CSE 265: System and Network Administration

- Reminder of online syllabus and schedule
- Who has sysadmin experience?
- Today
  - Managing (lots of) desktops
    - Loading, updating, configuring
  - Managing servers
    - Important to lots of people
  - Managing services
    - The reason for most servers

Initial OS and App Installation

- Automation solves many problems
  - Saves time/money; reduces mistakes; ensures uniformity
  - Examples: Solaris JumpStart, Red Hat Kickstart, AutoYaST, Preseed
  - Cloning (ghosting, disk imaging) sometimes an option
- Full automation much better than partial
  - Eliminate prompts in installation scripts
  - Can include automatically notifying people when complete
- Partial automation better than none
  - Needs to be well-documented for consistency

Initial OS and App Installation

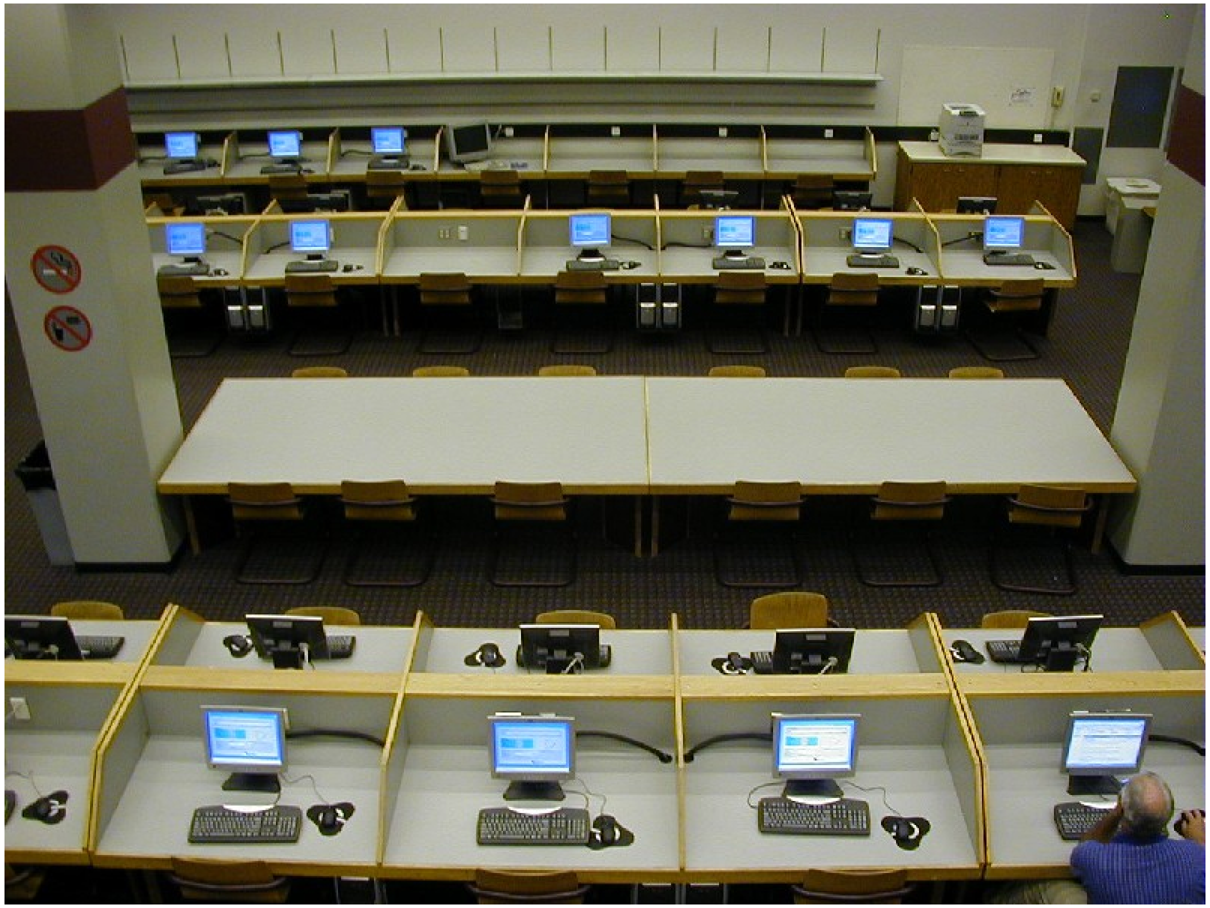
- Automation solves many problems
  - Saves time/money; reduces mistakes; ensures uniformity
  - Examples: Solaris JumpStart, Red Hat Kickstart, AutoYaST, Preseed
  - Cloning (ghosting, disk imaging) sometimes an option
- Full automation much better than partial
  - Eliminate prompts in installation scripts
  - Can include automatically notifying people when complete
- Partial automation better than none
  - Needs to be well-documented for consistency

RHEL installation still takes dozens of actions!

