Improvements in CTDB and Clustered Samba testing

Martin Schwenke <martin@meltin.net>

Samba Team
IBM (Australia Development Laboratory, Linux Technology Center)

SambaXP 2019

Overview

- Testing with CTDB local daemons
- Autocluster 1.x

Testing with CTDB local daemons

Testing with CTDB local daemons

Why?

- Test a subset of CTDB functionality on a single machine

Where?

- Developer workstation
- Nightly regression testing
- Samba autobuild
- GitLab CI
- . . .

How?

- `simple` testsuite had the ability to start several daemons
- Daemons were started and stopped via dummy tests
- Less starts/stops made testing faster...
- . . . but possibly error prone

Testing with CTDB local daemons

Why make this standalone?

Testing with CTDB local daemons

Why make this standalone?

- A useful development and debugging tool

Testing with CTDB local daemons

Why make this standalone?

- A useful development and debugging tool
- Enable cluster testing in autobuild

Testing with CTDB local daemons

Why make this standalone?

- A useful development and debugging tool
- Enable cluster testing in autobuild
- Test scalability: how many local daemons can we run?

Testing with CTDB local daemons

Why make this standalone?

- A useful development and debugging tool
- Enable cluster testing in autobuild
- Test scalability: how many local daemons can we run?
- Cool hack?

Testing with CTDB local daemons

Available commands

```
$ ./tests/local_daemons.sh -h
usage: ./tests/local_daemons.sh <directory> <command> [ <options>... ]

Commands:
  setup          Set up daemon configuration according to given options
  start          Start specified daemon(s)
  stop           Stop specified daemon(s)
  onnode         Run a command in the environment of specified daemon(s)
  print-socket   Print the Unix domain socket used by specified daemon(s)
  dump-logs      Dump logs for specified daemon(s) to stdout

All commands use <directory> for daemon configuration

Run command with -h option to see per-command usage
```

Testing with CTDB local daemons

setup command usage

```
$ ./tests/local_daemons.sh foo setup -h
./tests/local_daemons.sh <directory> setup [ <options>... ]

Options:
  -F            Disable failover (default: failover enabled)
  -N <file>     Nodes file (default: automatically generated)
  -n <num>      Number of nodes (default: 3)
  -P <file>     Public addresses file (default: automatically generated)
  -R            Use a command for the recovery lock (default: use a file)
  -S <library>  Socket wrapper shared library to preload (default: none)
  -6            Generate IPv6 IPs for nodes, public addresses (default: IPv4)
```

Testing with CTDB local daemons

setup command demo

```
$ ./tests/local_daemons.sh foo setup -n 100
Node 87 will have no public IPs.
$ ls foo
node.0   node.19  node.29  node.39  node.49  node.59  node.69  node.79  node.89  node.99
node.1   node.2   node.3   node.4   node.5   node.6   node.7   node.8   node.9   nodes
node.10  node.20  node.30  node.40  node.50  node.60  node.70  node.80  node.90  public_addresses
node.11  node.21  node.31  node.41  node.51  node.61  node.71  node.81  node.91
node.12  node.22  node.32  node.42  node.52  node.62  node.72  node.82  node.92
node.13  node.23  node.33  node.43  node.53  node.63  node.73  node.83  node.93
node.14  node.24  node.34  node.44  node.54  node.64  node.74  node.84  node.94
node.15  node.25  node.35  node.45  node.55  node.65  node.75  node.85  node.95
node.16  node.26  node.36  node.46  node.56  node.66  node.76  node.86  node.96
node.17  node.27  node.37  node.47  node.57  node.67  node.77  node.87  node.97
node.18  node.28  node.38  node.48  node.58  node.68  node.78  node.88  node.98
$ ls foo/node.0
ctdb.conf  db  debug-hung-script.sh  events  functions  nodes  notify.sh  public_addresses  run  var
$ pidof ctdbd
$
```

Testing with CTDB local daemons

start command usage

```
$ ./tests/local_daemons.sh foo start -h
usage: ./tests/local_daemons.sh <directory> start <nodes>

<nodes> can be  "all", a node number or any specification supported by onnode
```

Testing with CTDB local daemons

start command demo

```
$ ./tests/local_daemons.sh foo start 4
$ pidof ctdbd | wc -w
2
$ ./tests/local_daemons.sh foo start 0-9

>> NODE: 127.0.0.1 <<

>> NODE: 127.0.0.2 <<

>> NODE: 127.0.0.3 <<

...

>> NODE: 127.0.0.9 <<

>> NODE: 127.0.0.10 <<
$ pidof ctdbd | wc -w
20
$ ./tests/local_daemons.sh foo start all

>> NODE: 127.0.0.1 <<

...

>> NODE: 127.0.0.100 <<
$ pidof ctdbd | wc -w
200
```

