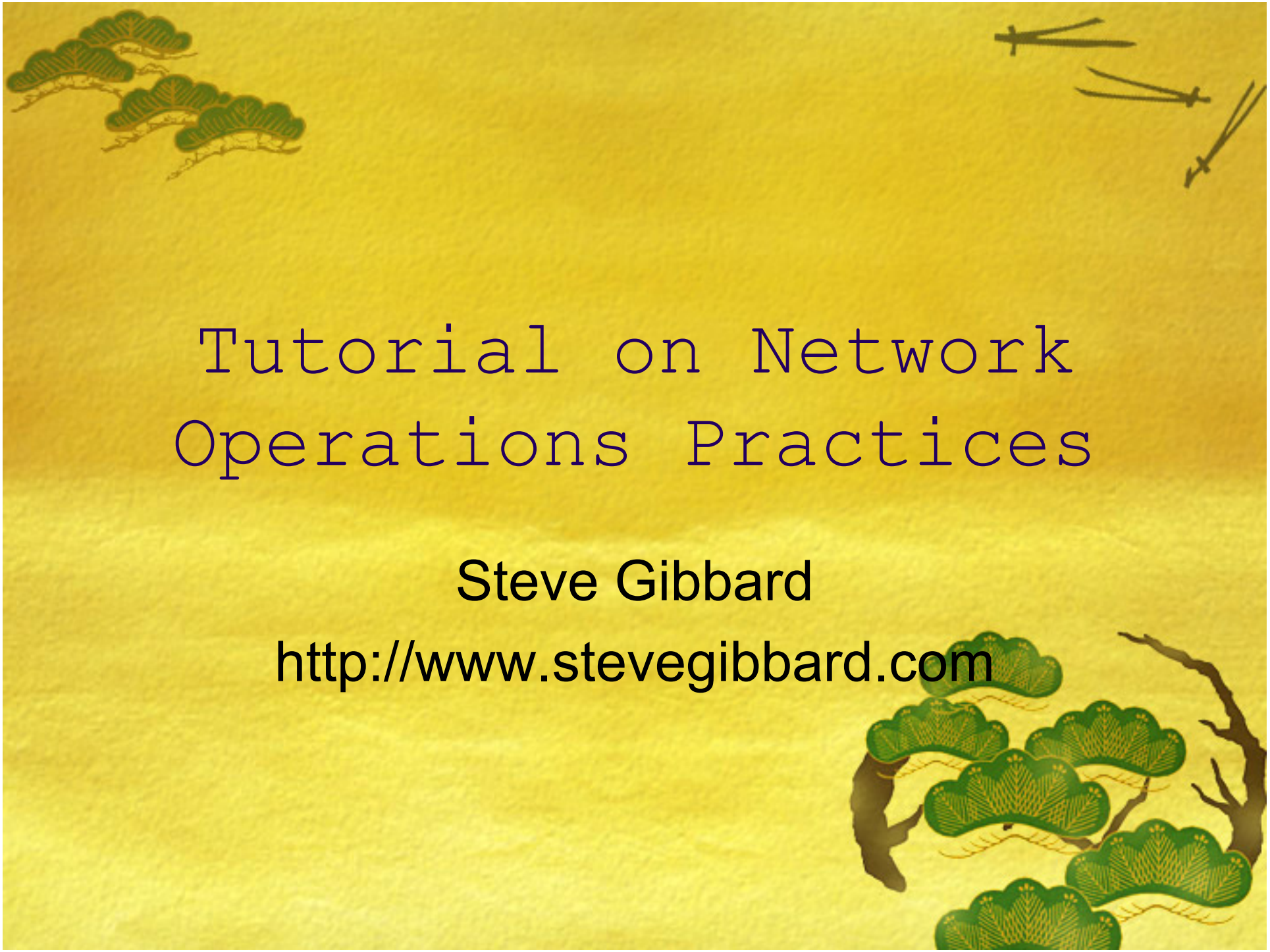
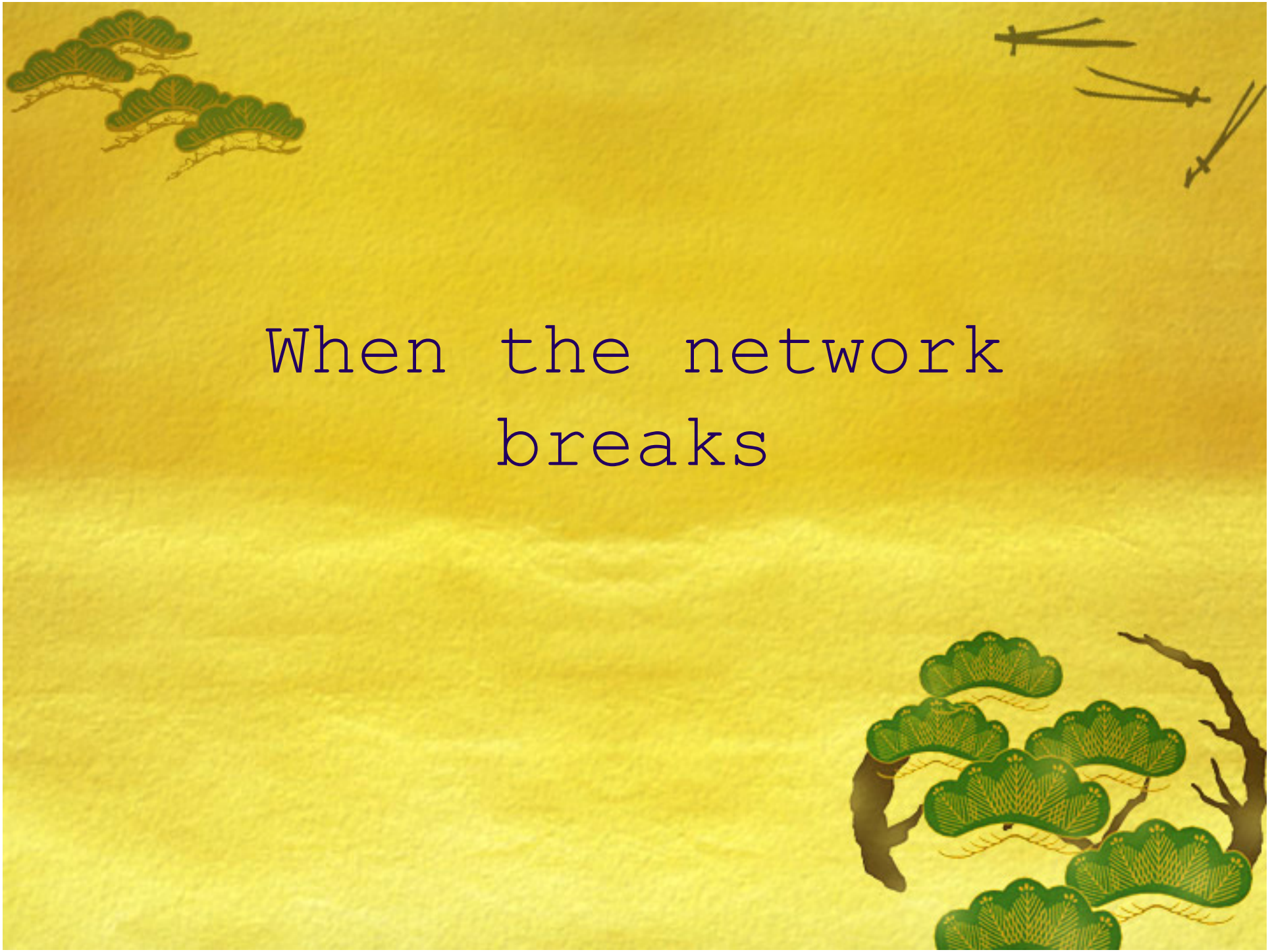

