ZooKeeper Information, Installation, and Disaster Recovery Documentation

What is ZooKeeper?

From the stackoverflow article:
You may describe ZooKeeper as a replicated synchronization service with eventual consistency.

http://stackoverflow.com/questions/3662995/explaining-apache-ZooKeeper
http://www.amazon.com/gp/product/1449361307 (JohnB is ordering this)

For some info on "replicated synchronization service" see this article
https://msdn.microsoft.com/en-us/library/dn589787.aspx

Also: http://stackoverflow.com/questions/15884697/difference-between-data-replication-and-synchronization

Why do we care?

From the LucidWorks article mentioned below:
SolrCloud automates a lot of the manual labor required in [the older] master-slave through using ZooKeeper nodes to monitor the state of the cluster as well as additional Solr features that understand how to interact with other machines in their Solr cluster.

How does ZooKeeper work with SolrCloud ?
(Our primary use of ZooKeeper here is SolrCloud)

I like this article as it gives a good overview and IT-type advice on some important considerations for the ZooKeeper machines.
https://support.lucidworks.com/hc/en-us/articles/201298317-What-is-SolrCloud-And-how-does-it-compare-to-master-slave-

TL;DR Summary of the article above:

SolrCloud is a set of new features and functionality added in Solr 4.0 to enable a new way of creating durable, highly available Solr clusters with commodity hardware. While similar in many ways to master-slave, SolrCloud automates a lot of the manual labor required in master-slave through using ZooKeeper nodes to monitor the state of the cluster as well as additional Solr features that understand how to interact with other machines in their Solr cluster.

In short, where master-slave requires manual effort to change the role of a node in a given cluster and to add new nodes to a cluster, SolrCloud aims to automate a lot of that work and allow seamless addition of new nodes to a cluster and to work around downed nodes with minimal oversight.

Important points to remember:

1. ZooKeeper **must** to be up and healthy before you start SOLR, Kafka, or the microservices.

You can start the SOLR or Kafka VMs, as long they don't start the applications on startup - or if they start in "non ZooKeeper Cloud" mode. If the SOLR or Kafka applications start on OS boot, and they start in "Cloud" mode, they will fail to start if they can't connect successfully to ZooKeeper. In that case, make sure ZooKeeper ensemble is running first.

At this point, although they were set up "for production" and Solr boots when the OS boots, the Solr VMs require a restart in "Cloud" mode before it "talks" to ZooKeeper. This appears to be a bug in SOLR 5.4 (I tried to solve, but couldn't – no answer on stackoverflow either). Just be aware that ZooKeeper had better be there when you issue that restart command.

2. You need to follow these instructions on at LEAST 3 machines or VMs in order to have a full ZooKeeper "ensemble". Obviously, although the list of machines in the zoo.cfg file (step 4) will be the same on each machine, the number in the myid file (step 3) had better be different on each VM. The simplest and most logical is just 1, 2, 3, etc. -- unless there is a need for some complex numbering scheme.

I haven't tried to "break" this – in my set up the number in the myid file on a particular VM is the same number as is in the zoo.cfg file for that VM's IP address. No guarantees for what happens if you mix them up. I wouldn't do it.

Nothing is perfect, especially software.

http://arstechnica.com/information-technology/2015/05/the-discovery-of-apache-zookeepers-poison-packet/

TL;DR summary:
At high throughput, ZooKeeper may "lock up" IF you're running Linux 3.0+ kernel on Xen 4.1 and 4.3. I don't think this should be an issue for us.

ZooKeeper Installation and Configuration Steps.
==========

1. Untar the ZooKeeper tar.gz file in a directory of your choice.
(/opt?) The ZooKeeper folks don't seem to have any real concern about
"installation for Production"

-- (a simple tar -zxvf [filename] will install it on Linux)

2. Create a data directory (where ZooKeeper keeps it's data) where you want it

I put it inside the ZooKeeper directory thus:
/home/john/ZooKeeper-3.4.6/data

(Note: I have some very simple .sh files that can start and stop ZooKeeper.
We may want to make VM's that start ZooKeeper on boot and handle
service ZooKeeper start / service ZooKeeper stop -- I didn't go that far)

Consider:

--The "owner" of this directory should probably be whatever user will be starting the ZooKeeper process. I did this as my own user and used sudo all over the place in my dev environment so we probably need to decide on a user (and any other Admin-specific things) and do it all that way...

-- You might want to put it in /var and call it ZooKeeper_data. As far as I can tell, it does not matter as long as the config file in the next step has the right path to the data directory.

3. Add the "myid" file.
You read right, no file extension.
It should reside inside the data directory you created in step 2.

It should contain only a single number that represents the id
of this server. These must be unique to the ZooKeeper ensemble.
No other server in the ensemble should have the same id number
in it's myid file.

