Bitrix Site Manager

Recommendations on configuring web systems for proper operation with Bitrix Site Manager

Contents

Introduction

This document is intended for use by technicians and system administrators.

The current document contains recommendations on how to customize the server software settings. These operations are executed by the Bitrix experts for web projects with more than 10 thousand unique site visitors per day, or for the ones with insufficient system resources, in order to arrange the optimized load.

In this document, it is assumed that the web-project developer owns the basic knowledge in server software, databases configuring and PHP programming language.

All the recommendations listed below are mostly related to **UNIX** or **Windows** systems using the **Apache** web server.

These recommendations are just examples of how to properly customize the system. They are supposed to help you refine the instructions according to the available resources, equipment and servers in configuration.

The document has been elaborated thanks to CifNet USA (www.cifnet.com) and personally to Eugene Kruglov. The CifNet administers the majority of dedicated servers on which the *Bitrix Site Manager* software is installed.

⚠ ***Attention!*** *The recommendations mentioned in this document work best in case all the mandatory requirements are fulfilled, and the recommended settings are specified on the page Settings ->Tools -> Site checker in the administrative section of the product.*

The standard settings of the server software are designed for standard equipment and applications. Modifying several server configurations allows:

- significantly increase the overall performance of the system;
- reduce the time required for page generation;
- improve the system's tolerance to peak loads.

Configuring the Apache web server

The **Apache** web server is the key application that processes user requests and runs the PHP logic of the product.

Being loaded with PHP and other modules, the web server consumes a considerable part of both shared memory and memory for a separate process.

The optimization strategy is to reduce the amount of memory consumed, the number of simultaneously running web server processes, the number of process restarts and duration of process lifecycles.

Reduction of memory consumed by the web server

It's desirable to exclude all the non-used modules from the processes of compiling and web server loading. Such an approach allows reducing the amount of memory consumed by the web server at start and for each separate process.

It's important that the PHP should operate with the web server as a module (**mod_php4**) and be loaded by the web server at start.

Today the **Apache** 1.3.XX web server is usually used as it has proved to operate most efficiently. Sometimes, **Apache** 2.XX is also used, but this version occasionally causes performance and PHP multithreading problems.

Creation of two-level configuration: Front-end plus Back-end

In the standard configuration, the web server processes all requests to executable pages, graphic files, binary files, cascading style sheets and other components of the site. Considering that the web server consumes quite a lot of memory, one of the top tasks is to **minimize the number of requests processed by the web server**. A perfect situation is as follows: in 99% of cases, the web server processes only requests to PHP pages. All the other requests shouldn't be processed by the memory-consuming web server.

Another problem of processing requests is that site visitors often use relatively **slow communication channels** compared to the ones of the web server. For example, the time required to generate a page can be 0,1 second, while the time required to download this page, even compressed, to the client can take 5 to 50 seconds or even more. Unfortunately, during the whole process of downloading the page the server will hold in its memory an idle process waiting for the completion of the download procedure; it cannot free up this memory until the page is completely downloaded on the client's computer.

The best solution to the above problems can be the creation of the two-level system of the **Front-end+Back-end** type for request processing.

All requests to the server and all its pages are received by a special proxy process (**Front-end**). As a proxy server, you can use **SQUID** (http://www.squid-cache.org/), **OOPS** (http://www.oops-cache.org/), **NGINX**, **mod_accel** (http://sysoev.ru/en/) or any other product with a similar functionality.

Front-end processes all the requests that can be processed independently. Graphic files and style sheets can be requested from the web server only in accordance with the cache policies. After that, the files are stored in the cache memory of the proxy server and are transferred to users without referring to the **Apache** server. It's recommended to cache the graphic files every

3-5 days. Here's an example of customizing cache in the **.htaccess** file located in the root of the web server:

```
ExpiresActive on

ExpiresByType image/jpeg "access plus 3 day"

ExpiresByType image/gif "access plus 3 day"
```

⚠ ***Attention!*** *The above example requires that the web server should allow re-defining variables in the* ***.htaccess*** *file, and that the* ***mod_expires*** *module should be installed.*

In some cases, the **Front-end** cache policies are defined regardless of the **Back-end** settings.