Tutorial on Network Operations Practices
Steve Gibbard
http://www.stevegibbard.com

Introduction

- What are we covering?
  - How to maintain your network.
    - What to do when it breaks.
    - How to manage changes.
    - How to keep your network from breaking.
    - Documentation.
    - External Communication.
  - We're not covering specific router or systems configurations.
    - Lots of other tutorials and workshops cover those.
  - Mostly, good operational practices mean resisting the urge to tinker.

Why is this important?

- Why are good operational practices important?
  - They keep your network running smoothly, which is good for your customers.
  - They keep your life from being interrupted, which is good for you.


When the network breaks

When your network breaks

- You need to restore service now.
  - Your customers expect it.
  - Customers will claim to be “losing millions of dollars an hour.”
- Follow your procedures.
- Don’t panic.
- You don’t need a permanent fix right away.

Prioritization

- What services do you care most about?
- What sorts of customer requests get high priority?
- Does your night shift NOC person know that?
- Separate request-types into different priority levels.
  - Document the priority levels.
  - Document your procedures for different priorities.

Example priority system

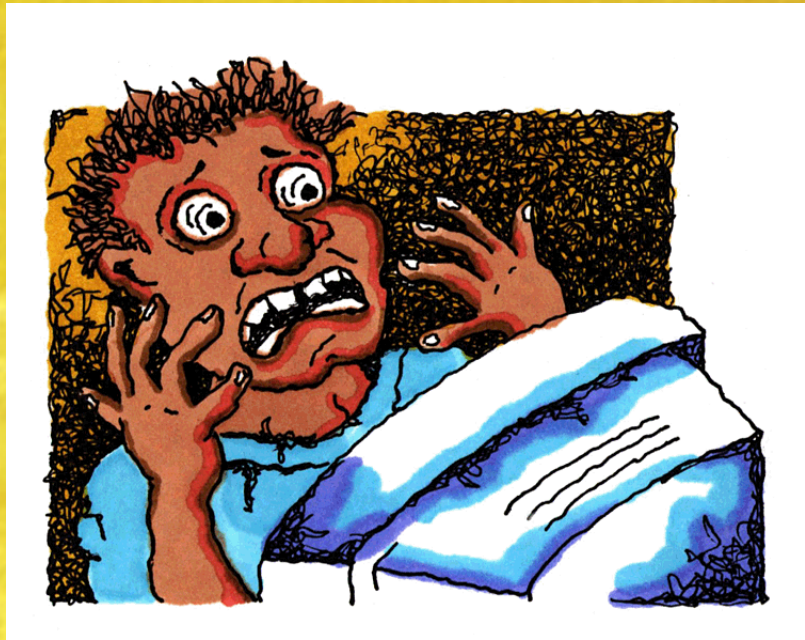
- Priority 1: Problem affecting more than ten customers.
  - Wake up the on-call person.
  - On-call person should respond within 30 minutes.
- Priority 2: Problem affecting less than ten customers.
  - Don’t page on-call.
  - Fix the problem on the next business day.
- Priority 3: Customer change requests.
  - Don’t bother anybody right away.
  - Change within three business days.

