

Redis Cache Release Notes: Aug 2013

Release Information

Original Launch: August 14, 2013

Requirements: Service Pack (SP) 11 or later and a [Redis server](#)

The Redis Cache Building Block enables Blackboard Learn's cache framework to use Redis as an additional type of memory store outside the heap. [Redis](#) is an open source, high-performance, networked advanced key-value store. Other companies that use Redis include Twitter, Craigslist, Github, Stackoverflow, and Instagram.

Supported Platforms

The Redis Cache Building Block is supported on the following platforms:

- Blackboard Learn 9.1 SP 11 (Build: 9.1.110082.0) with [Cumulative Patch \(CP\) 8](#) or later
- Blackboard Learn 9.1 SP 12 (Build: 9.1.120113.0) with [CP 2](#) or later
- Blackboard Learn 9.1 SP 13 (Build: 9.1.130093.0)

Performance, Scalability, and Availability Highlights

- Increases request throughput and response times
- No risk of performance loss
- Cache is shared across all application servers and the hottest subset of data is kept local
- Cache is available under JVM heap pressure
- Enables the system to scale with a manageable JVM heap size
- Bigger cache and faster overflow storage
- Fast and efficient object graph serialization
- Asynchronous puts and updates via write-behind strategy
- Increases off-heap caching opportunity
- Reduces on-heap cluster invalidation overhead
- Caches that survive application restarts and crashes
- Rich monitoring features for preventive maintenance and troubleshooting

Redis Cache Server Monitoring

Redis Cache Server monitoring is available when the Admin Console is installed. Look for "Redis Cache" in the main menu of the Admin Console.

Admin Console: r6x64o11-pv010.pd.local
Tools for advanced monitoring of the Learn system.

Monitors
Detailed view of supported monitors and their associated listeners.

Database
Detailed information about database connection pool usage and potential performance issues related to long running SQL queries.

System Captures
Collect logs, system metrics and thread data for offline analysis.

Threads
Navigate a list of the current threads in the Java virtual machine.

JMX Browser
Browse the JMX MBean hierarchy.

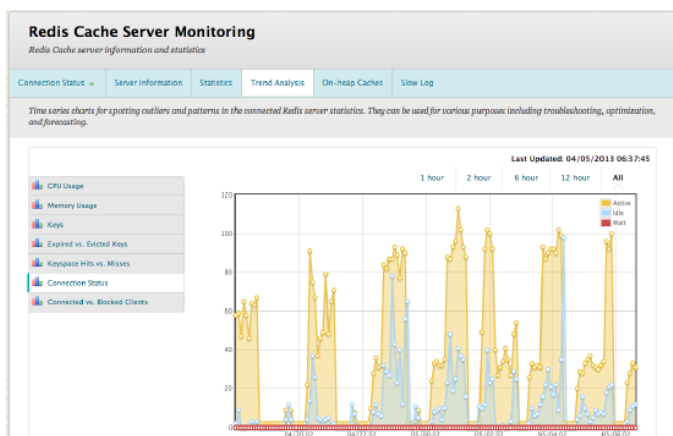
Caches
Detailed statistics related to in-memory system caches. This data facilitates better memory and performance tuning of the Learn system.

Memory
Memory pool details, including the memory usage of the system over time.

System Information
Detailed information about the operating system and Java virtual machine.

Logs
Monitor system logs live through the console.

Redis Cache
Monitor Redis Cache server status and statistics.



When Do You Need It?

The Redis Cache building block is recommended if one or more of these scenarios are observed:

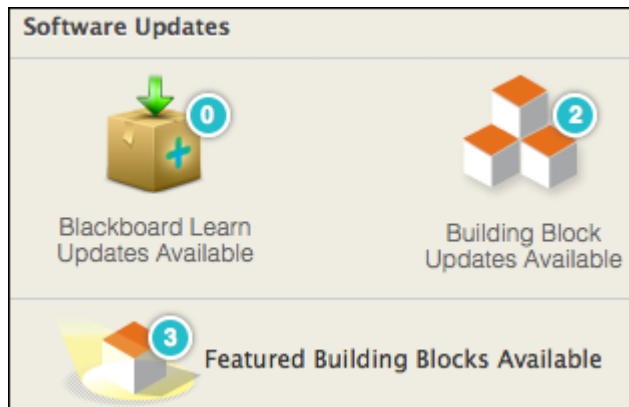
- High cache eviction rate
- System resource pressure on the database server
- Pressure on JVM heap utilization
- Horizontally large scale Blackboard Learn deployments to reduce cluster cache invalidation traffic

How to Install and Activate the Redis Cache Building Block

You can use Software Updates to install the Redis Cache building block directly from the System Admin panel. Alternatively, you can visit [Behind the Blackboard Downloads](#) and find the Redis Cache building block to download and install manually.

