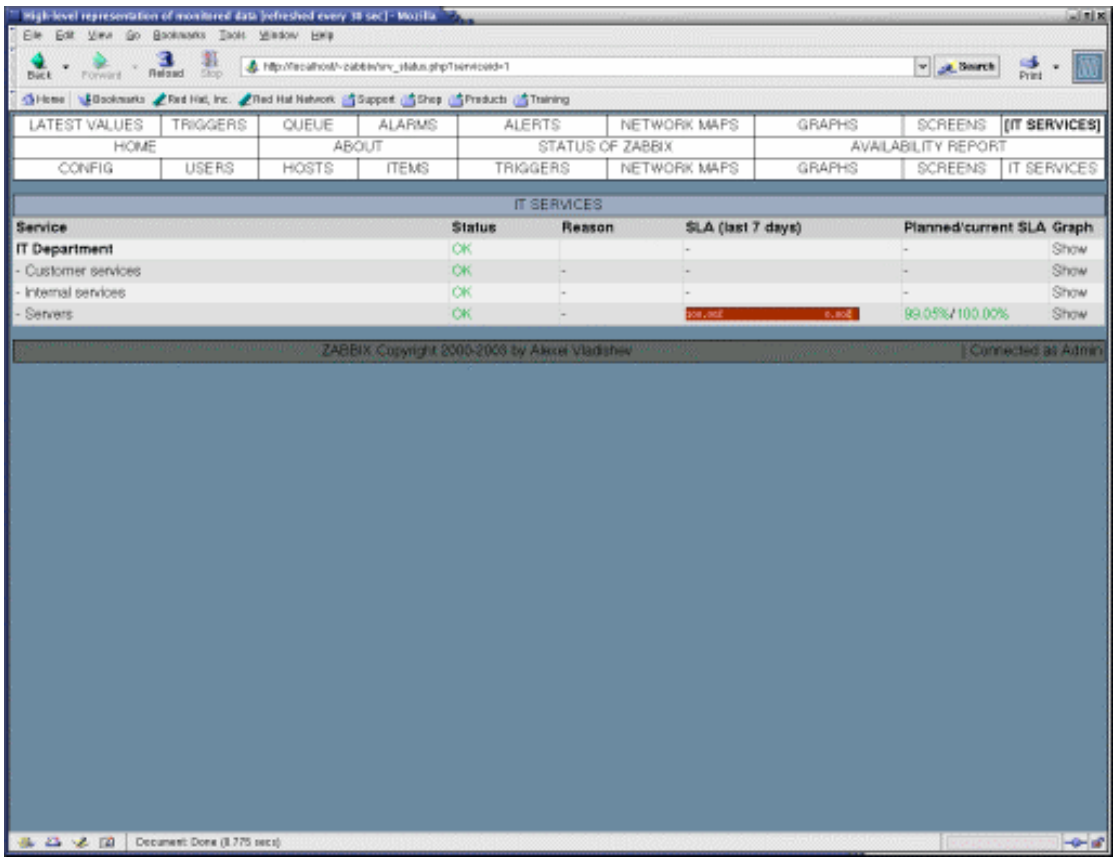
SCREENS

The screen shows user screens.



IT SERVICES

The screen shows high-level tree. Status of each service is displayed.



18 ZABBIX utilities

18.1 Start-up scripts

The scripts are used to automatically start/stop ZABBIX processes during system's start-up/shutdown.

The scripts are located under directory *misc/init.d.*

18.2 snmptrap.sh

The script is used to receive SNMP traps. The script must be used in combination with **snmptrapd**, which is part of package net-snmp.

Configuration guide:

1. Install **snmptrapd** (part of net-snmp or ucd-snmp)

2. Edit *snmptrapd.conf*. Add this line:

traphandle default /bin/bash /home/zabbix/bin/snmptrap.sh

3. Copy *misc/snmptrap/snmptrap.sh* to *~zabbix/bin*

4. Edit *snmptrap.sh* to configure some basic parameters

5. Add special host and trapper (type "string") item to ZABBIX. See *snmptrap.sh* for the item's key.

6. Run **snmptrapd**

19 System resources consumed by ZABBIX

The table contains list of system resources used by ZABBIX processes.