Paging/Escalation

- What happens when there's an alert?
  - Do you have a NOC with judgment, or an auto-pager?
  - Can your NOC fix it?
  - Do they have to page somebody else?
  - If paged, do you fix it yourself or talk NOC through fixing it?
- Generating too many alerts causes them to get ignored.
- Getting woken up about stuff that doesn't matter is bad.

Don't panic

- It's the middle of the night. You're tired.
- It's tempting to start changing things.
  - You'll feel like you're doing something.
  - Don't!
- A leading cause of network outages is network engineers.
  - If you try to fix a problem before you understand it, you'll probably make it worse.



What can you do?

- Somebody should be in charge.
- Don't try for a permanent fix.
- Find out what's down.
- Is there redundancy?
  - Turn off the broken component. Watch the service come back up. Go back to sleep.
- Broken non-redundant hardware:
  - Will a reboot fix it?
  - Replace the broken components with spares.
  - Copy your configurations exactly. Don't introduce new changes.

What can you do?

- Recent changes gone wrong:
  - Network engineers are a leading cause of network outages.
  - Back out the changes. Restore the old configurations. Use the back-out procedure from your change plan.
  - Don’t be inventive. Just get things back to a known-stable configuration.

What can you do?

- Mystery problems:
  - It was working. We didn't touch anything. All the pieces seem ok.
  - What are the symptoms? Do they tell you anything?
  - Escalate.
  - Involve vendors.
  - How badly do you need the misbehaving components?
  - What's the minimum stable configuration you can get to?

When you can't fix it

- What if you can't fix it?
- You need to build something new in a hurry.
- You can only use components you already have.
- Still, spend some time on design. You'll get the time back in the construction process.

When you can't fix it

- The redesign and rebuild approach will cause you several hours of downtime.
- Any problems with your plan will make it take longer.
- What you come up with will probably have to be replaced again soon.
- Sometimes it's your only option, but be sure about that before you "dive in."

Planning

- So, you turned something off or propped something up, and went back to sleep…
- Now it's daytime. It's time for a real fix.
- Your network is running. It's not an emergency.
- Your interim configuration is probably unstable.

It's not an emergency

- Take time to understand the problem and its cause.
- Figure out how you're going to put the network back together.
  - Try to avoid major changes. You had a working configuration before.
  - Can you restore the original configuration?
- Use your change management process.
  - Does your fix need off-line testing?
  - Will it cause downtime?
  - What if it doesn't work?

Failure analysis

- You've had a bad outage, and can't afford another one.
- You're having the same outage over and over again.
- Find out why.
  - Does the same component break repeatedly?
  - Are there problems with the network architecture?
  - Is it a mystery?

Mystery failures

- Collect what information you can.
  - What does the network look like when it's broken?
  - Is there other data that would point to a cause?
  - Does it happen at the same time every day?
- Problems you can see are easier to solve.
- Is there log data?
- What else happened at the same time?
- What *could* cause that sort of issue? Can you test hypotheses?
- Don't be afraid to ask for help.

Mystery failure stories

- My stories.
  - A BGP peering session was resetting daily.
    - The peer was threatening to turn off peering.
    - Our configuration was identical to our working configurations.
    - The peer said their configuration was known-working too.
    - The hold time was shorter than on other sessions.
    - Was the peering switch freezing for long enough to expire the hold time?
    - What else happened at that time?
- Audience stories.

Managing changes

Managing changes

- Sometimes you have to make changes.
  - Routine changes are changes you make regularly.
  - Non-routine changes are special cases. These are "Real changes."
- Don't make changes when you don't have to.

Geeks like to take stuff apart

- Geeks like to take stuff apart.
  - Taking your network apart and putting it back together is a really good way to learn how your network works.
  - Unfortunately, it's not good for your network.
- Your job is to to operate a stable network.
  - Avoid doing things "just because it would be cool."
  - Plan and think through network changes, network architecture, etc.

Routine changes

- Document procedures and follow them.
  - You know what worked last time.
  - Don’t make it up as you go along each time.
- Better yet, automate.
  - Software will do the same thing every time.
- Delegate routine changes to lower-level staff.
  - Spend your time on things that require your skills.

Automation example: Peering turn-up command:

Why type:

```
ssh user@router
enable
<password>
conf t
neighbor 192.168.1.5 remote-as 65454
neighbor 192.168.1.5 peer-group PEER
neighbor 192.168.1.5 description peer.net #12345
end
write
logout
```

When you could type:

```
peergen sdq 65454 192.168.1.5 peer.net 12345
```

Before making non-routine changes

- Ask questions:
  - Is this change necessary?
  - How will you make the change?
    - What procedures will you follow?
    - What configurations will you paste in?
    - How much downtime?
    - What resources do you need?
    - What might go wrong?

Be pessimistic (or prepared)

- What will you do if something goes wrong?
  - What do you need to check on?
  - What is your back-out plan?
- Have you tested your procedure?
- What assumptions are you making?
- How will you test?
- Have somebody else review the plan.

Scheduling

- How long a window do you need?
- What will be down during that window?
- When will customers accept downtime?
- Are your resources available?
- Do you have time to get stuck there?
- Will your co-workers be annoyed if you need their help?

During and after changes

- Make sure you're comfortable with your plan.
- Tell your NOC.
- Check on required resources.
- Follow the plan.
- Test when you're done, and at intervals.
- If the plan doesn't work:
  - Fixing obvious things on the fly can be ok.
  - If you can't figure it out, don't dig a deeper hole. Back out.