Testing with CTDB local daemons

onnode command usage

```
$ ./tests/local_daemons.sh foo onnode -h
usage: ./tests/local_daemons.sh <directory> onnode <nodes> <command>...

<nodes> can be  "all", a node number or any specification supported by onnode
```

Testing with CTDB local daemons

onnode command demo

```
$ ./tests/local_daemons.sh foo onnode 4 ctdb pnn
4
$ ./tests/local_daemons.sh foo onnode 4 ctdb nodestatus
pnn:4 127.0.0.5        OK (THIS NODE)
$ ./tests/local_daemons.sh foo onnode -q 0-9 ctdb nodestatus
pnn:0 127.0.0.1        OK (THIS NODE)
pnn:1 127.0.0.2        OK (THIS NODE)
pnn:2 127.0.0.3        OK (THIS NODE)
pnn:3 127.0.0.4        OK (THIS NODE)
pnn:4 127.0.0.5        OK (THIS NODE)
pnn:5 127.0.0.6        OK (THIS NODE)
pnn:6 127.0.0.7        OK (THIS NODE)
pnn:7 127.0.0.8        OK (THIS NODE)
pnn:8 127.0.0.9        OK (THIS NODE)
pnn:9 127.0.0.10       OK (THIS NODE)
$ ./tests/local_daemons.sh foo onnode 4 ctdb nodestatus all
Number of nodes:100
pnn:0 127.0.0.1        OK
...
pnn:4 127.0.0.5        OK (THIS NODE)
pnn:5 127.0.0.6        OK
...
pnn:99 127.0.0.100      OK
$ echo $?
0
```

Testing with CTDB local daemons

stop command usage

```
$ ./tests/local_daemons.sh foo stop -h
usage: ./tests/local_daemons.sh <directory> stop <nodes>

<nodes> can be  "all", a node number or any specification supported by onnode
```

Testing with CTDB local daemons

stop command demo

```
$ ./tests/local_daemons.sh foo stop 5
$ pidof ctdbd | wc -w
198
$ ./tests/local_daemons.sh foo stop 90-99
$ pidof ctdbd | wc -w
178
$ ./tests/local_daemons.sh foo stop all
[127.0.0.6] connect() failed, errno=111
[127.0.0.6] Failed to connect to CTDB daemon (foo/node.5/run/ctdbd.socket)
[127.0.0.96] connect() failed, errno=111
[127.0.0.96] Failed to connect to CTDB daemon (foo/node.95/run/ctdbd.socket)
[127.0.0.97] connect() failed, errno=111
[127.0.0.97] Failed to connect to CTDB daemon (foo/node.96/run/ctdbd.socket)
[127.0.0.94] connect() failed, errno=111
[127.0.0.94] Failed to connect to CTDB daemon (foo/node.93/run/ctdbd.socket)
...
[127.0.0.100] connect() failed, errno=111
[127.0.0.100] Failed to connect to CTDB daemon (foo/node.99/run/ctdbd.socket)
[127.0.0.91] connect() failed, errno=111
[127.0.0.91] Failed to connect to CTDB daemon (foo/node.90/run/ctdbd.socket)
[127.0.0.93] connect() failed, errno=111
[127.0.0.93] Failed to connect to CTDB daemon (foo/node.92/run/ctdbd.socket)
[127.0.0.95] connect() failed, errno=111
[127.0.0.95] Failed to connect to CTDB daemon (foo/node.94/run/ctdbd.socket)
[127.0.0.99] connect() failed, errno=111
[127.0.0.99] Failed to connect to CTDB daemon (foo/node.98/run/ctdbd.socket)
$ pidof ctdbd | wc -w
0
```

Testing with CTDB local daemons

dump-logs command usage

```
$ ./tests/local_daemons.sh foo dump-logs -h
usage: ./tests/local_daemons.sh <directory> dump-logs <nodes>

<nodes> can be  "all", a node number or any specification supported by onnode
```

- `<directory>` (e.g. `foo/`) can be pulled from a remote test machine and `dump-logs` can then be run locally
- Alternatively, just produce an output file via `dump-logs` and retrieve that...