So, the **Front-end** will cache all the images. Requests to content pages will not be cached but directed to the **Back-end** part of the system. The **Back-end** is a standard **Apache** web server operating with the non-standard port, 88, for example, and processing only those requests that come from the **localhost** or IP address of the proxy server.

⚠ ***Note for administrators!*** *It's recommended to use several internal IP addresses of the type 127.0.0.2, 127.0.0.3, etc. with the 80 port; otherwise, some undesirable re-directs to the non-operating Front-end port may occur.*

Let's see how a user request to a typical site page is processed.

The request is received by the proxy server at the address http://www.bitrixsoft.com on the 80 port, and re-directed to the **Apache** web server at the address http://127.0.0.2:80/. The request is executed by the **Apache** web server. The response is received by the proxy server, and connection between the proxy server and **Apache** is closed thus freeing up memory. The proxy server transfers the requested page to the site visitor by using a slow communication channel. Having received the page, the user's browser sends successive requests for graphic elements and style sheets. All the requests are received by the proxy server and processed without referring to the **Back-end**.

Practice shows that such a configuration allows to significantly offload the computer, reduce the amount of memory consumed, speed up request processing and allocate more memory for the database functioning. Use this configuration to unload the server while processing a large number of files such as music files, software packages, presentations and other objects.

Additional recommendations for the two-level configuration

► For proper operation of the statistics module, it's required to pass a real IP address from the **Front-end** to the **Back-end**. In particular, **SQUID** provides for the declaration of the variable *HTTP_X_FORWARDED_FOR*. To replace a variable in the product, it's required to insert the code that follows into the file **/bitrix/php_interface/dbconn.php**:

```
if(strlen($_SERVER["HTTP_X_FORWARDED_FOR"])>0 &&
$_SERVER["REMOTE_ADDR"]=="127.0.0.1")
{

if($p = strrpos($_SERVER["HTTP_X_FORWARDED_FOR"], ","))
     {
          $_SERVER["REMOTE_ADDR"] = $REMOTE_ADDR =
trim(substr($_SERVER["HTTP_X_FORWARDED_FOR"], $p+1));
          $_SERVER["HTTP_X_FORWARDED_FOR"] =
substr($_SERVER["HTTP_X_FORWARDED_FOR"], 0, $p);
     }
else

     $_SERVER["REMOTE_ADDR"]=$REMOTE_ADDR=$_SERVER["HTTP_X_FORWARDE
D_FOR"];
}
```

► In the **Apache** configuration on the **Back-end**, it's recommended to disable **KeepAlive**. Since the **Front-end** is located on this very computer or close to it, it's much more preferable to free up the resources in a quicker way;

► On the **Apache** web server, you can enable and configure the module **mod_deflate** (http://sysoev.ru/en/) by simultaneously disabling the compression module in the product. It allows to quicken page generation and processor consumption at peak loads;

To minimize the number of loaded processes on the **Back-end**, it's recommended to specify the following range of values of the *MaxClients* parameters in the **Apache** settings: 5-50. The exact value should be defined according to the available system resources and loads.

⚠ ***Example!*** *With the proper two-level configuration, 20 simultaneous processes are able to process about 100 thousand unique users per day or 1 000 000 pages.*

The feature limiting the number of processes allows to strictly confine memory consumption thus excluding computer failures in case of attacks and stresses. For the **Oracle** configuration, this feature allows to enable the **Persistent connection** to the database and reduce connection expenses and the number of **Oracle** processes.