Dilemma: To act or not to act

- UPS fails. Goes into bypass mode.
- UPS thought to be fixed.
- Turning UPS on causes explosion, and blows circuit breaker. Takes large number of customers offline.
- Utility power restored, but no back-up.
- UPS fixed again.
- Without UPS, risk of utility power failure.
- Cutover to UPS shown to be risky.
- What do you do?

Risk assessment

- Sometimes, all your choices are risky.
- Sometimes, you don't know what will happen.
- Or, you think you know what will happen.
- Use judgment. Pick the option you're least uncomfortable with.
- Do cost analysis on potential failures and improvements.

There are known knowns.These are things we know that we know. There are known unknowns. That is to say, there are things that we know we don't know. But there are also unknown unknowns. There are things we don't know we don't know.

-Donald Rumsfeld

More obvious choices

- Important network device loses redundant power supply controller. Chassis needs to be replaced.
- Until chassis is replaced, a UPS failure would cause a 15 minute outage. UPS failures are unlikely, but there's pressure to replace it sooner rather than later.
- An immediate replacement would require a two hour outage.
- Do you replace the device?

Tools

- Good tools make life much easier.
- If you've got more than a few routers, manual changes are a real pain.
- It's better to make a change once and have it happen everywhere.
- Tools don't have to be complex. RANCID clogin/jlogin makes tool development easy.

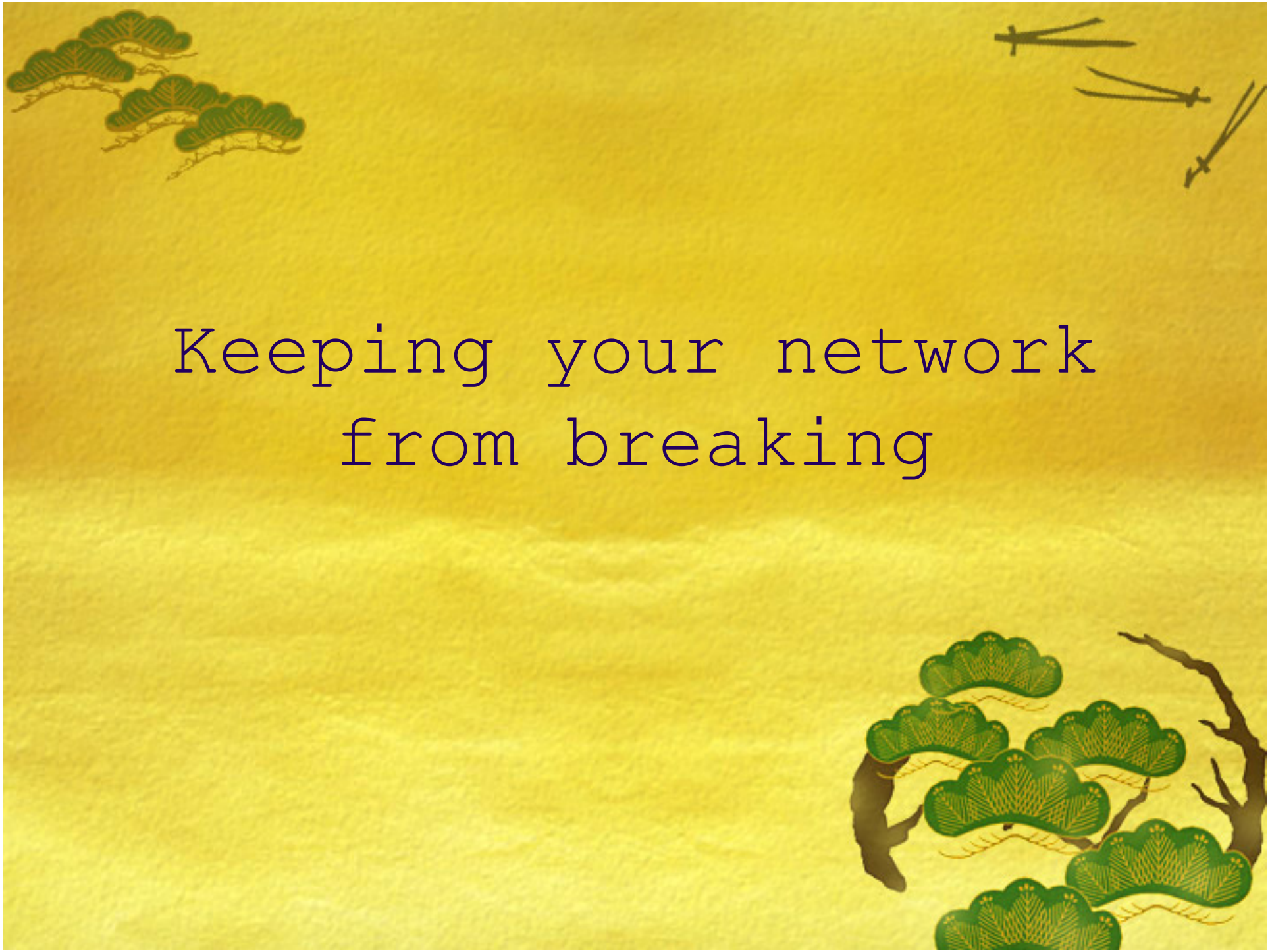
```
#!/bin/sh
UPASS=$1
ENABLEPASS=$2
ROUTERLIST=/usr/local/rancid/tools/rout
    erlist

for router in `cat $ROUTERLIST`
do
    /usr/local/rancid/bin/clogin -c \
          "conf t\r \
          username user pass $UPASS\r \
          enable secret $ENABLEPASS\r \
          end\r \
          write" \
    $router
done
```