Testing with CTDB local daemons

dump-logs command demo

```
$ ./tests/local_daemons.sh foo dump-logs all | wc -l
2018270
$ ./tests/local_daemons.sh foo dump-logs all | tail -n 10
2019/05/16 15:59:40.147328 node.52 ctdb-eventd[21219]: Shutting down
2019/05/16 15:59:40.147431 node.40 ctdbd[20933]: 127.0.0.41:4379: node 127.0.0.53:4379 is dead
2019/05/16 15:59:40.147450 node.40 ctdbd[20933]: Tearing down connection to dead node :52
2019/05/16 15:59:40.147738 node.52 ctdbd[21211]: Shutdown sequence complete, exiting.
2019/05/16 15:59:40.147762 node.52 ctdbd[21211]: CTDB daemon shutting down
2019/05/16 15:59:40.148231 node.40 ctdb-eventd[20941]: 00.test: shutdown event
2019/05/16 15:59:40.148371 node.40 ctdb-eventd[20941]: Received signal 15
2019/05/16 15:59:40.148386 node.40 ctdb-eventd[20941]: Shutting down
2019/05/16 15:59:40.148751 node.40 ctdbd[20933]: Shutdown sequence complete, exiting.
2019/05/16 15:59:40.148770 node.40 ctdbd[20933]: CTDB daemon shutting down
$ ./tests/local_daemons.sh foo dump-logs 0-9 | tail -n 2000 | head -n 10
2019/05/16 15:59:31.337143 node.8 ctdbd[18786]: Control modflags on node 0 - Unchanged - flags 0x0
2019/05/16 15:59:31.337213 node.9 ctdbd[18810]: Control modflags on node 0 - Unchanged - flags 0x0
2019/05/16 15:59:31.337628 node.4 ctdbd[17963]: Control modflags on node 0 - Unchanged - flags 0x0
2019/05/16 15:59:31.344454 node.0 ctdbd[18617]: Control modflags on node 1 - Unchanged - flags 0x0
2019/05/16 15:59:31.344470 node.1 ctdbd[18634]: Control modflags on node 1 - Unchanged - flags 0x0
2019/05/16 15:59:31.344523 node.2 ctdbd[18657]: Control modflags on node 1 - Unchanged - flags 0x0
2019/05/16 15:59:31.344605 node.3 ctdbd[18679]: Control modflags on node 1 - Unchanged - flags 0x0
2019/05/16 15:59:31.344663 node.6 ctdbd[18740]: Control modflags on node 1 - Unchanged - flags 0x0
2019/05/16 15:59:31.344726 node.7 ctdbd[18761]: Control modflags on node 1 - Unchanged - flags 0x0
2019/05/16 15:59:31.344776 node.8 ctdbd[18786]: Control modflags on node 1 - Unchanged - flags 0x0
```

Testing with CTDB local daemons

print-socket command usage

```
$ ./tests/local_daemons.sh foo print-socket -h
usage: ./tests/local_daemons.sh <directory> print-socket <nodes>

<nodes> can be  "all", a node number or any specification supported by onnode
```

Testing with CTDB local daemons

print-socket command demo

```
$ ./tests/local_daemons.sh foo print-socket 0-9
foo/node.0/run/ctdbd.socket
foo/node.1/run/ctdbd.socket
foo/node.2/run/ctdbd.socket
foo/node.3/run/ctdbd.socket
foo/node.4/run/ctdbd.socket
foo/node.5/run/ctdbd.socket
foo/node.6/run/ctdbd.socket
foo/node.7/run/ctdbd.socket
foo/node.8/run/ctdbd.socket
foo/node.9/run/ctdbd.socket
```

Testing with CTDB local daemons

How does this work?

Testing with CTDB local daemons

How does this work?

- `local_daemons.sh` is 450 lines of POSIX shell code

Testing with CTDB local daemons

How does this work?

- `local_daemons.sh` is 450 lines of POSIX shell code
- When `CTDB_TEST_MODE` is set then `ctdbd` and `ctdb` tool look at `CTDB_BASE` (e.g. `CTDB_BASE=foo/node.4`)

Testing with CTDB local daemons

How does this work?

- `local_daemons.sh` is 450 lines of POSIX shell code
- When `CTDB_TEST_MODE` is set then `ctdbd` and `ctdb` tool look at `CTDB_BASE` (e.g. `CTDB_BASE=foo/node.4`)
- Configuration files, socket(s), TDBs, logs, etc. are all relative to `CTDB_BASE` when running in test mode

Testing with CTDB local daemons

How does this work?

- `local_daemons.sh` is 450 lines of POSIX shell code
- When `CTDB_TEST_MODE` is set then `ctdbd` and `ctdb` tool look at `CTDB_BASE` (e.g. `CTDB_BASE=foo/node.4`)
- Configuration files, socket(s), TDBs, logs, etc. are all relative to `CTDB_BASE` when running in test mode
- `local_daemons.sh` sources some test infrastructure for:
  - finding things (e.g. helpers); and
  - setting up base directory for each node

Testing with CTDB local daemons

How does this work?

- `local_daemons.sh` is 450 lines of POSIX shell code
- When `CTDB_TEST_MODE` is set then `ctdbd` and `ctdb` tool look at `CTDB_BASE` (e.g. `CTDB_BASE=foo/node.4`)
- Configuration files, socket(s), TDBs, logs, etc. are all relative to `CTDB_BASE` when running in test mode
- `local_daemons.sh` sources some test infrastructure for:
  - finding things (e.g. helpers); and
  - setting up base directory for each node
- `local_daemons.sh` sets `ONNODE_SSH` to its own ssh implementation, which sets `CTDB_BASE` depending on target node and then runs the given command in a shell

Testing with CTDB local daemons

How does this work?