Managing (lots of) Desktops

- Three main sysadmin tasks for workstations
  - Initial loading of system software and applications
  - Updating system software and applications
  - Configuring network parameters
- Need to get all three right
  - Initial load must be consistent across machines
  - Updates must be quick
  - Network configuration best managed centrally
- Solution is automation (for supported platforms)

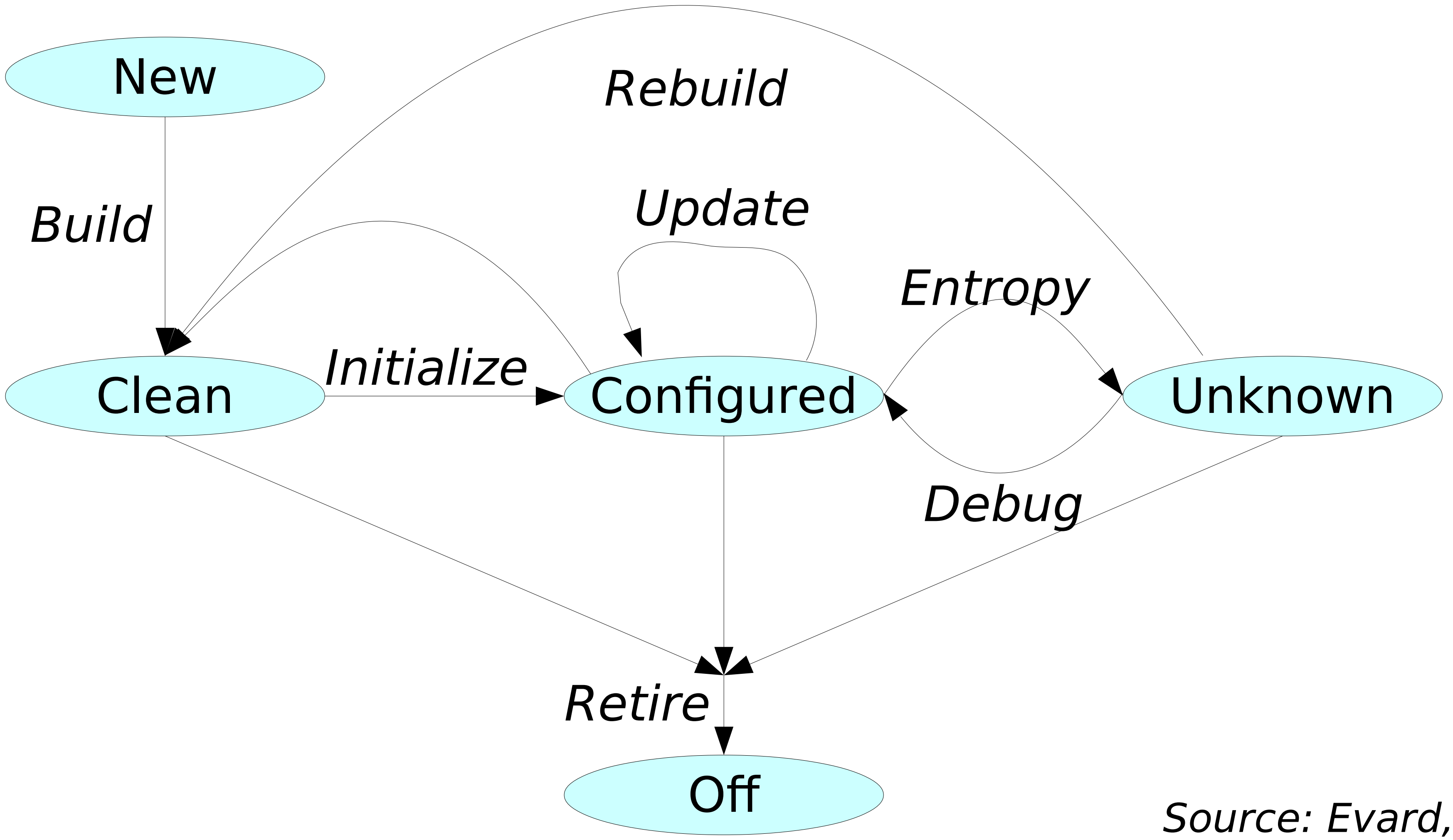
Lots of desktops



Many photos by Mark Miller (Lehigh LTS) in 2003 and 2004.