Nothing is needed except the id

nano /home/john/zookeeper-3.4.6/data/myid

#==============================
1
#==============================

4. Copy (cp on Linux) the zoo.sample.cfg to zoo.cfg

Then, modify your new zoo.cfg in TWO ways:

a. dataDir=/home/john/ZooKeeper-3.4.6/data
(or wherever YOU decide to put the data directory)

b. Add a list of ALL the ZooKeeper ensemble machines by IP or name (if name resolves) I used IP addresses exclusively. This is the list at the bottom of the zoo.cfg copy/paste below. I'm pasting here for convenience too:

server.1=192.168.56.5:2888:3888
server.2=192.168.56.6:2888:3888
server.3=192.168.56.7:2888:3888

I did not modify anything else in this file.

What the heck are those port numbers for?

http://zookeeper.apache.org/doc/trunk/zookeeperstarted.html
This page has something to say about the port numbers 2888:3888...

Finally, note the two port numbers after each server name: "2888" and "3888". Peers use the former port to connect to other peers. Such a connection is necessary so that peers can communicate, for example, to agree upon the order of updates. More specifically, a ZooKeeper server uses this port (2888) to connect followers to the leader.

When a new leader arises, a follower opens a TCP connection to the leader using this port. Because the default leader election also uses TCP, we currently require another port for leader election. This is the second port in the server entry (3888).

Naturally, it stands to reason that if there are conflicts on the first port when ZooKeeper is coming online, that ZooKeeper instance, at the very least, will not be able to come online. If that prevents a full quorum (as it would in the case of just three ZooKeeper instances) then ZooKeeper as a service is hamstrung and will be unable to serve SOLR, Kafka, or any of the microservices.

If the second port is blocked when a leader election is needed, the election will fail and something pretty significant is going to go wrong. I have not tested whether this port needs to be open when Zookeeper starts, although if I was programming it, I would have tried the port on startup and fail with error messages if it's unavailable.

Below is a copy/paste of the entire zoo.cfg file after I made my modifications. The file should be identical on all zookeeper machines.

/home/john/ZooKeeper-3.4.6/conf/zoo.cfg

```
==========================================
# The number of milliseconds of each tick
tickTime=2000
# The number of ticks that the initial
# synchronization phase can take
initLimit=10
# The number of ticks that can pass between
# sending a request and getting an acknowledgement
syncLimit=5
# the directory where the snapshot is stored.
# do not use /tmp for storage, /tmp here is just
# example sakes.
dataDir=/home/john/ZooKeeper-3.4.6/data
# the port at which the clients will connect
clientPort=2181
# the maximum number of client connections.
# increase this if you need to handle more clients
#maxClientCnxns=60
#
# Be sure to read the maintenance section of the
# administrator guide before turning on autopurge.
#
# http://ZooKeeper.apache.org/doc/current/ZooKeeperAdmin.html#sc_maintenance
#
# The number of snapshots to retain in dataDir
#autopurge.snapRetainCount=3
# Purge task interval in hours
# Set to "0" to disable auto purge feature
#autopurge.purgeInterval=1

server.1=192.168.56.5:2888:3888
server.2=192.168.56.6:2888:3888
server.3=192.168.56.7:2888:3888
==============================================
```

Notes: Bringing new ZooKeeper node into existing ensemble

1. Add the new ZooKeeper to the zoo.cfg file on EVERY machine thus:
server.X=192.168.56.XX:2888:3888

X = the number in the myid file on the server in question.

I did not need to restart any running servers in the case of adding a ZooKeeper server to the ensemble, which is how I verified this document. I'm not sure if we'll ever need to do that. If it is a concern for IT and Disaster Recovery, we should make sure we include some realistic disaster scenarios in our testing.

I'm going to leave that for the second stage when Kevin and I are talking about this.

2. Also, Exhibitor is of interest. I haven't used it and we should talk about it and evaluate its usefulness in our situation. Exhibitor is a tool that provides a browser-based GUI for working with ZooKeeper. It can also help with backups for ZooKeeper. Of course, it's another tool with it's own care and feeding.

http://techblog.netflix.com/2012/04/introducing-exhibitor-supervisor-system.html

Also here:
https://github.com/Netflix/exhibitor/wiki/Running-Exhibitor
https://github.com/Netflix/exhibitor/wiki/Building-Exhibitor

3. In Prod, ZooKeeper could be a service on the system if we want...

Here are some hints: http://www.programering.com/a/MjNxQjNwATc.html

4. How do you back up ZooKeeper?

http://stackoverflow.com/questions/6394140/how-do-you-backup-ZooKeeper/33729560#33729560

Or just have more nodes? In a sense, a cluster is it's own backup...

5. And this, from here: https://kafka.apache.org/documentation.html#zkversion
I don't think our usage is high enough to worry, but it's worth noting.

Use care with virtualization: It can work, depending on your cluster layout and read/write patterns and SLAs, but the tiny overheads introduced by the virtualization layer can add up and throw off ZooKeeper, as it can be very time sensitive.