- `local_daemons.sh` is 450 lines of POSIX shell code
- When `CTDB_TEST_MODE` is set then `ctdbd` and `ctdb` tool look at `CTDB_BASE` (e.g. `CTDB_BASE=foo/node.4`)
- Configuration files, socket(s), TDBs, logs, etc. are all relative to `CTDB_BASE` when running in test mode
- `local_daemons.sh` sources some test infrastructure for:
  - finding things (e.g. helpers); and
  - setting up base directory for each node
- `local_daemons.sh` sets `ONNODE_SSH` to its own ssh implementation, which sets `CTDB_BASE` depending on target node and then runs the given command in a shell
- Works nicely in CTDB's `simple` testsuite, which runs in Samba autobuild

Autocluster 1.x

Autocluster 1.x

What is Autocluster < 1.0?

Autocluster 1.x

What is Autocluster < 1.0?

- `git://git.samba.org/autocluster.git`

Autocluster 1.x

What is Autocluster < 1.0?

- `git://git.samba.org/autocluster.git`
- Originally a bash script, written by Tridge in July 2008

Autocluster 1.x

What is Autocluster < 1.0?

- `git://git.samba.org/autocluster.git`
- Originally a bash script, written by Tridge in July 2008
- . . . with supporting Kickstart scripts, libvirt XML templates and supporting scripts

Autocluster 1.x

What is Autocluster < 1.0?

- `git://git.samba.org/autocluster.git`
- Originally a bash script, written by Tridge in July 2008
- ... with supporting Kickstart scripts, libvirt XML templates and supporting scripts
- Used to create and configure virtual clusters for testing CTDB

Autocluster 1.x

What is Autocluster < 1.0?

- `git://git.samba.org/autocluster.git`
- Originally a bash script, written by Tridge in July 2008
- . . . with supporting Kickstart scripts, libvirt XML templates and supporting scripts
- Used to create and configure virtual clusters for testing CTDB
- Didn't try to solve general virtualisation/deployment/configuration problem

Autocluster 1.x

What is Autocluster < 1.0?

- `git://git.samba.org/autocluster.git`
- Originally a bash script, written by Tridge in July 2008
- . . . with supporting Kickstart scripts, libvirt XML templates and supporting scripts
- Used to create and configure virtual clusters for testing CTDB
- Didn't try to solve general virtualisation/deployment/configuration problem
- Pre-dates Chef (January 2009), Vagrant (March 2010), OpenStack (October 2010), Ansible (February 2012) but not Puppet (2005)

Autocluster 1.x

What is Autocluster < 1.0?

- `git://git.samba.org/autocluster.git`
- Originally a bash script, written by Tridge in July 2008
- . . . with supporting Kickstart scripts, libvirt XML templates and supporting scripts
- Used to create and configure virtual clusters for testing CTDB
- Didn't try to solve general virtualisation/deployment/configuration problem
- Pre-dates Chef (January 2009), Vagrant (March 2010), OpenStack (October 2010), Ansible (February 2012) but not Puppet (2005)
- Maintained by me for over 10 years

Autocluster 1.x

What is Autocluster < 1.0?

- `git://git.samba.org/autocluster.git`
- Originally a bash script, written by Tridge in July 2008
- . . . with supporting Kickstart scripts, libvirt XML templates and supporting scripts
- Used to create and configure virtual clusters for testing CTDB
- Didn't try to solve general virtualisation/deployment/configuration problem
- Pre-dates Chef (January 2009), Vagrant (March 2010), OpenStack (October 2010), Ansible (February 2012) but not Puppet (2005)
- Maintained by me for over 10 years
- Spike of activity in 2014 to add structure, separate out configuration stage from VM deployment

Autocluster 1.x

What was wrong with Autocluster < 1.0?

Autocluster 1.x

What was wrong with Autocluster < 1.0?

- Supported only RHEL/CentOS clusters with libvirt

Autocluster 1.x

What was wrong with Autocluster < 1.0?

- Supported only RHEL/CentOS clusters with libvirt
- Fragile creation of ‘base’ images via kickstart magic

Autocluster 1.x

What was wrong with Autocluster < 1.0?

- Supported only RHEL/CentOS clusters with libvirt
- Fragile creation of ‘base’ images via kickstart magic
- Got stuck at RHEL 6.9…

Autocluster 1.x

What was wrong with Autocluster < 1.0?

