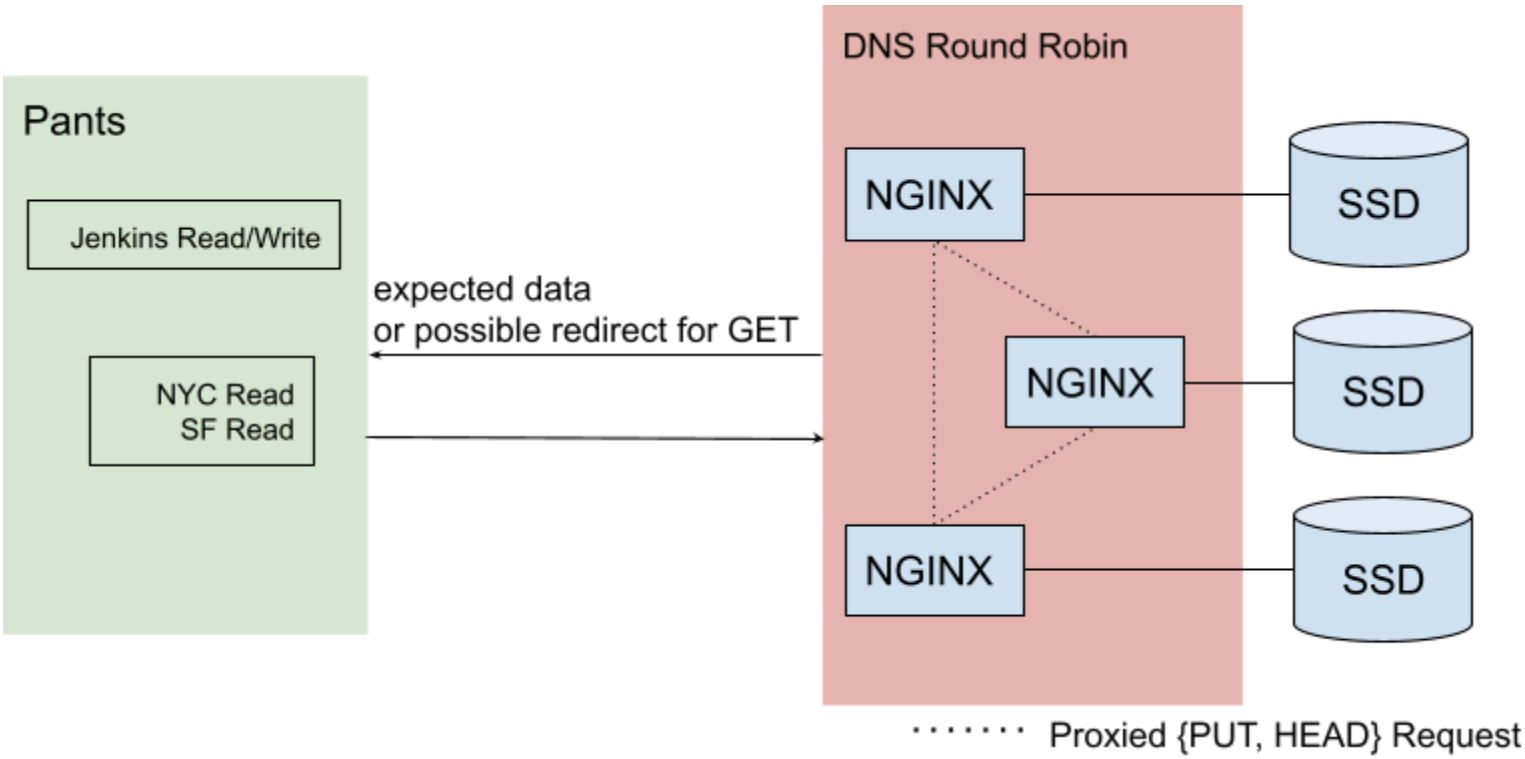
Foursquare Artifact Cache Topology



Nginx Configuration

- Replace each `${SERVERN}` with your servers (as many as you like).
- Point pants at `http://your-dns-rr` where `your-dns-rr` answers with A records for your proxy nodes. I think you can also get away with only RRing to a subset of nodes, but this hasn't been tested and I know of no good reason not to use all of the proxy nodes.

For HEADs and PUTs we have to proxy the traffic because Pants cannot follow redirects for those verbs. PUTs are low traffic and HEADs aren't too expensive to proxy. If a GET is sent to the wrong server then redirect.

Since we are using SSDs, let's try omitting varnish for simplicity of design and then it add it later if we need it.

Each nginx node represents two endpoints: one for proxying to the right shard, and one for serving requests.

- The **proxy node** (port 80)
  - Use consistent hashing to figure out which node in the fleet is the correct shard.
  - Proxy (stream) all requests through to that node.
  - Set a header (the hostname of the proxy node) to make it possible for the upstream backend node to return a redirect if necessary.
- The **backend node** (port 8080)
  - Serve HEAD and PUT requests directly using normal DAV operations on a directory.
  - For GET requests where the proxy node is the same as the backend node, serve the request directly.
  - For GET requests where the proxy node is not the same as the backend node, return a redirect to the current backend node. Pants will follow this redirect, and we get to avoid proxying the large data stream through the proxy node (pants fetches it directly after following the redirect).
  - For GET requests where the proxy node header isn't set, assume we are satisfying a redirect and serve the request directly.

With DNS RR, we get easy load balancing among the proxy nodes while maintaining sharding for the highest throughput access (lots of GETs).