You really don't want to install, configure, and update lots of machines individually.

Machine Life-Cycle



Source: Evard, 1997

Machine Life-Cycle

- There are five states and many transitions
  - Need to plan for them
- **Computer is only usable in the configured state**
- Want to maximize useful time
  - Minimize useless time
  - Setup and recovery should be fast and efficient --> automation (manual processes are slow and error-prone)
  - Slow (minimize) entropy
    - Restrict root privileges
    - Control where changes can and are made (e.g., 3rd party apps)
- Rebuilding & retiring may require moving data & apps

Use your own installation

- Don't trust the vendor's pre-installed OS
  - Adding apps to a truly clean installation can be easier
  - Their install image can change over time
  - You'll need to re-install eventually
    - Making your re-installation a different configuration
    - You want to be certain that you have everything (drivers, software, etc.) to re-install
  - You may not want or need their special applications and add-ons

Updating system and apps

- Over time, people find
  - New bugs
  - New security holes
  - New applications
- Updates can (and should) be automated, too
- Example automation systems include Linux package updaters like pup/yum and apt

Differences for updates

- Updates are performed on functioning machines
- The machine is already deployed
  - Can't flood network
  - May not have physical access
- Users of host will expect it to work after update
  - Must be extremely careful! Gradual deployment.
- Host may not be in known state
- Host may have live users (requiring downtime)
- Host may be disconnected periodically
- Host may dual-boot (long periods between updates)

Network Configuration

- Network config different from install
  - Values vary by location, rather than OS+apps
- Typical solution is to use DHCP
  - Eliminate time and manual error
    - By sysadmin or user (assigning himself an IP address and/or hostname)
  - More secure (only authorized systems get access)
  - Can assign a particular IP to an individual host
  - Centralized control makes updates and changes easier (e.g., new DNS server)

Managing Servers

- Different from desktop? Yes!
  - May serve tens, hundreds or many thousands of users
  - Requires reliability and high uptime
  - Requires tighter security
  - Often expected to last longer
  - Extra cost is amortized across users, life span

Managing Servers (cont.)

- Servers typically have
  - Different OS configurations than desktops
  - Deployment within the data center
  - Maintenance contracts
  - Disk backup systems
  - Better remote access

Server Hardware

- Buy server hardware for servers
  - More internal space
  - More CPU performance
  - High performance I/O (both disk and network)
  - More upgrade options
  - Rack mountable/optimized
- Use vendors known for reliability
  - Your time is valuable

Do servers really cost more?

- Typical vendor has three product lines
  - Home
    - Absolute cheapest purchase price
    - OEM components change often
  - Business
    - Longer life, reduced TCO
    - Fewer component changes
  - Server
    - Lowest cost per performance metric
    - Easier to service components and design

Inspiron

Optiplex

Precision

Maintenance contracts, spare parts

- All machines eventually break!
- Vendors have variety of service contracts
  - On-site with 4-hour, 12-hour, or next-day response
  - Customer-purchased spare parts get replaced when used
- How to select maintenance contract? Determine needs.
  - **Non-critical hosts**: next-day or two-day response time is likely reasonable, or perhaps no contract
  - **Large groups of similar hosts**: use spares approach
  - **Controlled model**: only use a small set of distinct technologies so that few spare part kits needed
  - **Critical host**: stock failure-prone and interchangeable parts (power supplies, hard drives); get same-day contract for remainder
  - **Large variety of models from same vendor**: sufficiently large sites may opt for a contract with an on-site technician

Data Backups

- Servers are often unique with critical data that must be backed up
  - Clients are often not backed up (most data is on server)
  - Consider separate administrative network
    - Might want to keep bandwidth-hungry backup jobs off of production network
    - Provides alternate access during network problems
    - Requires additional NICs, cabling, switches
  - (More details later in semester)