Resource	Type	Used by	Details
/etc/zabbix/zabbix_agentd.conf	File	zabbix_agentd	Configuration file
/etc/zabbix/zabbix_suckerd.conf	File	zabbix_suckerd	Configuration file
/etc/zabbix/zabbix_trapperd.conf	File	zabbix_trapperd	Configuration file
CPU	CPU	All ZABBIX processes and DB engine	Database engines is main consumer of CPU.

[to be finished]

20 Securing ZABBIX

[to be finished]

21 Database structure

The section describes structure of tables of ZABBIX database.

ACTIONS

The table contains list of actions to be applied if a trigger changes its state.

Column name	Type	Description
actionid	int	Unique action ID
triggerid	int	Trigger ID
scope	int	Scope of the actions: 0 – this trigger only 1 – hosts of the trigger 2 – all hosts
severity	int	Apply action if and only if trigger severity is equal or more than this value. Works for scope equal to 1 or 2.
userid	int	User ID
good	int	
subject	varchar	Subject of the message
message	varchar	The message itself
nextcheck	int	Time when the message should be sent next time
scope	int	Scope of this actions: 0 – this trigger 1 – all triggers of this host(s) 2 – all triggers of all hosts
severity	int	Perform action if severity of the trigger more or equal to this value. Ignored if 'scope'=0.

ALARMS

This table contains history of changes of trigger states. When a trigger changes its state, new record is added to the table.

Column name	Type	Description
triggerid	int	Trigger ID
clock	int	Time when trigger changed its status
value	int	Value of the alarm: 0 – trigger became FALSE 1 – trigger became TRUE 2 – trigger became UNKNOWN

ALERTS

This table contains history of alerts sent to users.

Column name	Type	Description
alertid	int	Unique alert ID
actionid	int	Action ID that generated this alert
mediatypeid	int	ID of media type
clock	int	Time when this alert was generated
type	varchar	Time of alert: EMAIL – email based alert
sendto	varchar	Recipient(s). In case of EMAIL, this specifies email address.
subject	varchar	In case of email, specifies subject of the email.
message	blob	Message
status	int	o – not sent 1 – successfully sent
retries	int	Number of retries. zabbix_suckerd will not send message if the value is more that 2. Increased after unsuccessful try.

CONFIG

The table contains global configuration parameters.

Column name	Type	Description
alarm_history	int	ZABBIX will delete records in table alarms older than this value (in days)
alert_history	int	ZABBIX will delete record in table alerts older than this value (in days)

FUNCTIONS

The table contains list of simple functions used in trigger expressions.

Column name	Type	Description
functionid	int	Unique function ID
itemid	int	Item ID
triggerid	int	Trigger ID
lastvalue	double	Last (most recent) value
function	varchar	Function name: LAST, MIN, MAX, PREV, DIFF, STR, AVG, COUNT,SUM, DELTA, CHANGE, ABSCHANGE
parameter	int	Parameter to the function. Ignored if function does not need any parameters

GRAPHS

The table contains list of user-defined graphs (charts).

Column name	Type	Description
graphid	int	Unique graph ID
name	varchar	Name of the graph
width	int	Graph width
height	int	Graph height

GRAPHS_ITEMS

The table contains list of monitored items belonging to graph.

Column name	Type	Description
gitemid	int	Unique ID
graphid	int	Graph ID
itemid	int	Item ID
sort_order	int	Sort order (0-100). 0 – draw first, 100 – draw last.
drawtype	int	Draw type: 0 – Line 1 – Filled 2 – Bold line 3 - Dot
color	int	Color

GROUPS

The table contains list of host groups.

