admin++, what root never told you

one or two lessons I learned while moving beyond root to Systems Administration

Ron Gorodetzky
ron@parktree.net
ron@digg.com

who am I

- Avid GNU/Linux user for more than 6 years
  - managed several small servers in that time
- Co-founder of Revision3
  - sole IT member for the first year and a half
- Senior Systems Administrator at Digg
  - managed root since before first public release
- Managing the volume that both these companies experience was completely new to me

what problems do these companies have to solve

- Revision3
  - we produce and distribute audio/video content (pod/vidcasts) (e.g. diggnation, indigital, pixelperfect)
  - over 100,000 downloads per week
  - file sizes ranging from 150-600MB
  - most downloads happen on the day of a release (big bandwidth spikes)
  - the website is popular, but needs only a small fraction of the total resources needed to deliver content

what problems do these companies have to solve (continued)

- Digg
  - user driven social content website. Content is submitted, voted on, and commented on by the community
  - lots and Lots of hits/pageviews/visitors
  - realtime-ish data (when you digg it is mostly-immediately reflected on the site)
  - unlike revision3 bandwidth usage is relatively low

sound advice

- Web 2.0 is here, surely web 3.0 is around the corner
- Everyone is building a new application, websites, and organizations that need infrastructures
- So, get some sleep, exercise, and be prepared for lots of work ahead
- Your one server isn't going to hold up by itself forever, start preparing now
- Read everything related to infrastructures, a lot of it is really cool

what having root should have taught you

- How to read documentation
- How to install software
- How to configure software
- How to read documentation
- How to troubleshoot some server issues
- How to setup simple networking
- How to read documentation

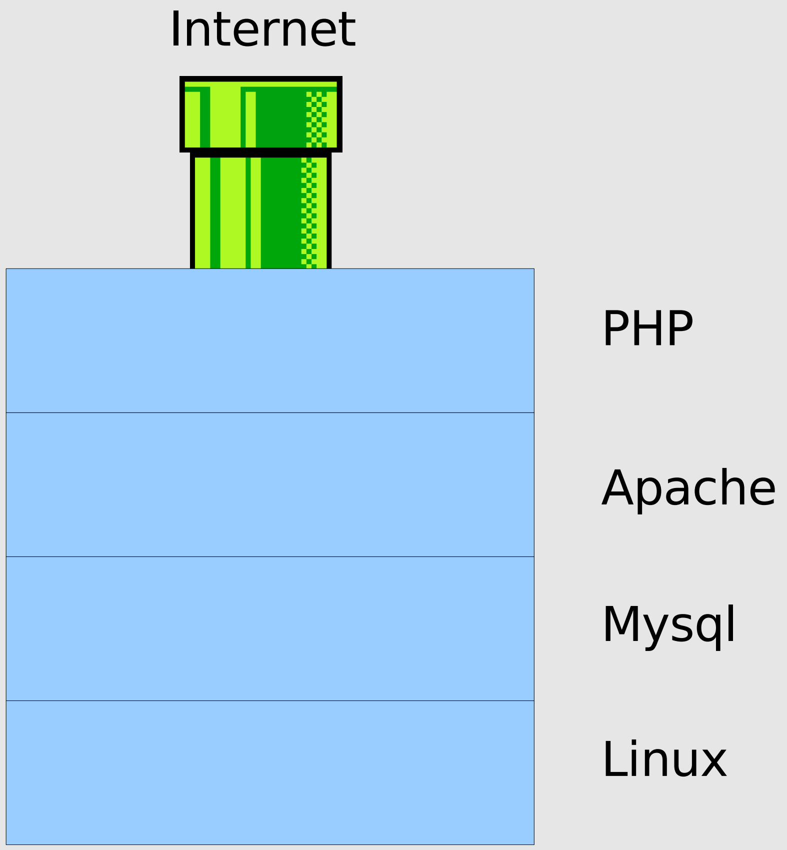
other things you hopefully learned

- Not to take the inner workings for granted
  - file system limitations
  - swap vs ram
- At least a little C
  - pointers, especially of the NULL variety
- Scripting
  - shell, python, perl

so you built your first web app

- You probably used a popular stack
  - LAMP (PHP, Perl, Python)
  - Ruby on Rails
  - Turbogears / Pylons
- You share it with your friends
- Everyone thinks you're a genius
- You put up some ads, charge some money, and try to make it big

and your organization probably looks like this



Easy!

at some point you'll need to grow

- You'll want a backup machine for redundancy
- You'll want a development machine for testing
- You'll want to serve more traffic (i.e. Scale)
  - first step is usually putting the database on a separate server from the front end

who's had some good ideas about scaling

- http://danga.com/words/2005_oscon/oscon-2005.pdf
  - how to scale your web application
  - everyone should read this
- http://labs.google.com/papers/
  - especially mapreduce and google file system

Hopefully these will help you to start thinking about your infrastructure in novel ways that you hadn't considered before.

scaling basics

- Cache!
- Spread data/resource usage across multiple machines (e.g. separate webserver and db)
- Often the basic frameworks won't be enough on their own
  - memcache
  - mogilefs
  - load balancing
- Basic scaling principles aren't that hard and can be implemented entirely on one machine so try to be aware of them and start to deal with it early.