Servers in the Data Center

- Servers should be located in data centers
- Data centers provide
  - Proper power (enough power, conditioned, UPS, maybe generator)
  - Fire protection/suppression
  - Networking
  - Sufficient air conditioning (climate controlled)
  - Physical security

Remote Administration

- Data centers are expensive, and thus often cramped, cold, noisy, and may be distant from admin office
- Servers should not require physical presence at a console
- Typical solution is a console server
  - Eliminate need for keyboard and screen
  - Can see booting, can send special keystrokes
  - Access to console server can be remote (e.g., ssh, rdesktop)
- Power cycling provided by remote-access power-strips
- Media insertion & hardware servicing are still problems

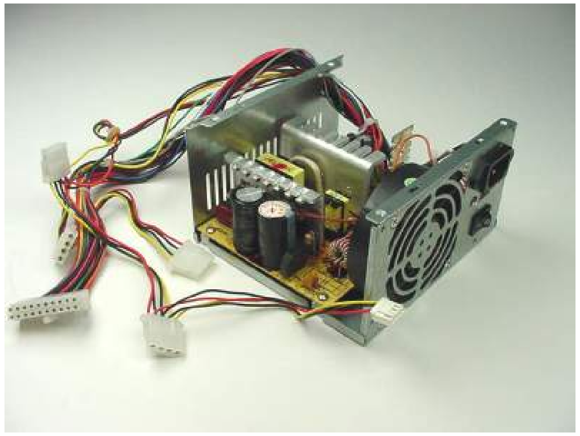
Mirrored Root Disks

- Disk drives fail!
- Often useful to consider RAID for data integrity
- The main system disk is often the most difficult to replace
- Software RAID often comes with the OS for “free”; hardware RAID is getting cheaper
- Two approaches for mirrored root disks:
  - Two disks; copy from the working disk to the clone at regular intervals (e.g., once a night)
  - Use hardware or software RAID to keep both in sync
- RAID disks still need to be backed up

Why?

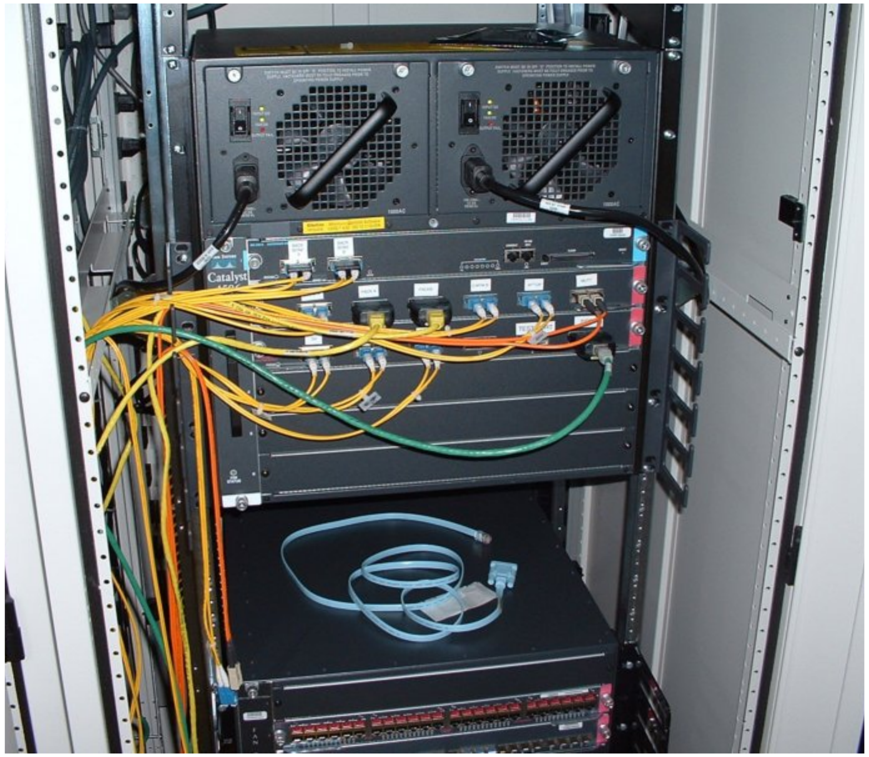
Redundant Power Supplies

- Power supplies 2nd most failure-prone part
- Ideally, servers should have redundant power supplies
  - Means the server will still operate if one power supply fails
  - Should have separate power cords
  - Should draw power from different sources (e.g., separate UPSes)



Router with dual power supplies

- This is a Cisco 4506 switch that serves as one of the backbone switches for Lehigh's network.
- Fiber (or copper if nearby) travels from this switch to each router on campus. An identical backbone switch is located in EWFMB.
- It has redundant power supplies, one connected to a UPS and one connected directly to commercial power.