Column name	Type	Description
groupid	int	Unique ID
name	varchar	Name of the group

HISTORY

This table contains history of integer values of items.

Column name	Type	Description
itemid	int	Item ID
clock	int	Timestamp
value	double	Value of the item

HISTORY_STR

This table contains history of string values of items.

Column name	Type	Description
itemid	int	Item ID
clock	int	Timestamp
value	varchar	Value of the item

HOSTS

The table contains list of monitored Hosts.

Column name	Type	Description
hostid	int	Unique host ID
host	varchar	Host name or IP address
status	int	Host status: 0 – monitored 1 – not monitored 2 – unreachable 3 – template
disable_until	int	In case if status is *unreachable*, do not check host and its items until this time

HOSTS_GROUPS

The table is used to define linkage between hosts and host groups.

Column name	Type	Description
hostid	int	Host ID
groupid	int	Group ID

ITEMS

This table contains definitions of monitored items.

Column name	Type	Description
itemid	int	Unique item ID
type	int	Type of the item: 0 – ZABBIX agent 1 – SNMPv1 2 – Trapper 3 – Simple check 4 – SNMPv2 5 - Internal
value_type	int	Type of received values: 0 – float 1 – string
snmp_community	varchar	Name of community for SNMP request. Example: public
snmp_oid	varchar	Object ID for SNMP request
hostid	int	Host ID
description	varchar	Description of the item
key_	varchar	Key to be sent to monitored host
delay	int	Update interval. Defines how often retrieve this item (in sec)
history	int	Do not store more than *history* days of history data
lastdelete	int	Time of last deletion from table *history*
nextcheck	int	Time when next value should be retrieved
lastvalue	double	Last value retrieved from host
lastclock	int	Time when last value was retrieved
prevvalue	double	Previous retrieved value
trapper_hosts	varchar	Comma-delimited list of IP addresses of hosts allowed sending data of the item. For item type *Trapper* only.
status	int	Status of the item: 0 – active 1 – disabled 3 – not supported by agent

units	varchar	Units of returned values. Can be empty. For example: bps, B
multiplier	int	Multiplier for units. If agent returns KB, then multiplier must be 1, i.e. 1024^1.

MEDIA

This table contains list of available medias for given user.

Column name	Type	Description
mediaid	int	Unique media ID
userid	int	User ID
mediatypeid	int	Media type ID
sendto	varchar	Address to send alert to
active	int	Status of the media: 0 – active 1 – disabled
severity	int	Bit: 0 – Not classified (1) 1 – Information (2) 2 – Warning (4) 3 – Average (8) 4 – High (16) 5 – Disaster (32)

MEDIA_TYPE

This table contains list of available medias for given user.

Column name	Type	Description
mediatypeid	int	Unique media type ID
type	int	Type: 0 – EMAIL 1 – external script
description	varchar	Description of the medi type
smtp_server	varchar	SMTP server. For 'type'=0 only.
smtp_helo	varchar	HELO value for SMTP server. For 'type'=0 only.
smtp_email	varchar	Email address of ZABBIX server. For 'type'=0 only.
exec_path	varchar	Name of an external script. Example: *lmt.sh*

PROFILES

This table contains user settings.

Column name	Type	Description
profileid	int	Unique media ID
userid	int	User ID
idx	varchar	Index for searching
value	varchar	Information for this index

RIGHTS

The table contains list of user permissions.

Column name	Type	Description
rightid	int	Unique ID
userid	int	User ID
name	varchar	Resource name
permission	char	Permission: R – read only U – read and write A – add H – restrict access (hide)
id	int	Resource ID

SCREENS

The table contains list of user-defined screens.

Column name	Type	Description
screenid	int	Unique screen ID
name	varchar	Screen name
cols	int	Number of screen columns
rows	int	Number of screen rows

SCREENS_ITEMS

The table contains configuration of screen cells.

