Monitor and Restart Offline Slaves

This script can monitor and restart offline nodes if they are not disconnected manually.

Of course you can disable email notification. Thanks author for email notification.

You can run this script in Jenkins Console. But I think it is good idea to create Jenkins task, which will be run periodically (for example hourly). Groovy Plugin needs for Jenkins jobs.

Also see Display Information About Nodes

```
import hudson.model.*
import hudson.node_monitors.*
import hudson.slaves.*
import java.util.concurrent.*

jenkins = Hudson.instance

import javax.mail.internet.*;
import javax.mail.*
import javax.activation.*


def sendMail (slave, cause) {

 message = slave + " slave is down. Check http://JENKINS_HOSTNAME:
JENKINS_PORT/computer/" + slave + "\nBecause " + cause
 subject = slave + " slave is offline"
 toAddress = "JENKINS_ADMIN@YOUR_DOMAIN"
 fromAddress = "JENKINS@YOUR_DOMAIN"
 host = "SMTP_SERVER"
 port = "SMTP_PORT"

 Properties mprops = new Properties();
 mprops.setProperty("mail.transport.protocol","smtp");
 mprops.setProperty("mail.host",host);
 mprops.setProperty("mail.smtp.port",port);

 Session lSession = Session.getDefaultInstance(mprops,null);
 MimeMessage msg = new MimeMessage(lSession);


 //tokenize out the recipients in case they came in as a list
 StringTokenizer tok = new StringTokenizer(toAddress,";");
 ArrayList emailTos = new ArrayList();
 while(tok.hasMoreElements()){
 emailTos.add(new InternetAddress(tok.nextElement().toString()));
 }
 InternetAddress[] to = new InternetAddress[emailTos.size()];
 to = (InternetAddress[]) emailTos.toArray(to);
 msg.setRecipients(MimeMessage.RecipientType.TO,to);
 InternetAddress fromAddr = new InternetAddress(fromAddress);
 msg.setFrom(fromAddr);
 msg.setFrom(new InternetAddress(fromAddress));
 msg.setSubject(subject);
 msg.setText(message)
```

```
 Transport transporter = lSession.getTransport("smtp");
 transporter.connect();
 transporter.send(msg);
}


def getEnviron(computer) {
   def env
   def thread = Thread.start("Getting env from ${computer.name}", { env
= computer.environment })
   thread.join(2000)
   if (thread.isAlive()) thread.interrupt()
   env
}

def slaveAccessible(computer) {
    getEnviron(computer)?.get('PATH') != null
}


def numberOfflineNodes = 0
def numberNodes = 0
for (slave in jenkins.slaves) {
   def computer = slave.computer
   numberNodes ++
   println ""
   println "Checking computer ${computer.name}:"
   def isOK = (slaveAccessible(computer) && !computer.offline)
   if (isOK) {
     println "\t\tOK, got PATH back from slave ${computer.name}."
     println('\tcomputer.isOffline: ' + slave.getComputer().
isOffline());
     println('\tcomputer.isTemporarilyOffline: ' + slave.getComputer().
isTemporarilyOffline());
     println('\tcomputer.getOfflineCause: ' + slave.getComputer().
getOfflineCause());
     println('\tcomputer.offline: ' + computer.offline);


   } else {
     numberOfflineNodes ++
     println "  ERROR: can't get PATH from slave ${computer.name}."
     println('\tcomputer.isOffline: ' + slave.getComputer().
isOffline());
     println('\tcomputer.isTemporarilyOffline: ' + slave.getComputer().
isTemporarilyOffline());
     println('\tcomputer.getOfflineCause: ' + slave.getComputer().
getOfflineCause());
     println('\tcomputer.offline: ' + computer.offline);
     sendMail(computer.name, slave.getComputer().getOfflineCause().
toString())
     if (slave.getComputer().isTemporarilyOffline()) {
      if (!slave.getComputer().getOfflineCause().toString().contains
("Disconnected by")) {
         computer.setTemporarilyOffline(false, slave.getComputer().
getOfflineCause())
```

```
        }
      } else {
          computer.connect(true)
      }
    }
  }
println ("Number of Offline Nodes: " + numberOfflineNodes)
println ("Number of Nodes: " + numberNodes)
```