Software Updates

Software Updates are located on the System Admin panel. When a new building block is available, or has been updated, you are notified by a change in the icons. Simply click the **Feature Building Blocks Available** icon and locate the Redis Cache building block. Click **Install**. You will need to make the building block available after it has been installed.



Behind the Blackboard Downloads

Download the Redis Cache building block from [Behind the Blackboard Downloads](#) home, located under Feature Building Blocks for Learn. Save the file on a local drive. Unzip the package to access the `.war` file.

1. Navigate to the **System Admin** panel > **Building Blocks**.
2. Click **Installed Tools**.
3. Click **Upload Building Blocks** from the action bar.
4. Browse for the Redis Cache building block `.war` file.
5. Click **Submit**.
6. You will get a success message when the building block is installed. Set the building block to **Active** and click **Approve**.


How to Enable Redis Cache

1. Install a Redis server. For steps, see [How to Install Redis Server](#).
2. Navigate to the Redis Cache building block settings page on the system admin panel.
3. Select the list of caches to enable Redis.
4. Provide Redis connection settings in the **Host**, **Port**, and **Password** fields.

5. Click **Submit**.
6. Make sure that the Redis server port and the RMI ports are open among Redis server and the Learn application servers.
7. Restart all Learn application servers (rolling restart is recommended for production).
8. Navigate to Redis connection status page on the system admin panel.
9. Verify that all application servers are connected to Redis.

How to Install Redis Server

These steps assume that you are installing Redis on Red Hat Enterprise Linux 6.

1. Prepare a Linux server. A dedicated machine or VM instance is highly recommended. Do not run Redis on Windows or Solaris. Blackboard Learn on Windows and Solaris can use Redis on Linux.
 - a. Set the Linux kernel overcommit memory setting to 1 by adding `vm.overcommit_memory=1` to `/etc/sysctl.conf`. Reboot the server or run the command `sysctl vm.overcommit_memory=1` for the change to take effect. See [Redis FAQ](#) for the reason.
 *Redis is a single-threaded server. It favors fast CPUs with large caches instead of many cores. Intel CPUs are currently known to be the best fit. Redis documentation states that it is not uncommon to get only half the performance on an AMD Opteron CPU compared to similar Nehalem EP, Westmere EP, or Sandy bridge Intel CPUs.*
 - b. Redis throughput is limited by the network well before the CPU. Use fast and/or multiple NICs.
 - c. (Optional) Redis runs slower on a VM, so they recommend running Redis on a physical box.
2. Download and build the latest stable version of Redis.

```
wget http://redis.googlecode.com/files/redis-x.x.xx.tar.gz
tar xzf redis-x.x.xx.tar.gz
mv redis-x.x.xx /usr/local/redis
cd /usr/local/redis
make
```

3. Prepare the Redis server configuration file by copying `$REDIS_HOME/redis.conf` to `$REDIS_HOME/6379.conf`. Set the following properties:

```
daemonize yes # Use "no" when run under daemontools.
pidfile /var/run/redis/redis_6379.pid
port 6379
timeout 300 # Reclaim inactive connections (seconds).
tcp-keepalive 0 # Keep connections live from network tools
likefirewall.
loglevel notice # Keep it moderately verbose.
logfile /var/log/redis/redis_6379.log
databases 1
#save 900 1 # Disable snapshotting.
#save 300 10 # Disable snapshotting.
#save 60 10000 # Disable snapshotting.
requirepass xxxx # Use at least 16 characters with mix lowercase
and uppercase
# characters, numbers, and symbols. Try not to use words or
# phrases in your password.
maxclients 10000
maxmemory xxxx # Make sure Redis doesn't use swap (bytes).
maxmemory-policy volatile-lru # We use expire.
maxmemory-samples 3
appendonly no
slowlog-log-slower-than 50000 # Execution time threshold
(microseconds).
slowlog-max-len 128 # Length of the slow log.
```