Documentation

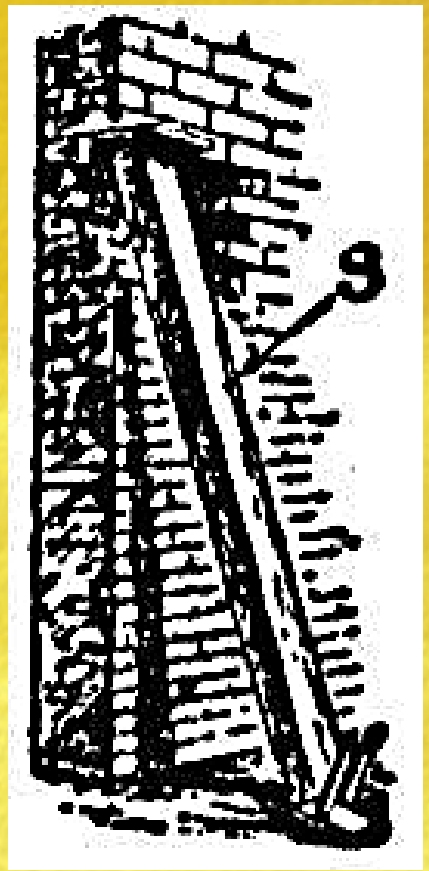
- When you change something, document it.
  - Otherwise, you get woken up when it breaks.
  - If you don't remember the details, you're in real trouble.
  - Or, you might not work there anymore.
- Stick to standard configurations.
  - People will know what to expect.
  - You only have to document them once.
- Documentation on your laptop doesn't help. Use a wiki, or something.

Keeping your network from breaking



Keeping your network from breaking

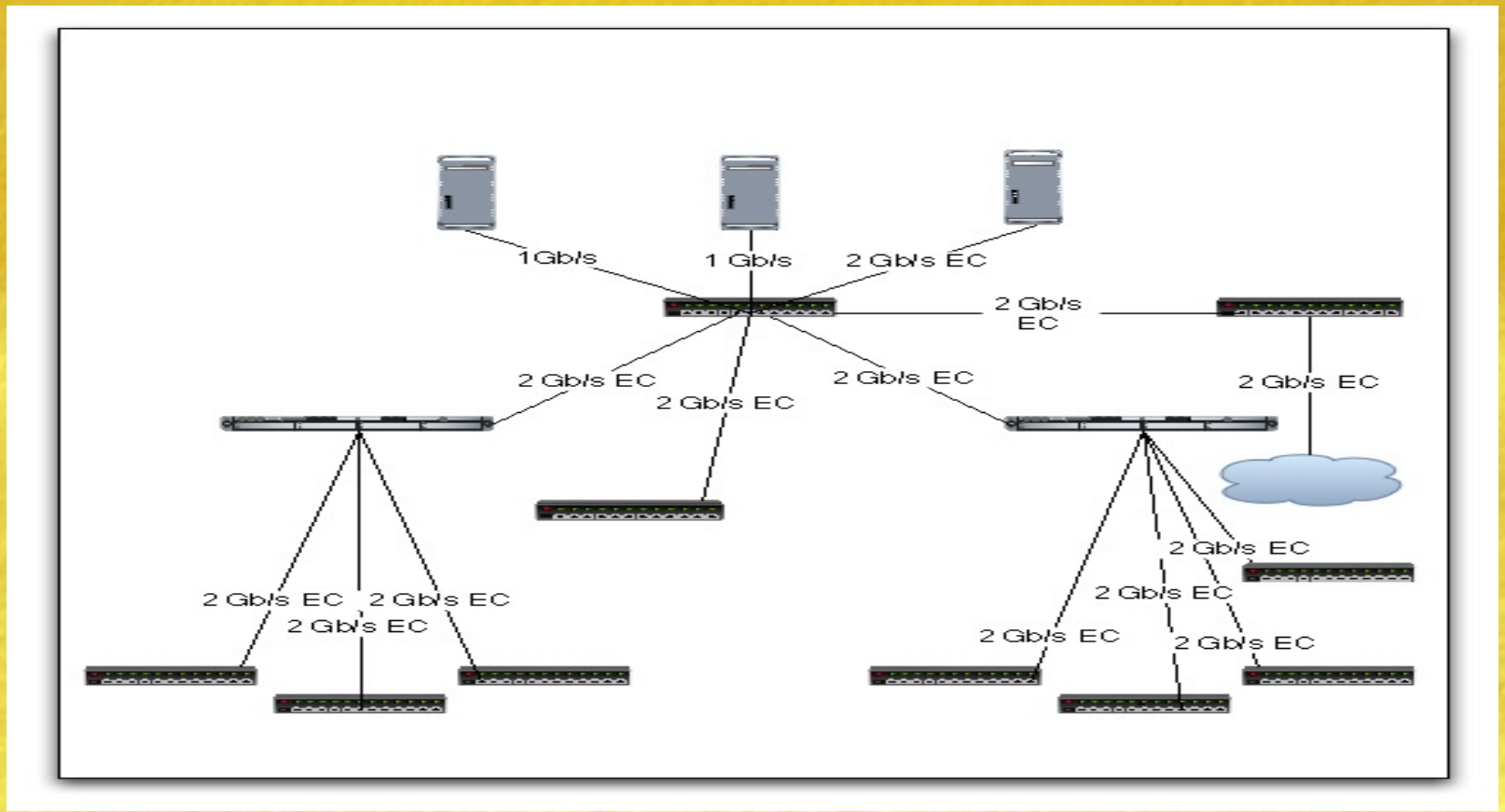
- Architecture: How to design a stable network.
- Procedures: How to operate that network.