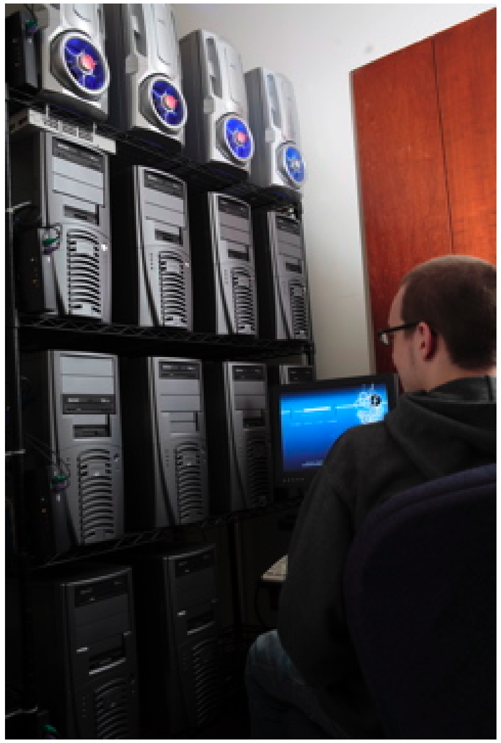


Hot-swap Components

- Redundant components should be hot-swappable
  - New components can be added without downtime
  - Failed components can be replaced without outage
- Hot-swap components increases cost
  - But consider cost of downtime
- Always check
  - Does OS fully support hot-swapping components?
  - What parts are not hot-swappable?
  - How long/severe is the service interruption?

Alternatives to Expensive Servers

- Server appliances
  - Dedicated-purpose, already optimized
  - Examples: file servers, web servers, email, DNS, routers, etc.
- Many inexpensive machines
  - Common approach for web services
    - Google, Hotmail, Yahoo, etc.
  - Use full redundancy to counter unreliability
  - Can be useful (but need to consider total costs, e.g., support and maintenance, not just purchase price)



Managing Services

- Services distinguish a structured computing environment from a bunch of standalone computers
- Larger groups are typically linked by shared services that ease communication and optimize resources
- Typical environments have <u>many</u> services
  - DNS, email, authentication, networking, printing
  - Remote access, license servers, DHCP, software repositories, backup services, Internet access, file service
- Providing a service means
  - Not just putting together hardware and software
  - Making service reliable
  - Scaling the service
  - Monitoring, maintaining, and supporting the service

Designing a solid service

- Get customer requirements
  - Reason for service
    - How service will be used
    - Features needed vs. desired
    - Level of reliability required
    - Justifies budget level
  - Define a service level agreement (SLA)
    - Enumerates services
    - Defines level of support provided
    - Response time commitments for various kinds of problems
  - Estimate satisfaction from demos or small usability trials

Designing a solid service

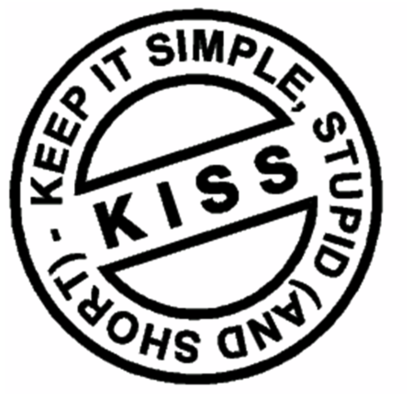
- Get operational requirements
  - What other services does it depend on?
    - Only services/systems built to same standards or higher
    - Integration with existing authentication or directory services?
  - How will the service be administered?
  - Will the service scale for growth in usage or data?
  - How is it upgraded? Will it require touching each desktop?
  - Consider high-availability or redundant hardware
  - Consider network impact and performance for remote users
- Revisit budget after considering operational concerns

Designing a solid service

- Consider an open architecture
  - E.g., open protocols and open file formats
  - Proprietary protocols and formats can be changed, may cause dependent systems/vendors to become incompatible
  - Beware of vendors who "embrace and extend" so that claims can be made for standards support, while not providing customer interoperability
  - Open protocols allow different parties to select client vs. server portions separately
  - Open protocols change slowly, typically in upward compatible ways, giving maximum product choice
  - No need for protocol gateways (another system/service)