4. Add a Redis home directory to `root`.

```
# Redis
export REDIS_HOME=/usr/local/redis
```

5. Create a Redis `user`.

```
useradd -d /home/redis -s /bin/sh redis
chown -R redis:redis $REDIS_HOME
chmod 700 $REDIS_HOME
```

6. Prepare the Redis init script.

```
cp utils/redis_init_script /etc/init.d/redis_6379
```

7. Configure the Redis init script.

```
# chkconfig: - 85 15
# description: Redis is a persistent key-value database
# processname: redis
REDISUSER="redis"
REDISPORT=6379
EXEC=/usr/local/redis/src/redis-server
CLIEXEC=/usr/local/redis/src/redis-cli
PIDFILE=/var/run/redis/redis_6379.pid
CONF="/usr/local/redis/6379.conf"
$EXEC $CONF ==change to==> /bin/su - $REDISUSER -c "$EXEC $CONF"
```

8. Activate the Redis service.

```
mkdir /var/run/redis /var/log/redis
chown redis:adm /var/run/redis /var/log/redis
sudo chmod 750 /var/log/redis
cd /etc/init.d
chkconfig --add redis_6379
```

9. Start the Redis server.

```
service redis start
```

Security

Hardening Redis Server

Two components secure the Redis server:

- Adding proper security configurations to the Redis application.

- Adding secure configurations to the server running Redis.

Adding secure Redis settings involves making sure that the Redis application is as secure internally as it can be. Secure configurations on the server are server settings and configurations that can be used to secure the server and the Redis application without involving the Redis application itself. The following sections describe two types of configurations.

Secure Redis Settings

The two important settings for the Redis server itself are:

- Using a secure Password for Redis.
- Blocking access to some Redis commands.

To learn more about Redis' recommended security configuration, see [Redis Security](#).

Redis Password

The Redis server is very efficient at serial string lookups, which is exactly what the Redis password lookup request is. Additionally, the Redis password is required only by the Blackboard application. Users do not need to remember it, so the password used should be much stronger than a traditional password. While normally you should follow the [owasp password length and complexity guidelines](#) for setting password length, because end users do not need to remember this password, the convention of choosing passwords based on users' ability to remember the password can be ignored.

Guideline	Reason
Use at least 16 characters, instead of 8	A brute force attack would take n to the 8th power times as long as an 8 character password, which is the normal minimum.
Mix lowercase and uppercase characters, numbers, and symbols	As with normal passwords, mixing case, numbers, and symbols reduces the likelihood of a successful dictionary attack and adds complexity for a brute force attack because all types of characters need to be checked.
Try not to use words or phrases in your password	Because this password does not have to be remembered, you can remove dictionary attacks entirely by not using any words or phrases.

The password setting itself must be added to the Redis configuration file, using the `requirepass` directive, such as:

```
requirepass ^a@$4J0|-|k!4P+f%
```

Blocking Redis Commands

Because Blackboard is using Redis just as a cache, a number of commands that are not being used. Some of these commands could be used by a malicious user to cause Denial of Service via cache misses, denial of service by running the server out of space, as well as information exposure by allowing malicious users to obtain the entire set of cache data. Redis provides an interface for altering or dropping these commands entirely, which Blackboard recommends for a few specific commands:

Key	Reason for Removal Recommendation
APPEND	Allows modification of keys to force cache misses
BGSAVE	Command that saves the dataset to disk, which could allow information exposure if a user obtains the file
RENAME	Renames a key, which would force cache misses
SAVE	Same as BGSAVE
SPOP	"Removes a random member from the set"
SREM	"Removes one or more members from the set"

To block these commands, edit the Redis configuration file to add the following:

```
rename-command APPEND ""  
rename-command BGSAVE ""  
rename-command RENAME ""  
rename-command SAVE ""  
rename-command SPOP ""  
rename-command SREM ""
```

By renaming the command to the empty string, you block anything from accessing it. Additionally, any new Redis commands that are created should also be blocked in this way because the building block will not be using them.

Secure Server configuration

Securing the server running Redis means taking steps to ensure that only users who need access to the Redis server can access it, as well as preventing unauthorized access to the Redis application through file and network permissions.