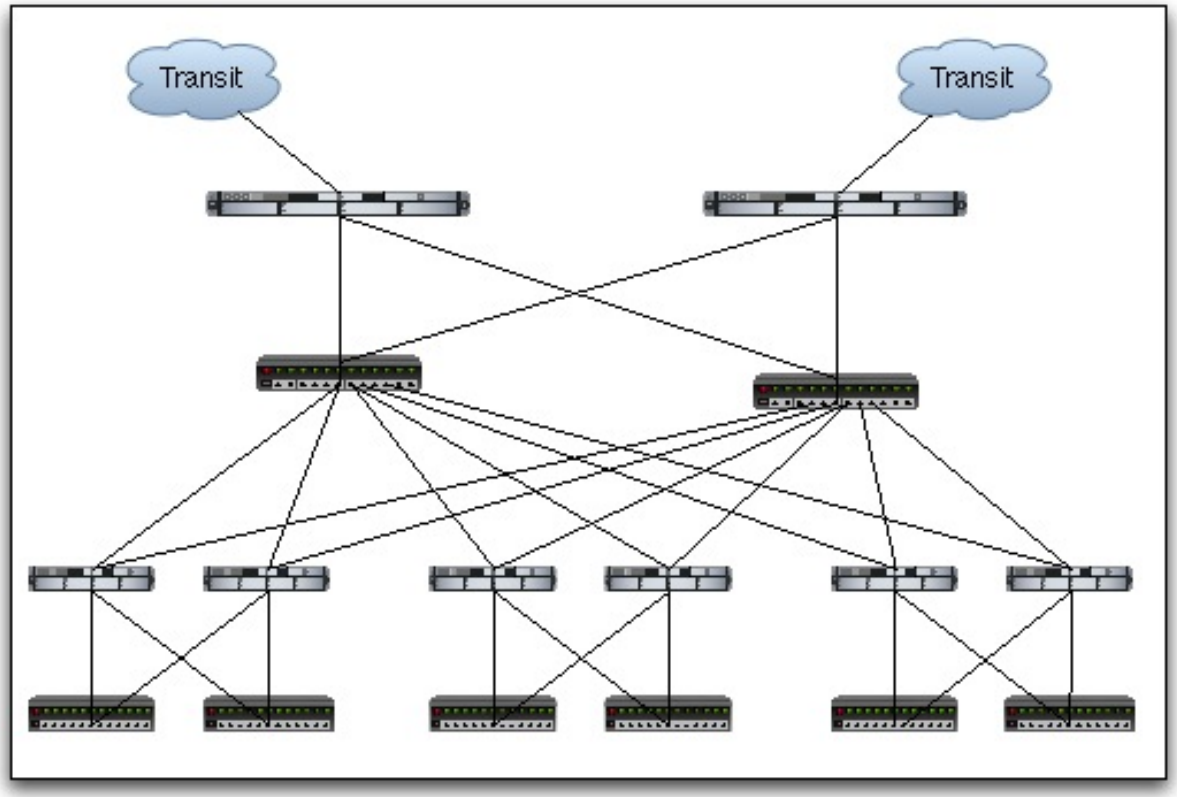
Architecture

- Avoid single points of failure.
  - Ideally, network failures are self-correcting.
  - Otherwise, being able to turn off broken components is nice.
- The “KISS Principle” says, “Keep it Simple, Stupid.”
- Scaling: If you’re successful, your network will need to grow.
  - You don’t need to build the whole thing right away, but don’t make growth require a redesign.

What are the vulnerabilities?