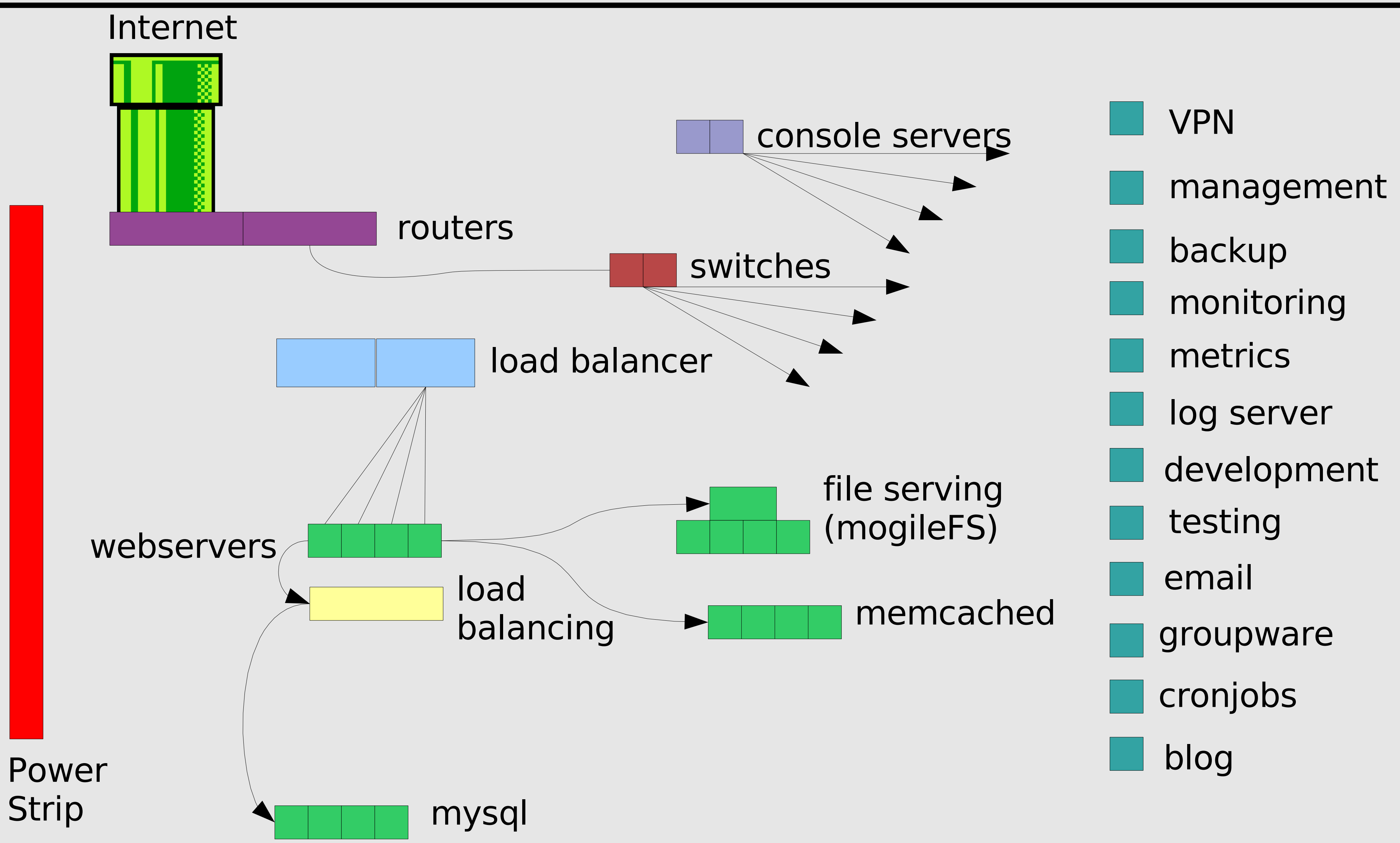
these machines have to go somewhere

- You'll need dedicated servers
- You can get more managed servers
  - let the provider deal with network, server maintenance, etc.
- Colocation
  - take your servers there, but let them deal with primary network issues
- Datacenter
  - get yourself a space in a datacenter and do everything yourself