It's desirable to adjust the **Back-end** web server parameters so that the number of processes corresponds to your load and the number of clients. That will allow reducing inefficient re-creation of processes and disk operations. For example:

```
MinSpareServers 20

StartServers 20

MaxClients 20
```

Configuring PHP

To **minimize the amount of memory** consumed by the processes running on the web server on which PHP is launched, it's recommended to exclude all the non-used modules from the compilation and dynamic load processes.

To exclude the phase of PHP code compilation, reduce the load on processors and the time required for page generation, it's recommended to use **PHP code pre-compilers**. There are commercial and free products of the kind:

- Zend Performance Suite - http://www.zend.com/
- eAccelerator (FREE) - http://eaccelerator.sourceforge.net/HomeUk
- Turck MMCache (FREE) - http://turck-mmcache.sourceforge.net/
- PHP Accelerator - http://www.php-accelerator.co.uk/
- Alternative PHP Cache - http://apc.communityconnect.com/
- AfterBurner Cache - http://www.bwcache.bware.it/

The most stable and multifunctional product is the one supplied by **Zend**, a **PHP developer**. Another product, **eAccelerator**, is also widely used; it's free.

Using one of the above products at peak loads on the site allows for considerable reduction of the load on the processor by excluding the compilation phase.

To speed up operation with PHP, **it's recommended to save** the session files to a special directory which is a virtual disk **in memory**, or use *session.save_handler=mm* в **php.ini**

*Switch from file based sessions to shared memory sessions. Compile PHP with the --with-mm option and set session.save_handler=mm in **php.ini**. Informal benchmarks suggest that session management time is halved by this simple change. Added 1 Dec 2001. This hint should only be used for PHP 4.2.0 and above as there were bugs before this.*

http://us4.php.net/ref.session

If possible, it's recommended to use the system "RAM disk":

⚠ ***Note:*** *Optionally you can use shared memory allocation (mm), developed by Ralf S. Engelschall, for session storage. You have to download mm and install it. This option is not available for Windows platforms. Note that the session storage module for mm does not guarantee that concurrent accesses to the same session are properly locked. It might be more appropriate to use a shared memory based file system (such as tmpfs on Solaris/Linux, or /dev/md on BSD) to store sessions in files, because they are properly locked.*

Customizing the MySQL database

Optimizing operation with the database for the **MySQL** version of the product is one of the key strategies of optimizing the whole system, since the product interacts with the database very closely.

Optimization here means reducing the number of disk operations while working with the database and enable parallel processing for the most appropriate usage of computer resources.

MySQL typically operates with the **MyISAM** data format. A plain data format stores each datasheet or index in a separate file. On the whole, on sites with medium load this format is the quickest one; however, it doesn't guarantee data integrity and security as it implies no transactions. The major drawback of **MyISAM**, as far as performance is concerned, is locking on the table level while performing various operations. As a result, with peak loads of **MySQL** and, in particular, **MyISAM** tables become a bottleneck in the system being an obstacle for increasing the computer performance and the number of requests to be processed. It also leads to increasing the time required for the page loading due to waiting for the tables on the **MySQL** level.

It's recommended to save all tables used in the project in the data format **InnoDB** (http://dev.mysql.com/doc/mysql/en/innodb.html). The **InnoDB** format, starting with the **MySQL** 4.0 version, is included into the standard product package and guarantees reliable data storage, transactionality and data locking on the string level.

Do the following to change the table type to **InnoDB**:

► On the administrative menu *Settings -> Tools -> SQL query*, execute the following command:

```
SHOW TABLES
```

As a result, you'll receive a list of all the current tables of the product.

► For each table, execute the following command:

```
ALTER TABLE <TABLE NAME>, type=InnoDB
```

In the FAQ section of the site www.bitrixsoft.ru, you can find an example of creating a script for converting tables into the **InnoDB** format.

► Upon converting all the tables of your database into the **InnoDB** format, add the following code into the file **/bitrix/php_interface/dbconn.php**:

```
define("MYSQL_TABLE_TYPE", "InnoDB")
```

The format conversion allows avoiding the bottleneck in the performance while working with the database, and using the system resources to the full extent.