Column name	Type	Description
screenitemid	int	Unique ID of the screen item
screenid	int	Screen ID
resource	int	Type of resource connected to the cell: 0 – graph 1 – simple graph 2 – map
resourceid	int	Resource ID
width	int	Width of the displayed resource. Ignored for Map.
height	int	Height of the displayed resource. Ignored for Map.
x	int	Coordinate X.
y	int	Coordinate Y.

SERVICES

This table contains list of defined IT services.

Column name	Type	Description
serviceid	int	Unique service ID
name	int	Description of the service
status	int	Status of the service
algorithm	int	Algorithm used to calculate status of the service: 0 – none 1 – MAX status of child services 2 – MIN status of child services
triggerid	int	Is the service linked to the trigger: NULL – it is not otherwise, trigger ID
sortorder	int	Defines sort order for IT Service form.
goodsla	float	Acceptable SLA (in %). Example: 99.5

SERVICES_ALARMS

This table is used to store history of state changes of IT Services.

Column name	Type	Description
servicealarmid	int	Unique ID
serviceid	int	ID of IT Service
clock	int	Timestamp
value	int	[explain ...]

SERVICES_LINKS

This table is used to define connections between different services to form IT Service tree.

Column name	Type	Description
linkid	int	Unique link ID
serviceupid	int	ID of higher level service
servicedownid	int	ID of lower level service
soft	int	0 – hard link 1 – soft link

SYSMAPS

The table contains list of defined network maps.

Column name	Type	Description
sysmapid	int	Unique network map ID
name	varchar	Name of the network map
width	int	Width of the map
height	int	Height of the map

SYSMAPS_HOSTS

The table contains definition of host displayed on network map.

Column name	Type	Description
shostid	int	Unique ID
sysmapid	int	Network map ID
hostid	int	Host ID
label	varchar	Label displayed under host icon
x	int	X
y	int	Y

SYSMAPS_LINKS

The table contains list of connectors between hosts displayed on network map.

Column name	Type	Description
linkid	int	Unique link ID
sysmapid	int	Network map ID
shostid1	int	ID of first host
shostid2	int	ID of second host
triggerid	int	ID of trigger connected to this link. Can be NULL.

TRIGGERS

The table contains list of triggers.

Column name	Type	Description
triggerid	int	Unique trigger ID
expression	varchar	Trigger's expression
url	varchar	URL
description	varchar	Description of the trigger
status	int	Status of the trigger: 0 – enabled 1 – disabled
value	int	Value of the trigger: 0 – FALSE 1 – TRUE 2 – UNKNOWN
priority	int	Severity of the trigger: 0 – not classified 1 – just for information ...
lastchange	int	Time when trigger value was changed
dep_level	int	Dependency level. The number shows how many triggers depend on this trigger.

SESSIONS

The table contains list of user sessions.

Column name	Type	Description
sessionid	varchar	User ID
userid	int	User ID
lastaccess	int	Last time the session was used

TRIGGER_DEPENDS

The table contains list of trigger dependencies.

Column name	Type	Description
triggerid_down	int	Trigger ID
triggerid_up	int	Trigger ID. This trigger depends of triggerid_down.

USERS

The table contains list of ZABBIX's users.

Column name	Type	Description
userid	int	Unique user ID
name	varchar	Name of the user
surname	varchar	Surname of the user
alias	varchar	Short name of the user
passwd	varchar	MD5 hash of user password

22 Extending ZABBIX

API (sender, agent, etc)

23 Problems and Common Errors

First of all, trace file (syslog or FileLog) must be checked for any information that may be helpful.
If problem is not listed here, check list of known bugs at http://zabbix.sourcefoge.net. There is a great chance that the issue you are facing is known and already resolved.

23.1 Authentication failed in case if user name and password is correct

During initial installation, no *data.sql* loaded into database.

Check system date on both ZABBIX server and client machine where WEB browser is running. The difference should not be more than 20 minutes. This is fixed in ZABBIX 1.0beta9 and newer versions.

