Demystifying
The Contrib Webserver

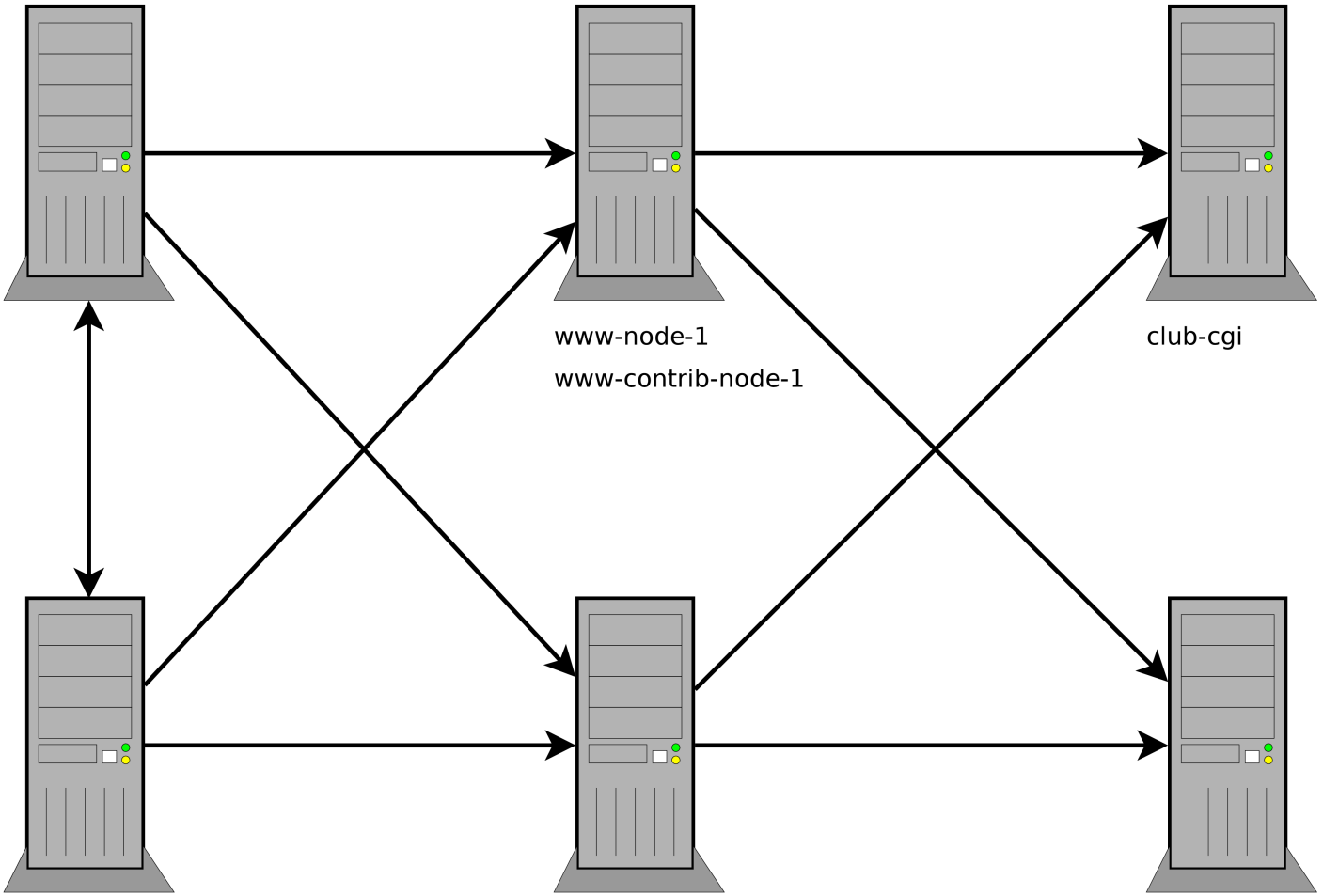
The Diagram

Load Balancers

Frontends

CGI Servers

lb-1 (master)
www{1,2}
www-contrib{1,2}



www-node-1
www-contrib-node-1

club-cgi

lb-2 (backup)
www{1,2}
www-contrib{1,2}

www-node-2
www-contrib-node-2

contrib-cgi
my-contrib

Infrastructure Hierarchy

- Load balancers
  - Everything through here -- lb1 master lb2 backup
  - if lb1 down lb2 steals ips
  - virtual ips of services get swapped around -- www1 www2 www-contrib1 www-contrib2
  - www1 doesn't really correspond to a real machine -- virtual ip -- only certain ports forwarded to a real machine
  - DNS A records for www -- go to www1 www2 ips
  - do not run AFS clients

Infrastructure Hierarchy

- Frontends
  - www-node-{1,2} www-contrib-node-{1,2}
  - possibly redirect to localhost instead of back to lbs if accessing lb tier from here?
  - what actually run apache
  - get sad if AFS gets sad
  - serve static web pages (anything that's not CGI in AFS) -- www.club www.contrib ftp rsync
  - do magic to figure out requests that are for CGI scripts and forward to contrib-cgi and club-cgi as appropriate

Infrastructure Hierarchy

- CGI servers
  - never hit directly
  - club-cgi contrib-cgi my-contrib (configuration for contrib)
  - no lb on my-contrib

Linux Virtual Services

- tech behind lb
- can be done via NAT (we don't do this)
  - all traffic has to go through gateway
  - all state lost on failover -- okay to drop connections on short-lived www connections
- can have lb forward requests, backends respond directly
  - sort of ip tunneling -- put new header on the top
  - machines must support encapsulation of packets
  - get more fragmentation
  - spoofing prevention can break this, since backend responds with virtual ip of lb

Linux Virtual Services

- can do the last option, but instead of sending ip, can send raw ethernet packets if you're on the same ethernet segment
  - can mess up ARP tables -- service servers do not ARP
  - this is what we do

contrib-cgi

- executed by a user on contrib-cgi that is unique to each user
- every andrew user has an account on contrib-cgi
- script takes andrew UID and adds a big offset, goes into /etc/passwd.contrib which ends up via passwd update script in /etc/passwd
- each org gets a cgi user too; generated in a scary way to get (probably) stable names and uids for orgs too
- generate contrib-org.conf with a bunch of RewriteRules to send /org/foo to the fake user
- none of this magic happens on club-cgi -- just club users from club passwd file

suexec on contrib-cgi

- used by apache to execute scripts with perms of user
- "run this script as this user"
- written in a paranoid way, sanity checks perms
- extensive club modifications
  - remove sanity checks which don't work in AFS
  - attempts to get krb tickets and afs tokens if you're set up for contrib key (see below)
  - sets some rlimits (max processes, max memory) to prevent DoS -- cgi_limits.conf and generate cgi_limits.db
  - redirects standard error to logfile -- does not catch forgetting Content-Type header

contribkey

- contrib/andrewuser@club
- set up via my-contrib
  - Procedure
- Allows authenticated AFS access from CGIs