- Supported only RHEL/CentOS clusters with libvirt
- Fragile creation of ‘base’ images via kickstart magic
- Got stuck at RHEL 6.9…
- …although some early versions of RHEL 7.x worked

Autocluster 1.x

Motivation for rewriting

Autocluster 1.x

Motivation for rewriting

- Broken, unmaintainable

Autocluster 1.x

Motivation for rewriting

- Broken, unmaintainable
- Large parts of this have since been done better

Autocluster 1.x

Motivation for rewriting

- Broken, unmaintainable
- Large parts of this have since been done better
- Tim Potter gave me a Chef demonstration in late 2014

Autocluster 1.x

Motivation for rewriting

- Broken, unmaintainable
- Large parts of this have since been done better
- Tim Potter gave me a Chef demonstration in late 2014
- Michael Adam demonstrated Vagrant at SambaXP 2015

Autocluster 1.x

Motivation for rewriting

- Broken, unmaintainable
- Large parts of this have since been done better
- Tim Potter gave me a Chef demonstration in late 2014
- Michael Adam demonstrated Vagrant at SambaXP 2015
- Stuck. . .

Autocluster 1.x

A vague plan. . .

Autocluster 1.x

A vague plan. . .

- Rewrite on top of other existing software

Autocluster 1.x

A vague plan. . .

- Rewrite on top of other existing software
- Checked reviews, tutorials and documentation

Autocluster 1.x

A vague plan. . .

- Rewrite on top of other existing software
- Checked reviews, tutorials and documentation
- Vagrant

Autocluster 1.x

A vague plan. . .

- Rewrite on top of other existing software
- Checked reviews, tutorials and documentation
- Vagrant
  - Michael's demo looked very promising and has a `git repo` containing a useful starting point

Autocluster 1.x

A vague plan...

- Rewrite on top of other existing software
- Checked reviews, tutorials and documentation
- Vagrant
  - Michael's demo looked very promising and has a `git repo` containing a useful starting point
  - Using alternate OS for cluster nodes should be easy

Autocluster 1.x

A vague plan...

- Rewrite on top of other existing software
- Checked reviews, tutorials and documentation
- Vagrant
  - Michael's demo looked very promising and has a `git repo` containing a useful starting point
  - Using alternate OS for cluster nodes should be easy
  - Create cluster of containers after a little extra development?

Autocluster 1.x

A vague plan. . .

- Rewrite on top of other existing software
- Checked reviews, tutorials and documentation
- Vagrant
  - Michael's demo looked very promising and has a git repo containing a useful starting point
  - Using alternate OS for cluster nodes should be easy
  - Create cluster of containers after a little extra development?
- Ansible (mature, declarative, lightweight on nodes: SSH only)

Autocluster 1.x

A vague plan...

- Rewrite on top of other existing software
- Checked reviews, tutorials and documentation
- Vagrant
  - Michael's demo looked very promising and has a `git repo` containing a useful starting point
  - Using alternate OS for cluster nodes should be easy
  - Create cluster of containers after a little extra development?
- Ansible (mature, declarative, lightweight on nodes: SSH only)
- January this year:
  Could this be a Google Summer of Code project?

Autocluster 1.x

A vague plan...

- Rewrite on top of other existing software
- Checked reviews, tutorials and documentation
- Vagrant
  - Michael's demo looked very promising and has a `git repo` containing a useful starting point
  - Using alternate OS for cluster nodes should be easy
  - Create cluster of containers after a little extra development?
- Ansible (mature, declarative, lightweight on nodes: SSH only)
- January this year:
  Could this be a Google Summer of Code project?
- It would have to be feasible...

Autocluster 1.x

A vague plan...

- Rewrite on top of other existing software
- Checked reviews, tutorials and documentation
- Vagrant
  - Michael's demo looked very promising and has a `git repo` containing a useful starting point
  - Using alternate OS for cluster nodes should be easy
  - Create cluster of containers after a little extra development?
- Ansible (mature, declarative, lightweight on nodes: SSH only)
- January this year:
  Could this be a Google Summer of Code project?
- It would have to be feasible...
- How do we determine feasibility?

Autocluster 1.x

A vague plan...

- Rewrite on top of other existing software
- Checked reviews, tutorials and documentation
- Vagrant
  - Michael's demo looked very promising and has a `git repo` containing a useful starting point
  - Using alternate OS for cluster nodes should be easy
  - Create cluster of containers after a little extra development?
- Ansible (mature, declarative, lightweight on nodes: SSH only)
- January this year:
  Could this be a Google Summer of Code project?
- It would have to be feasible...
- How do we determine feasibility?
- Hmmm...