Designing a solid service

- Favor simplicity
  - Simple systems are more reliable, easier to maintain, and less expensive
  - Typically a features vs. reliability trade-off
- Take advantage of vendor relationships
  - Provide recommendations for standard services
  - Let multiple vendors compete for your business
  - Understand where the product is going



Designing a solid service

- Machine independence
  - Clients should access service using generic name
    - e.g., www, calendar, pop, imap, etc.
  - Moving services to different machines becomes invisible to users
  - Consider (right from the start) what it will take to move the service to a new machine
- Supportive environment
  - Data center provides power, AC, security, networking
  - Only rely on systems/services also found in data center (within protected environment)
    - e.g., don't depend on a service from a PC in the closet

Designing a solid service

- Reliability
  - Build on reliable hardware
  - Exploit redundancy when available
    - Plug redundant power supply into different UPS on different circuit
  - Components of service should be tightly coupled

    Why?

  - Make service as simple as possible
  - Independent services on separate machines, when possible

Designing a solid service

- Reliability
  - Build on reliable hardware
  - Exploit redundancy when available
    - Plug redundant power supply into different UPS on different circuit
  - Components of service should be tightly coupled
    - Reduce single points of failure
      - e.g., all on same power circuit, network switch, etc.
    - Includes dependent services
      - e.g., authentication, authorization, DNS, etc.
  - Make service as simple as possible
  - Independent services on separate machines, when possible

Designing a solid service

- Restrict access
  - Customers should not need physical access to servers
    - Fewer people -> more stable, more resources, more secure
  - Eliminate any unnecessary services on server (security)
- Centralization and standards
  - Building a service = centralizing management of service
  - May be desirable to standardize the service and centralize within the organization as well
    - Makes support easier, reducing training costs
    - Eliminates redundant resources

RESTRICTED ACCESS
EMPLOYEES
AND
AUTHORIZED PERSONNEL ONLY
Valid identification is required to enter this facility.
An employee must authorize a visitor before entering the facility.
Identification must be visible at all times while in this facility.
BRADY. BRADYID.COM

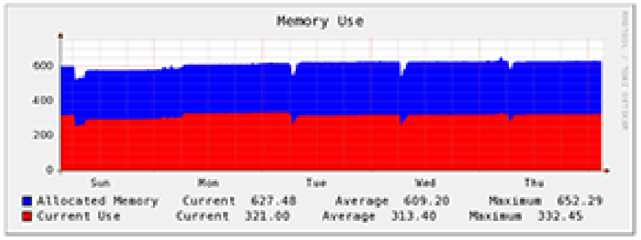
Designing a solid service

- Performance
  - If a service is deployed, but slow, it is **unsuccessful**
  - Need to build in the ability to scale
    - Can't afford to build servers for service every year
    - Need to understand how the service can be split across multiple machines if needed
  - Estimate capacity required for production (and get room for growth)
  - First impression of user base is very difficult to correct
  - When choosing hardware, consider whether service is likely
    - Disk I/O, memory, or network limited

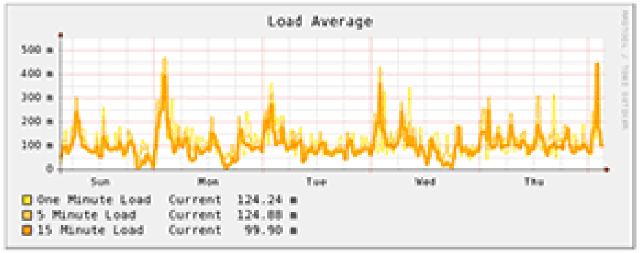
Designing a solid service

- Monitoring
  - Helpdesk or front-line support must be automatically alerted to problems
  - Customers that notice major problems before sysadmins are getting poor service
  - Need to monitor for capacity planning as well
- Service roll-out
  - First impressions
    - Have all documentation available
    - Helpdesk fully trained
    - Use slow roll-out

Memory Use



Load Average


Load Average
500 m
400 m
300 m
200 m
100 m
Sun
Mon
Tue
Wed
Thu
🟨 One Minute Load Current 124.24 m
🟧 5 Minute Load Current 124.88 m
🟧 15 Minute Load Current 99.90 m