Browser should be configured to accept cookies. ZABBIX uses cookies to store session-related information.

23.2 Undefined references to compress and uncompress

/usr/lib/libmysqlclient.a(my_compress.o): In function 'my_uncompress':

my_compress.o(.text+0xa1): undefined reference to 'uncompress'

/usr/lib/libmysqlclient.a(my_compress.o): In function 'my_compress_alloc':

my_compress.o(.text+0x138): undefined reference to 'compress'

These messages indicate that *./configure* is unable to find **libz** installed. The problem should only appear if flag *enable-static* is specified. Install package **libz-dev** package to get rid of the problem.

23.3 zabbix_suckerd: error while loading shared libraries: cannot open shared object file: No such file or directory

OS was unable to find required library. Connect as 'root'. Add path of MySQL library to */etc/ld.so.conf*. Run '*ldconfig*'.

24 ZABBIX in the future

The section contains incomplete list of improvements planned to be included into future releases of ZABBIX.

24.1 This manual TODO

Add “To be finished...” to all unfinished sections

24.2 TODO (ZABBIX version 1.1)

- True distributed monitoring
- Data encryption (SSL)
- Centralised monitoring of log files
- ZABBIX to periodically generate and send by email set of standard reports in PDF/HTML/PS format
- Graphs to be included into alert’s email
- support for IT Helpdesk
- more reports
- all ZABBIX errors and warnings to be numbered and documented and explained
- better support for SNMP (send and receive SNMP traps)
- same time navigator for all graphs
- change trigger status to UNKNOWN when status of a host is changed
- network map. Connect an icon to a trigger/service
- create one universal *chart.php* to be used for graphs, charts, and trends
- add MRTG-like graphs
- add support for *check_service[ldap]*
- improve functionality of IT Services
- dlopen/dlsym for agents
- **zabbix_suckerd** to send ICMP pings
- support of WIN32 as ZABBIX server platform

25 Support of ZABBIX

25.1 Free support

Free support is available using excellent Sourceforge's capabilities. Support is provided by Alexei Vladishev. Email address of Alexei Vladishev, author and main developer of ZABBIX, is alex@gobbo.caves.lv. Alexei can also be reached by calling +371 9275351. Calls will be accepted from 20:00 till 23:00 London time. Please use phone support in case of urgency or major problems only. Alexei speaks English, Russian and Latvian.

The table contains list of resources that might be helpful if you look for free support. When asking for help, provide the following information:

- version of ZABBIX
- DB engine
- platform (OS version)

Resource	Description
Mailing lists	Three mailing lists exist: - zabbix-announce - zabbix-development - zabbix-users Location: http://sourceforge.net/mail/?group_id=23494
Discussion forums	The following WEB forums exist: - Help - Open - Developers Location: http://sourceforge.net/forum/?group_id=23494
Feature requests	WEB page that contains list of feature requests raised by users of ZABBIX. Location: http://sourceforge.net/tracker/?atid=378686&group_id=23494&func=browse
Bug reports	WEB page that contains list of known bugs. Location: [insert location]

25.2 Commercial support

Commercial support is currently available to Baltic States (Estonia, Latvia and Lithuania) only. Hopefully, other countries will be included to the list as soon as ZABBIX 1.0 (stable) is ready.

If support is an issue for you, several options are available as well. Please, contact alex@gobbo.caves.lv to get additional information on available commercial support levels.

Benefits of commercial support may include:

- guaranteed response time
- high priority email and phone support
- direct contact to ZABBIX developers
- customer-specific development made by ZABBIX developers
- training sessions
- turn-key monitoring solutions
- feature requests are considered top priority
- printed ZABBIX manual plus installation on CD (pre-compiled distribution for supported platforms)
- subscription to updates and fixes
- installation and configuration services
- per-incident support
- pre-compiled binaries for all supported platforms

26 Credits

ZABBIX team wants to thank the guys from *http://sourceforge.net* for providing hosting for the project. Our team also wants to thank all the ZABBIX users who have sent corrections and suggestions. This sort of feedback helps us make the software better.