Autocluster 1.x

First, try Vagrant + `vagrant-libvirt`

- Starting a cluster with shared storage is racy (`vagrant-libvirt` issue #825)
- Mounting filesystem from host via NFS needs NFS packages...
- ...which aren't in base image (aka. 'box')...
- ...but package installation times out if network is slow...
- ...so don't install packages in Vagrant...
- ...so don't use shared/synced folders...
- `vagrant-cachier` is unreliable...and uses NFS — no!
- `vagrant-timezone`/Ruby can't work out host timezone
- Sometimes private network interfaces do not come up
- Private networks controlled by NetworkManager by default — problems with node reboot

Autocluster 1.x

Minimal Vagrant + `vagrant-libvirt` solution

Autocluster 1.x

Minimal Vagrant + `vagrant-libvirt` solution

- `Vagrantfile` (~120 lines) + short helper script(s)

Autocluster 1.x

Minimal Vagrant + `vagrant-libvirt` solution

- `Vagrantfile` (~120 lines) + short helper script(s)
- Loads YAML configuration file

Autocluster 1.x

Minimal Vagrant + `vagrant-libvirt` solution

- `Vagrantfile` (~120 lines) + short helper script(s)
- Loads YAML configuration file
- Configure using basic Vagrant capabilities:
  - Shared block devices, with generated serial numbers
  - Proxies based on host proxies
  - Private networks, not controlled by NetworkManager
  - Default route

Autocluster 1.x

Minimal Vagrant + `vagrant-libvirt` solution

- `Vagrantfile` (~120 lines) + short helper script(s)
- Loads YAML configuration file
- Configure using basic Vagrant capabilities:
  - Shared block devices, with generated serial numbers
  - Proxies based on host proxies
  - Private networks, not controlled by NetworkManager
  - Default route
- Configure/check with short helper scripts:
  - Password-less SSH root access to and between nodes
  - Check that configured IP addresses for private networks are present

Autocluster 1.x

Node configuration via Ansible

Autocluster 1.x

Node configuration via Ansible

- Initial experiments

Autocluster 1.x

Node configuration via Ansible

- Initial experiments
- Ansible plays are (somewhat) declarative

Autocluster 1.x

Node configuration via Ansible

- Initial experiments
- Ansible plays are (somewhat) declarative
- Ansible plays should be idempotent

Autocluster 1.x

Node configuration via Ansible

- Initial experiments
- Ansible plays are (somewhat) declarative
- Ansible plays should be idempotent
- Read Best Practices for Working With Playbooks

Autocluster 1.x

Node configuration via Ansible

- Initial experiments
- Ansible plays are (somewhat) declarative
- Ansible plays should be idempotent
- Read Best Practices for Working With Playbooks
- Read it again! Awesome!

Autocluster 1.x

Node configuration via Ansible

- Initial experiments
- Ansible plays are (somewhat) declarative
- Ansible plays should be idempotent
- Read Best Practices for Working With Playbooks
- Read it again! Awesome!
- Iterated to try to get things right

Autocluster 1.x

Node configuration via Ansible

- Initial experiments
- Ansible plays are (somewhat) declarative
- Ansible plays should be idempotent
- Read Best Practices for Working With Playbooks
- Read it again! Awesome!
- Iterated to try to get things right
- Result...

```
$ git show --stat 51ff83d | tail -n 1
 69 files changed, 1169 insertions(+)
```

Autocluster 1.x

Ansible playbook —

```
$ cat ansible/node/site.yml
---
- import_playbook: ad.yml
- import_playbook: base.yml
- import_playbook: build.yml
- import_playbook: cbuild.yml
- import_playbook: storage.yml
- import_playbook: test.yml
- import_playbook: nas.yml
$ cat ansible/node/nas.yml
---
- hosts: nas-nodes
  remote_user: root

  roles:
    - common
    - clusterfs
    - nasrepos
    - ctdb
    - storage
    - nas
```

Ansible playbook — roles

```
$ find ansible/node/ -maxdepth 2 -type d
ansible/node/
ansible/node/roles
ansible/node/roles/common
ansible/node/roles/build
ansible/node/roles/clusterfs
ansible/node/roles/nas
ansible/node/roles/nasrepos
ansible/node/roles/ad
ansible/node/roles/storage
ansible/node/roles/ctdb
```

Autocluster 1.x

Ansible playbook — common role main task

```
$ cat ansible/node/roles/common/tasks/main.yml
---
- include_tasks: "{{ ansible_os_family | lower }}/{{ task }}.yml"
  with_list:
  - packages
  - firewall
  - ntp
  loop_control:
    loop_var: task

- meta: flush_handlers

- include_tasks: generic/{{ task }}.yml
  with_list:
  - selinux
  - autocluster
  - hosts
  - resolv_conf
  - ssh
  ...
  loop_control:
    loop_var: task
```

Ansible playbook — common role tasks

```
$ ls -1 ansible/node/roles/common/tasks/*
ansible/node/roles/common/tasks/main.yml

ansible/node/roles/common/tasks/generic:
autocluster.yml
hosts.yml
mount_home.yml
resolv_conf.yml
rsyslog.yml
selinux.yml
ssh.yml
timezone.yml

ansible/node/roles/common/tasks/redhat:
firewall.yml
ntp.yml
packages.yml
```