Redundant network design



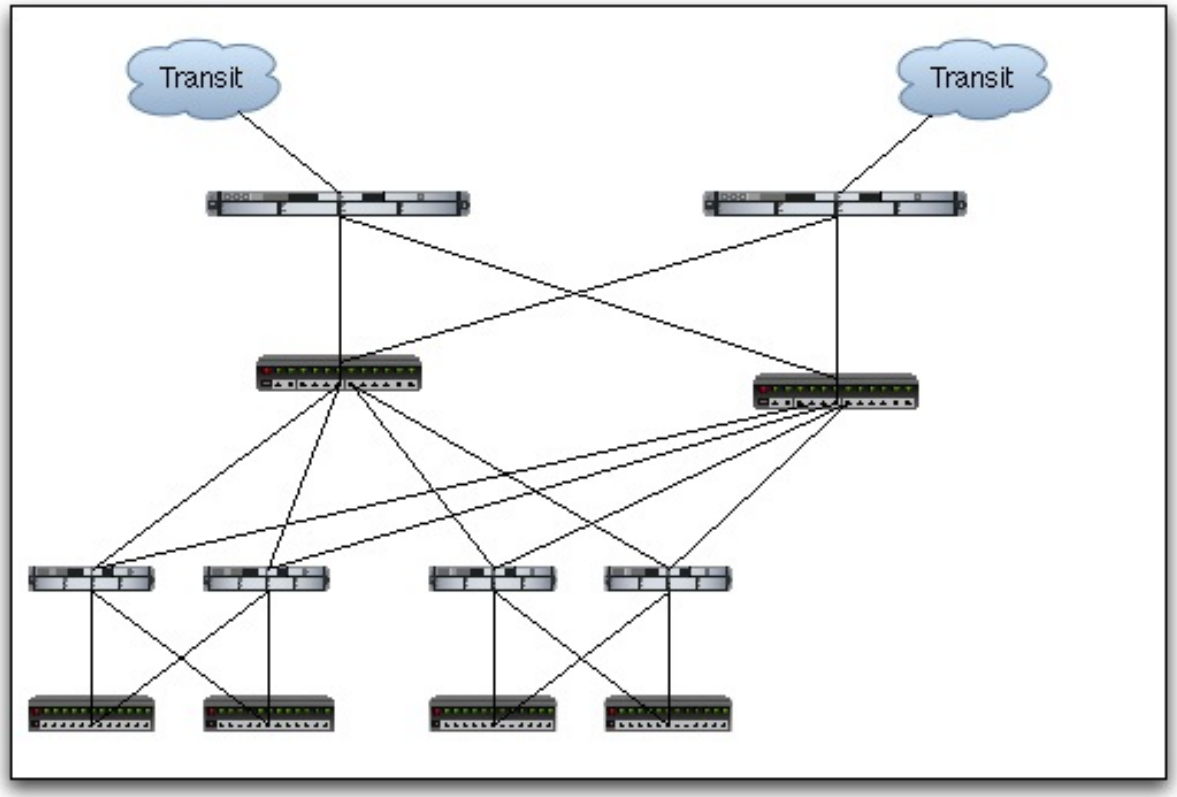
Limits of redundancy

- Redundancy is a statistical game.
  - You can still have bad luck.
  - More pieces are good, but diminishing returns hit quickly.
- Interconnected devices can fail together.
- Redundancy protocols can introduce complexity and cause problems.
- Some vulnerabilities can take out both sides:
  - Software bugs.
  - Load-related problems.
  - Attacks.

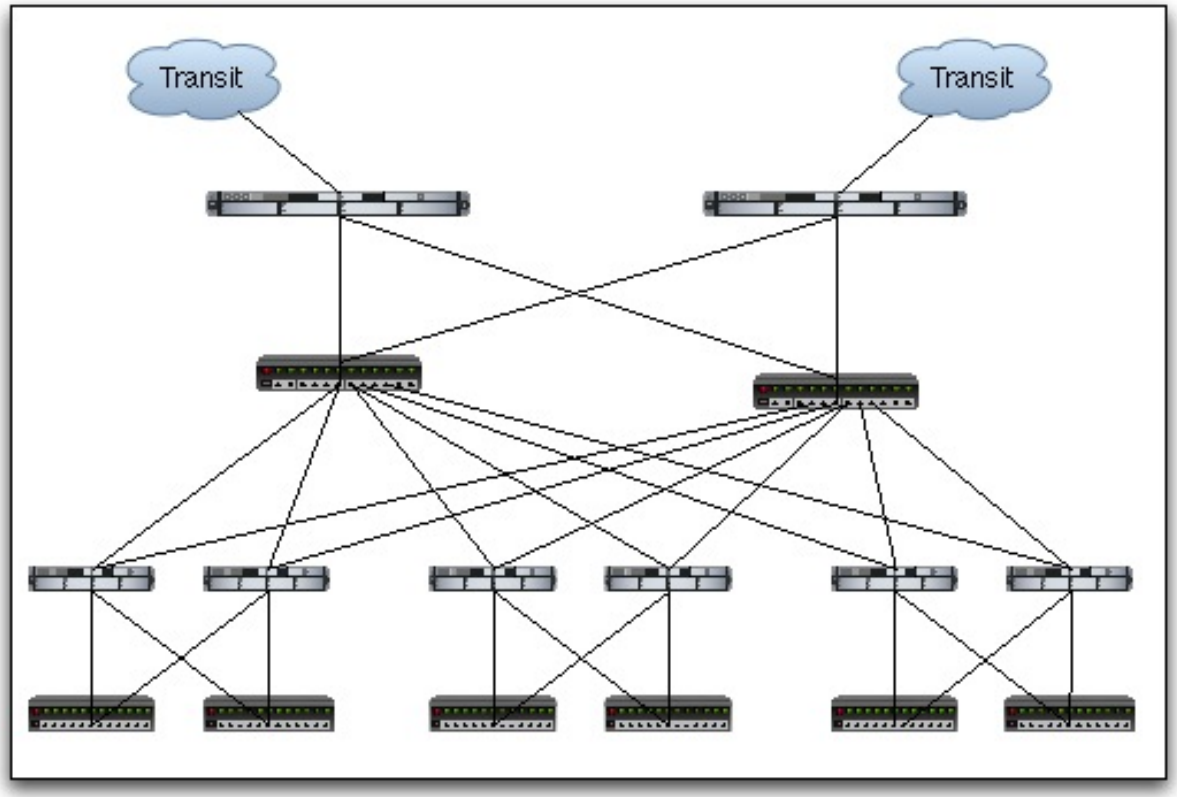
Scaling

- What is scaling?
  - How big does your network need to be now?
  - How big might it need to be eventually?
  - How will you get from here to there?
- How do you design for scalability?
  - Make network out of standard modular “nodes”.
  - Don’t make nodes dependent on each other.
  - Avoid limiting how many nodes can be connected.
  - Use a hierarchy.