Run Redis without Root Context

The user root, on Linux machines, has full access to any file on the system, regardless of permissions or access control. This is why it is dangerous to use the root user to run applications because well crafted attacks could allow a malicious user to control the account running that application. Because Redis does not need that level of

access to the system, a user account with decreased permissions, just enough to run Redis, should be used to install and run the Redis application.

Blackboard recommends not using this account for anything except running the Redis server. It should not have any permissions to the operating system or other applications or daemons running on this server.

Secure the Redis Configuration File

The password used by the Redis server, which was described in an earlier section, is stored in cleartext within the configuration file. This is outside of Blackboard's control because this is how Redis functions at this time. So, you need to limit the users who have access to the file. It should be limited only to the user that runs the Redis application.

This can be accomplished using the following command, assuming that the redis configuration file is `redis.conf`:

```
chown 600 redis.conf
```

Block Unnecessary Traffic to Redis Server

The Redis server, assuming the only application running on it is Redis, should not be accessible to the open internet. It should be able to be reached only in the following circumstances:

- An administrator needs to do work on the Redis server
- Blackboard communications over the Redis port

This will prevent unauthorized users from attempting to bring down the Redis server, run unauthorized commands in Redis, and so on. Assuming that the default Redis port, 6379 is in use, add the following types of rules to the firewall for this server:

Rule	Explanation
Block all traffic to and from the server	This should be the default behavior, except in the case of the exceptions below.
Create an exception for loopback requests	The server should be able to access itself, if for no other reason than monitoring incoming requests to Redis using the <code>redis-cli commandmonitor</code> .
Create an exception for traffic from and to the Blackboard servers only over Redis port	The Blackboard server should be able to communicate with Redis over port 6379, and Blackboard should be able to get responses from the Redis server.
Create an exception allowing for remote administration from a specific machine	Administrators may need access to the server for the purposes of upgrading Redis, checking logs, monitoring performance, and so on. Unless administrators have direct access to the machine, they will need to be able to access the server

remotely. SSH traffic should be limited to a single machine, or a small set of machines, so normal users can't communicate to the server over port 22.

Administrators will also need to create exceptions for environment specific items, such as required shared resources. These will be specific to each institution.

Implement Traffic Monitoring

In case of attack against the Redis server from an outside source, some form of intrusion detection should be in place. A few items that can be monitored to flag traffic as possibly malicious, or drop the traffic entirely:

1. If anything other than a Blackboard application server or system administration server tries to access the Redis server, it should be flagged.
2. If a bad password is entered, it should be flagged.
3. If too many requests are made to Redis server from a single location, flag that traffic. Monitoring typical usage patterns, as well as speculating on edge cases (say, during finals week), can help determine how many requests per minute are expected

Implementing these intrusion detection rules will help determine whether anyone is trying to penetrate the Redis server or application.

Building Block Permissions

Java Type	Name	Actions	Why is this permission necessary?
persist	*	*	<ul style="list-style-type: none">• The Settings page needs to persist and load configuration data into SYSTEM_REGISTRY table.• Persists and loads Redis server and cache statistics for the specified duration to the Stats database for trend analysis.

java.io.FilePermission	<ol style="list-style-type: none"> 1. BB_HOME/- 2. BB_CONTENT/- 3. "\${java.home}/- 	<ol style="list-style-type: none"> 1. read, write, delete 2. read, write, delete 3. read 	<ul style="list-style-type: none"> • Reads properties files under Blackboard Learn home directory. • Saves a hash value unique to a Learn instance, shared by all Learn application servers within a cluster. It is used in Redis cache key to avoid collision from other Redis cache usage by another Learn instance or other applications. • Ehcache diskoverflow feature
java.lang.reflect.ReflectPermission	suppressAccessChecks	*	<ul style="list-style-type: none"> • Used by Spring framework and an object graph serialization framework, Kryo.
java.lang.RuntimePermission	*	*	<ul style="list-style-type: none"> • Used by various Blackboard Learn components including log services and service managers.
java.net.SocketPermission	*	connect, accept,	<ul style="list-style-type: none"> • Connection to Redis server and

		resolve, listen	peer Blackboard Learn application nodes.
javax.management. MBeanServerPermission	*	*	<ul style="list-style-type: none"> To expose cache statistics to MBean.
javax.management. MBeanPermission	*	*	<ul style="list-style-type: none"> To expose cache statistics to MBean.
attribute	user.authinfo	get	<ul style="list-style-type: none"> Gets user roles for operations.