Autocluster 1.x

Ansible playbook — storage role

```
$ cat ansible/node/roles/storage/tasks/main.yml
---
- include_tasks: generic/{{ task }}.yml
  with_list:
  - clusterfs-{{ clusterfs.type }}
  loop_control:
    loop_var: task
$ ls -1 ansible/node/roles/storage/tasks/generic
clusterfs-gpfs-once.yml
clusterfs-gpfs.yml
```

Autocluster 1.x

Integration into bash script

Autocluster 1.x

Integration into bash script

- Integrate running `ansible-playbook`
- Integrate running `vagrant`

Autocluster 1.x

Integration into bash script

- Integrate running `ansible-playbook`
- Integrate running `vagrant`
- Both use a YAML configuration file. . .

Autocluster 1.x

Integration into bash script

- Integrate running `ansible-playbook`
- Integrate running `vagrant`
- Both use a YAML configuration file...
- ...generated by shell script from shell configuration file

Autocluster 1.x

Integration into bash script

- Integrate running `ansible-playbook`
- Integrate running `vagrant`
- Both use a YAML configuration file. . .
- . . . generated by shell script from shell configuration file
- Can I write a Python script to parse YAML config and produce an old-style shell configuration for the few remaining variables needed in the script?

Autocluster 1.x

Integration into bash script

- Integrate running `ansible-playbook`
- Integrate running `vagrant`
- Both use a YAML configuration file. . .
- . . . generated by shell script from shell configuration file
- Can I write a Python script to parse YAML config and produce an old-style shell configuration for the few remaining variables needed in the script?
- Of course!

Autocluster 1.x

Integration into bash script

- Integrate running `ansible-playbook`
- Integrate running `vagrant`
- Both use a YAML configuration file. . .
- . . . generated by shell script from shell configuration file
- Can I write a Python script to parse YAML config and produce an old-style shell configuration for the few remaining variables needed in the script?
- Of course!
- Hmmm. . . there isn't a lot of shell script left. . .

Autocluster 1.x

Integration into bash script

- Integrate running `ansible-playbook`
- Integrate running `vagrant`
- Both use a YAML configuration file. . .
- . . . generated by shell script from shell configuration file
- Can I write a Python script to parse YAML config and produce an old-style shell configuration for the few remaining variables needed in the script?
- Of course!
- Hmmm. . . there isn't a lot of shell script left. . .
- Why not invoke `vagrant` and `ansible-playbook` from the Python script?

Autocluster 1.x

Integration into bash script

- Integrate running `ansible-playbook`
- Integrate running `vagrant`
- Both use a YAML configuration file...
- ...generated by shell script from shell configuration file
- Can I write a Python script to parse YAML config and produce an old-style shell configuration for the few remaining variables needed in the script?
- Of course!
- Hmmm...there isn't a lot of shell script left...
- Why not invoke `vagrant` and `ansible-playbook` from the Python script?
- Wow! It's all just a small Python script now!

Autocluster 1.x

How much Python?

Autocluster 1.x

How much Python?

```
$ git show --oneline --stat 5cc52f2
5cc52f2 Rewrite autocluster in Python
 Makefile             |   8 +-
 autocluster.py       | 729 ++++++++++++++++++++++++++++++++++++++++++++
 autocluster.spec.in  |  10 ++-
 defaults.yml         | 102 ++++++++++++++++++++++++++
 example.yml          |  34 +++++++++
 5 files changed, 874 insertions(+), 9 deletions(-)
```

Autocluster 1.x

How much Python?

```
$ git show --oneline --stat 5cc52f2
5cc52f2 Rewrite autocluster in Python
 Makefile                |   8 +-
 autocluster.py          | 729 ++++++++++++++++++++++++++++++++++++++++++++
 autocluster.spec.in     |  10 ++-
 defaults.yml            | 102 ++++++++++++++++++++++++++
 example.yml             |  34 +++++++++
 5 files changed, 874 insertions(+), 9 deletions(-)
```

- Includes shared storage volume creation and deletion
- Doesn't include additional host setup functionality

Autocluster 1.x

How much removed?

Autocluster 1.x

How much removed?

```
$ git show --oneline --stat acab4ff | cat
acab4ff Remove bash autocluster script and supporting files
 README                                            |   43 +-
 autocluster                                       | 1639 -------------------
 ...
 .../scripts/cluster_configure/cluster-configure.py |  452 ------
 ...
 .../all/root/scripts/tasks/setup_clusterfs_gpfs.sh |  208 ---
 ...
 config.d/00base.defconf                           |  546 -------
 config.d/02kickstart.defconf                      |   47 -
 config.d/05diskimage_guestfish.defconf            |  193 ---
 config.d/05diskimage_guestmount.defconf           |  150 --
 config.d/05diskimage_loopback.defconf             |  237 ---
 config.d/10shareddisk.defconf                     |  311 ----
 ...
 templates/nas-kickstart.cfg                       |  122 --
 templates/node.xml                                |   35 -
 vircmd                                            |  161 --
 117 files changed, 2 insertions(+), 7211 deletions(-)
```

