

CLAIMS

What is claimed is:

1. A method for recovery from node crash, applicable to a proxy server in a master-slave system, the master-slave system further comprising a target master node controlled by the proxy server and a target slave node corresponding to the target master node, the method comprising:

obtaining a pre-stored persistent file from the target slave node when the target master node and the target slave node both crash; wherein, the target slave node stores a backup of cache data cached in the target master node, and the persistent file is generated based on cache data in the target slave node;

deploying a target master node that does not crash based on the persistent file; and

deploying a target slave node corresponding to the target master node that does not crash.

2. The method according to claim 1, wherein, nodes in the master-slave system are created on a node builder;

deploying a target master node that does not crash based on the persistent file comprises:

in a case where a first node builder crashes and a reboot of the first node builder fails, creating a new node builder in a physical machine controlled by the proxy server, wherein the first node builder is a node builder where the crashed target master node is created; and creating a new target master node in the new node builder based on the persistent file; and

in a case where the first node builder does not crash or is successfully rebooted after crashing, rebooting the crashed target master node based on the persistent file.

3. The method according to claim 2, wherein, the master-slave system further comprises a non-target master node controlled by the proxy server in addition to the target master node, and a non-target slave node corresponding to the non-target master node;

all the master nodes and slave nodes in the master-slave system form a cache database cluster, each node in the cache database cluster carries a node configuration file, which

contains a universal unique identifier (UUID) of the node;

wherein creating a new target master node in the new node builder based on the persistent file comprises:

determining whether the target node configuration file of the crashed target master node is lost;

if the configuration file is lost, creating a node in the new node builder based on the persistent file, adding the created node into the cache database cluster, obtaining a UUID of the target master node from nodes other than the target master node in the cache database cluster, and configuring the created node with the UUID of the target master node, so as to complete the creation of a new target master node; and

if the configuration file is not lost, creating a node in the new node builder based on the persistent file and the target node configuration file, and adding the created node to the cache database cluster, so as to complete the creation of a new target master node.

4. The method according to claim 3, wherein, rebooting the crashed target master node based on the persistent file includes:

determining whether the target node configuration file in the crashed target master node is lost;

if the target node configuration file is lost, obtaining a node configuration file from nodes other than the target master node in the cache database cluster, and rebooting the crashed target master node based on the persistent file and the obtained node configuration file; and

if the target node configuration file is not lost, rebooting the crashed target master node based on the persistent file and the target node configuration file.

5. The method according to claim 2, further comprising:

monitoring whether the new node builder is successfully created;

if not, generating an alarm message of failure.

6. The method according to claim 1, further comprising:

controlling the deployed target master node to disable a write service directed to its own cache database; and

after deploying a target slave node corresponding to the target master node that does not crash, controlling the deployed target master node to enable the write service directed to its own cache database.

7. The method according to claim 6, further comprising:

determining whether all cache data written into the deployed target slave node is completely persisted into persistent data;

if the cache data is completely persisted into persistent data, performing the operation of controlling the deployed target master node to enable a write service directed to its own cache database.

8. The method according to any one of claims 1-7, wherein, deploying a target slave node corresponding to the target master node that does not crash comprises:

deploying the target slave node corresponding to the target master node that does not crash based on a master-slave replication mechanism.

9. The method according to claim 2, wherein, the new node builder is created in a physical machine where the first node builder is located to replace the first node builder.

10. An apparatus for recovery from a node crash, applicable to a proxy server in a master-slave system, the master-slave system further comprising a target master node controlled by the proxy server and a target slave node corresponding to the target master node, and the apparatus comprising:

a determining module, configured for obtaining a pre-stored persistent file from the target slave node when the target master node and the target slave node both crash; wherein, the target slave node stores a backup of cache data cached in the target master node, and the persistent file is generated based on cache data in the target slave node;

a first deploying module, configured for deploying a target master node that does not crash based on the persistent file; and

a second deploying module, configured for deploying a target slave node corresponding to the target master node that does not crash.

11. The apparatus according to claim 10, wherein, nodes in the master-slave system are created on a node builder;

the first deploying module is further configured for:

in a case where a first node builder crashes and a reboot of the first node builder fails, creating a new node builder in a physical machine controlled by the proxy server; and creating a new target master node in the new node builder based on the persistent file, wherein the first node builder is a node builder where the crashed target master node is created; and

in a case where the first node builder does not crash or is successfully rebooted after crashing, rebooting the crashed target master node based on the persistent file.

12. The apparatus according to claim 11, wherein, the master-slave system further comprises a non-target master node controlled by the proxy server in addition to the target master node, and a non-target slave node corresponding to the non-target master node;

all the master nodes and slave nodes in the master-slave system form a cache database cluster, each node in the cache database cluster carries a node configuration file, which contains a universal unique identifier (UUID) of the node;

the first deploying module is further configured for:

in a case where a first node builder crashes and a reboot of the first node builder fails, creating a new node builder in a physical machine controlled by the proxy server, and determining whether the target node configuration file of the crashed target master node is lost;

if the configuration file is lost, creating a node in the new node builder based on the persistent file, adding the created node into the cache database cluster, obtaining a UUID of the target master node from nodes other than the target master node in the cache database cluster, configuring the created node with the UUID of the target master node, so as to complete the creation of a new target master node;

if the configuration file is not lost, creating a node in the new node builder based on the persistent file and the target node configuration file, and adding the created node to the cache database cluster, so as to complete the creation of a new target master node;

in a case where the first node builder does not crash or is successfully rebooted after crashing, determining whether the target node configuration file in the crashed

target master node is lost;

if the target node configuration file is lost, obtaining a node configuration file from nodes other than the target master node in the cache database cluster, and rebooting the crashed target master node based on the persistent file and the obtained node configuration file; and

if the target node configuration file is not lost, rebooting the crashed target master node based on the persistent file and the target node configuration file.

13. The apparatus according to claim 11, further comprising:

a monitoring module configured for monitoring whether the new node builder is successfully created; and

a generating module configured for generating an alarm message of failure when the monitoring module determines that no new node is successfully created.

14. The apparatus according to claim 10, further comprising:

a control module configured for, controlling the deployed target master node to disable a write service directed to its own cache database; and

controlling the deployed target master node to enable the write service directed to its own cache database after a target slave node corresponding to the target master node that does not crash is deployed.

15. The apparatus according to claim 14, further comprising:

a judging module, configured for determining whether all cache data written into the deployed target slave node is completely persisted into persistent data;

the control module is further configured for controlling the deployed target master node to enable a write service directed to its own cache database when the judging module determines that the cache data has been completely persisted.

16. The apparatus according to any one of claims 10-15, wherein, the second deploying module is configured for:

deploying the target slave node corresponding to the target master node that does not crash based on a master-slave replication mechanism.

17. An electronic device comprising a processor and a memory, wherein,

the memory is configured for storing a computer program; and

the processor is configured for performing, when executing the program stored in the memory, any method according to any of claims 1-9.

18. A computer-readable storage medium having a computer program stored thereon which, when executed by a processor, causes the processor to perform any method according to any of claims 1-9.

19. A computer program product containing instructions which, when executed by a computer, cause the computer to perform any method according to any of claims 1-9.

20. A computer program which, when executed by a computer, causes the computer to perform any method according to any of claims 1-9.