26.1 Developers of ZABBIX

ALEXEI VLADISHEV

Has written most of ZABBIX code including PHP front-end.

26.2 Contributors to ZABBIX

In alphabetical order:

ALEXANDER KALIMULIN

Help with various issues related to C, C functions, etc

ALEXANDER KIRHENSTEIN

Suggested fixes to make ZABBIX work under SCO.

ARTURS ABOLTINS

Patch to allow connection to MySQL using UNIX socket. Support for graceful shutdown in case MySQL server goes down (not implemented yet). Idea and initial code for ZABBIX screens.

CHARLIE COLLINS

Start-up scripts. Significant improvements of the Manual. Thanks Charlie!

DENIS USTIMENKO

Support for querying SNMP parameters by IP address.

DANIEL ESTER

Support for SNMP values of type timetick.

DANIEL HIGGINS

Improvements for email sending routines. Other changes.

ERIK CARLSEEN

Many excellent ideas.

EUGENY BACULA

Many suggestions for improvements.

HARALD HOLZER

RPMs and zabbix.spec.

IGOR MICKO

Plenty of interesting ideas based on real use of ZABBIX in large monitoring environment.

JAEN-BAPTISTE MARIOTTE

Help with testing

JEFF REDDING

Support for non-GCC compilers

JOHN CRUNK

Start-up scripts for RedHat 8.0

JOSH KONKOL

Help with testing

JÜRGEN SCHMITZ

Idea and implementation of check_service_perf[*]

KASPARS CIKMACS

Lots of new ideas based on real experience of using ZABBIX.

LAURIS STIGLICS

Select criteria in for “Status of Triggers”

LUKAS MACURA

Many ideas.

MARC LEDENT

Original implementation of proc_cnt[*] for Solaris.

MARIUSZ ...

Support for system[procload] on Solaris 2.6. Improvements for graphs. Improvements for system maps.

MICHAL SUSZYCKI

Help with autoconf and automake issues.

MIKE HOOLEHAN

Help with making the ZABBIX Manual correct and understandable.

OLIVER SIEGMAR

Fixes in SQL statements of WEB frontend.

RICKARD PLARS

Help with fixing coredump for zabbix_suckerd.

SEBASTIEN “SLIX” LIENARD

Fixed selection of hosts and icons in sysmap.php. Other fixes.

SHAWN MARRIOTT

Proofreading of the Manual.

VICTOR KIRHENSTEIN

Native ZABBIX agent for WIN32 platforms.

27 Glossary of terms

Term	Description
GNU	
SNMP	Simple Network Management Protocol
URL	
ROI	Return On Investment
PHP	Server-side scripting language for creating dynamic Web pages.
GD	
PNG	
UNIX	
WEB	
SLA	

28 GNU General Public License

GNU GENERAL PUBLIC LICENSE
Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps: (1) copyright the software, and (2) offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

GNU GENERAL PUBLIC LICENSE
TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a "work based on the Program" means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term "modification".) Each licensee is addressed as "you".

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

1. You may copy and distribute verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b) You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c) If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: if the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may copy and distribute the Program (or a work based on it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a) Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b) Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c) Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

4. You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

5. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

6. Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

7. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

8. If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

9. The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

10. If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY

11. BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

12. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 59 Temple Place, Suite 30, Boston, MA 02111-1307 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than `show w' and `show c'; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program `Gnomovision' (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989
Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

29 References

MySQL Reference Manual for version 3.23.39

PostgreSQL User's Guide

PHP Manual

The ZABBIX home page at http://zabbix.sourceforge.net

timed

hdparm

MySQL home page at http://www.mysql.com

PostgreSQL home page at http://www.postgresql.org

30 Appendix

30.1 Warnings and Errors

[to be finished]