Autocluster 1.x

Experiments

Autocluster 1.x

Experiments

- Docker containers?
  - Vagrant CentOS 7 docker image (`roboxes/centos7`) can not be used with Vagrant: no vagrant user
  - Vagrant CentOS 7 docker image (`roboxes/centos7`) can not be upgraded: RPM checksum failure on a systemd package
  - `systemd` + SELinux + Docker == raging dumpster fire
    - SELinux is not namespaced
    - Disabling SELinux in Docker container disables it on host
    - Must run host in permissive mode
  - Vagrant does not set up private networks in Docker containers
  - Some of this attempt is stashed away in a branch

Autocluster 1.x

Experiments

- Docker containers?
  - Vagrant CentOS 7 docker image (`roboxes/centos7`) can not be used with Vagrant: no vagrant user
  - Vagrant CentOS 7 docker image (`roboxes/centos7`) can not be upgraded: RPM checksum failure on a systemd package
  - `systemd` + SELinux + Docker == raging dumpster fire
    - SELinux is not namespaced
    - Disabling SELinux in Docker container disables it on host
    - Must run host in permissive mode
  - Vagrant does not set up private networks in Docker containers
  - Some of this attempt is stashed away in a branch
- VirtualBox
  - Have `libvirt`, so no motivation to use this directly
  - Learned some things about VirtualBox + Vagrant
  - Have untested VirtualBox support is stashed away in a branch

Autocluster 1.x

What has been lost?

- Some IBM TSM (hierarchical storage management) support
- Support for testing `vsftpd` and `httpd`
- Multipath access to shared storage
- Support for iSCSI shared storage

Autocluster 1.x

What has been won?

Autocluster 1.x

What has been won?

- Maintainability

Autocluster 1.x

What has been won?

- Maintainability
- Base images (aka. ‘boxes’) are someone else’s problem

Autocluster 1.x

What has been won?

- Maintainability
- Base images (aka. 'boxes') are someone else's problem
- Ease of adding target platforms (e.g. Debian)
- Ease of adding alternate cluster filesystems

Autocluster 1.x

What has been won?

- Maintainability
- Base images (aka. 'boxes') are someone else's problem
- Ease of adding target platforms (e.g. Debian)
- Ease of adding alternate cluster filesystems
- Integrated host setup command and Ansible playbook

Autocluster 1.x

To do?

Autocluster 1.x

To do?

- Make Ansible playbooks more idempotent:

```
PLAY RECAP *****************************************************
m1ad1                : ok=35   changed=1    unreachable=0    failed=0
m1base1              : ok=28   changed=0    unreachable=0    failed=0
m1build1             : ok=33   changed=2    unreachable=0    failed=0
m1cbuild1            : ok=39   changed=3    unreachable=0    failed=0
m1n1                 : ok=89   changed=22   unreachable=0    failed=0
m1n2                 : ok=73   changed=14   unreachable=0    failed=0
m1n3                 : ok=73   changed=14   unreachable=0    failed=0
```

  Not a high priority because the focus is on initial configuration rather than ongoing configuration management

Autocluster 1.x

To do?

- Make Ansible playbooks more idempotent:

```
PLAY RECAP *****************************************************
m1ad1                : ok=35   changed=1    unreachable=0    failed=0
m1base1              : ok=28   changed=0    unreachable=0    failed=0
m1build1             : ok=33   changed=2    unreachable=0    failed=0
m1cbuild1            : ok=39   changed=3    unreachable=0    failed=0
m1n1                 : ok=89   changed=22   unreachable=0    failed=0
m1n2                 : ok=73   changed=14   unreachable=0    failed=0
m1n3                 : ok=73   changed=14   unreachable=0    failed=0
```

  Not a high priority because the focus is on initial configuration rather than ongoing configuration management

- Add some variations previously mentioned

Autocluster 1.x

To do?

- Make Ansible playbooks more idempotent:

```
PLAY RECAP *****************************************************
m1ad1              : ok=35   changed=1    unreachable=0    failed=0
m1base1            : ok=28   changed=0    unreachable=0    failed=0
m1build1           : ok=33   changed=2    unreachable=0    failed=0
m1cbuild1          : ok=39   changed=3    unreachable=0    failed=0
m1n1               : ok=89   changed=22   unreachable=0    failed=0
m1n2               : ok=73   changed=14   unreachable=0    failed=0
m1n3               : ok=73   changed=14   unreachable=0    failed=0
```

  Not a high priority because the focus is on initial configuration rather than ongoing configuration management
- Add some variations previously mentioned
- Improve host setup?

Legal Statement

Questions?