or outsource

- Distributing large volumes of data is hard
  - lots of bandwidth
  - lots of computers
  - great candidate for outsourcing (e.g. cachefly, bitgravity, akamai)
- Doing lots of computations or storing lots of data can also be hard
  - clients contribute resources (e.g. folding@home, store state in cookies not the DB)
  - Amazon EC2 and S3 (I imagine more of these types of services will pop up eventually)
- This keeps the # of resources you need to manage lower

in a datacenter your infrastructure looks more like this



now you need to manage it all

- It's all remote
  - equipment can't hear your threats of violence
  - console servers and emergency modem connection are your friends
- You'll need to use “enterprise” networking equipment
  - bigger switches
    - vlans are pretty cool
  - bigger routers
    - BGP, peering, fiber
- Remotely managed power strips
  - control power to individual pieces of equipment (fun!)

servers need even more hand holding

- You need to be able to interact with bad servers remotely
  - power and console
  - IPMI (?)
- Server network reinstallation
  - fai, kickstart, systemimager
- Server configuration
- Automate everything early and often !!
- Servers will die and everything should still work!!!

what are some server management tools

- CFEngine
  - http://www.cfengine.org/
- Puppet
  - http://reductivelabs.com/projects/puppet/index.html
- Bcfg2
  - http://www.bcfg2.org
- ISConf
  - http://trac.t7a.org/isconf/
  - from the http://www.infrastructures.org people

some notes about these tools

- They each approach the problem of server configuration in different ways
- There is no one Right Way (tm)
- Each organization has their own system
- So study all the options out there and use the one that fits the infrastructure model in your head

some datacenter advice

- Don't waste time making your own cables!
- Get wide, deep cable management
- Wire all possible ports
  - with ethernet especially useful with vlan magic
- Decide on a labeling strategy
  - but, no label is better than a wrong label!
- In a datacenter you're really paying for power, not space, so feel free to resist the urge to pack everything too tightly

monitoring and metrics

- You need to know when things go wrong
  - SNMP, nagios, monit, openNMS
- You also want to track resource usage
  - SNMP, cacti, ganglia, mrtg, openNMS
- Don't forget to monitor the monitoring server!
  - not getting pages is great, but can be the sign of a problem

being investor friendly even if you have none

- Don't spend too much money too fast
  - FLOSS is of course great for this
  - smart efficient designs help too
  - good news: salaries are almost always more than equipment so don't be afraid to get what you need
- User statistics
  - you'll need to prove you really have the customers you claim
  - web applications compete with other public numbers like alexa and comscore
  - reporters and investors may not believe your internal numbers so you may need to outsource some web metrics to a third party
    - due to privacy concerns you probably shouldn't take this decision lightly

systems for the rest of the organization

- you'll of course need email
  - but you'll probably really need groupware (calendar, contacts, etc.)
- Much of the business world has become accustomed to MS exchange-style groupware
  - should work with mobile devices
  - should allow others to manage executive calendars (e.g. secretaries and office managers)
- Project management
  - bugzilla, trac, wikis

tracking your developments (documenting everything)

- Development
  - you want to track bugs and code releases
  - this means you need conventions for development, branches, and releases
- Operations
  - keep track of all your equipment, connections, and RMAs
    - first spreadsheets/diagrams, then get more sophisticated
  - keep track of changes, code pushes, and scripts

testing and deployment

- When you grow people have less tolerance for things breaking.
  - Myspace breaks this rule (search for “Inside Myspace.com”)
- Have an testing environment that mimics production
- Treat test code the same as you would production
- But load does funny things to code and servers so be prepared for strange failures

the value of going back and doing things right (tm)

- If you don't do things correctly now you'll have to do it later under more pressure and with less time
- There will always be new better methodologies to growing and managing an infrastructure
- Doing things a little better now will give you some breathing room in the future
- Thoughtful software design should be accompanied by thoughtful infrastructure design

questions?

some random stuff for those who don't have questions

- In bash you can't escape a single quote inside of single quotes, so you have to do this:
  - `echo 'the quotes, they'\''re everywhere!'`
- Regular expressions are great, but not always fast. Sometimes processing strings the old fashioned way (by looking for characters of interest one by one) is much much faster. (e.g. log processing, take a look at how Visitors does it)
- Linux was mentioned in the first sketch of the first episode of Comedy Central's Chappelle's Show. This is likely why it was such a success.