⚠ ***Attention!*** *For better database performance while working with* ***InnoDB****, it's recommended to customize* ***my.cnf*** *for* ***MySQL*** *in the section of* ***InnoDB*** *parameters* http://dev.mysql.com/doc/mysql/en/innodb-configuration.html

Pay attention to the following parameters:

```
set-variable    = innodb_buffer_pool_size=250M

set-variable    = innodb_additional_mem_pool_size=50M

set-variable    = innodb_file_io_threads=8

set-variable    = innodb_lock_wait_timeout=50

set-variable    = innodb_log_buffer_size=8M
```

If **MyISAM** is not actively used, you can free up memory for the **InnoDB** parameters.

It's desirable that the data cache should include the major data amount used by the product. For the proper database functioning, about 60-80% of free memory are usually allocated in the system.

It's also recommended to perform **MySQL** multithreading.

⚠ ***Note for administrators.*** *Converting data to* ***InnoDB*** *may considerably slow down some large-scale read and update operations. The reason is that all operations that involve data changing are executed by using transactions. So, you should be careful while converting your project data into the* ***InnoDB*** *data format.*

⚠ ***Attention!*** *If you've decided to continue working with the* ***MyISAM*** *data type, it's mandatory to configure* ***MySQL*** *to increase the amount of cacheed information, sorting areas and minimize the number of disk operations. Using 60-80% of RAM for the database may significantly accelerate operation of a typical project.*

An important parameter affecting memory consumption by **MySQL** is the maximum number of synchronous connections *max_connections*. By using the two-level architecture of the **Front-end/Back-end** type, you can considerably reduce the number of simultaneous connections and free up memory intended for sorting data in memory and spooling. If the number of *MaxClients* is specified in the settings of the **Back-end** web server, you may consider that the maximum number of running processes to the database will approximately correspond to the maximum number of simultaneous connections. It's recommended to adjust the *max_connections* parameter so that you can have 15-30% reserved free connections of the maximum value.

Customizing the Oracle database

The **Oracle** version of the product *Bitrix Site Manager* has been tested and operates with **Oracle** 9i and 10g.

Using **Oracle** as the database allows for significant increase in the system reliability and performance at peak loads. The **Oracle** architecture provides for the usage of the server resources to the full extent and perfect interaction of applications while operating with large data amount, generating extensive reports or a large number of simultaneous connections. **Oracle** is also good for its ability to work with scalable Internet projects.

General recommendations on customizing **Oracle** are almost the same as the **Oracle** recommendations on configuring the system in order to reduce read and sorting disk operations.

Be very attentive while specifying the values of system variables for memory management. It's recommended to use up to 60-80% of RAM for data cacheing thanks to managing the following variables:

- db_cache_size
- shared_pool_size
- pga_aggregate_target

To lower expenses on the repeated analysis of SQL requests, it's recommended to define the following values:

```
cursor_space_for_time=TRUE

cursor_sharing=SIMILAR

star_transformation_enabled=TRUE
```

If **Oracle** and the web server are installed on the same machine, it's recommended to use the **IPC protocol** (*PROTOCOL = IPC*) and (*KEY = EXTPROC*) to connect to the database – it allows to avoid working with the IP protocol suite.

If the two-level architecture (**Front-end** and **Back-end**) of request processing is implemented, and the *MaxClients* is defined, it's recommended to insert into the file **/bitrix/php_interface/dbconn.php** the code that follows:

```
define("DBPersistent",true);
```

Conclusions

Testing the product at peak loads verifies that by selecting the right approach towards resource allocation and configuration issues, one can increase the performance of the equipment in several times.

For additional information, contact the technical support department of the Bitrix company: www.bitrixsoft.com , section “Technical support”, or via e-mail admin@bitrixsoft.com .