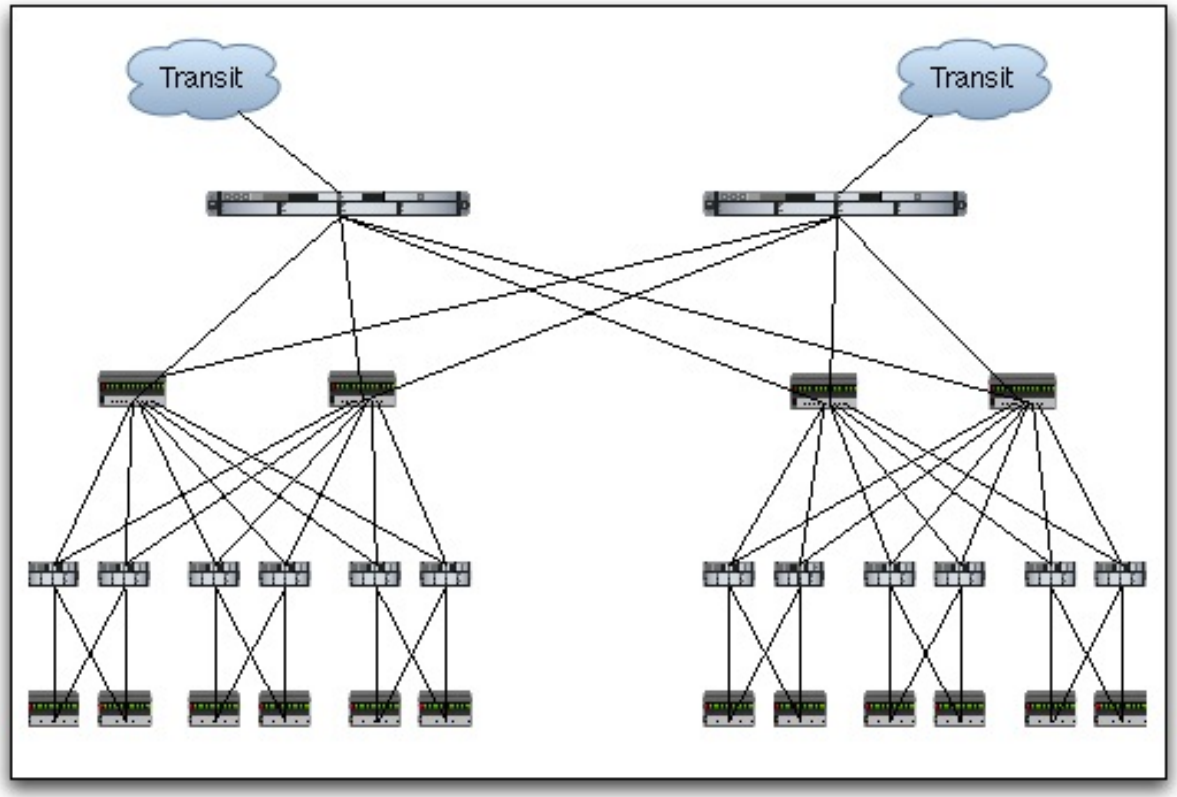
Scalable network design



Somewhat bigger



After scaling



Standardization: Templates

- Standard configurations make life much easier.
- You shouldn't keep reinventing things.
- Knowing how one device is configured should mean knowing how the others are configured.
- Changes can be standardized, too.

```
interface FastEthernet0/1
 description <exch-name> switch
 ip address <exch-addr>
 no ip proxy-arp
 full-duplex
 no cdp enable
!
interface FastEthernet0/0
 description trunk to switch.<loc-
    name>
 no ip address
 no ip proxy-arp
 speed 100
 full-duplex
 no shutdown
!
interface FastEthernet0/0.1
 description <loc-name> subnet
 encapsulation dot1Q 1 native
 ip address <local-ip> 255.255.255.240
```

Procedures

- Think about services, not components.
- Repair components proactively.
- Monitor, but don't over-monitor.
- Prioritize alerts. Don't get woken up when you don't need to.
- Plan network changes carefully.
  - Network engineers are a leading cause of network outages.

Networks provide services

- Your network exists to provide services.
- What services do you care about?
  - Web? Mail? DNS? Other?
- What components are required to provide those services?
  - Routers? Switches? Servers? Circuits? Power?
- Those components are going to break.
- What happens when they break?

Monitoring

- Are both sides of redundant pairs working?
- How are you doing on capacity?
  - Circuits, CPU load, memory, disk space.
  - Network and server performance.
- Don’t over-monitor.
- Prioritize your alerts.
- I’ll say more about handling alerts later.

Be proactive

- Do repairs proactively.
  - If you see a problem, schedule a time to fix it.
  - Use your change management process. Don't cause an outage in the process.
- Think about what can go wrong.
  - Have plans in place to deal with failures.
  - Practice them.
- Forecast capacity. Don't let network growth become an emergency.

Auditing

- Are your configurations standardized?
- Are your redundant pairs really redundant?
  - Do your cables go where you think they do?
  - Are all your routing protocol sessions up?
  - Do you have enough capacity?
- Testing: If you're confident and brave, schedule a window and turn components off.
  - But make sure you know what you're doing first.
- Documentation. Can you find information?

Documentation

- Have information you'll need before you need it:
  - Network diagrams.
  - Service contract numbers.
  - Useful phone numbers.
  - Circuit IDs and end points.
  - Why things were done.
- Where to store documentation:
  - Wikis allow for collaborative editing.
  - Interface descriptions put information right where you need it.
  - Ticket systems show history ("why was this done this way?")

Dealing with customers/peers

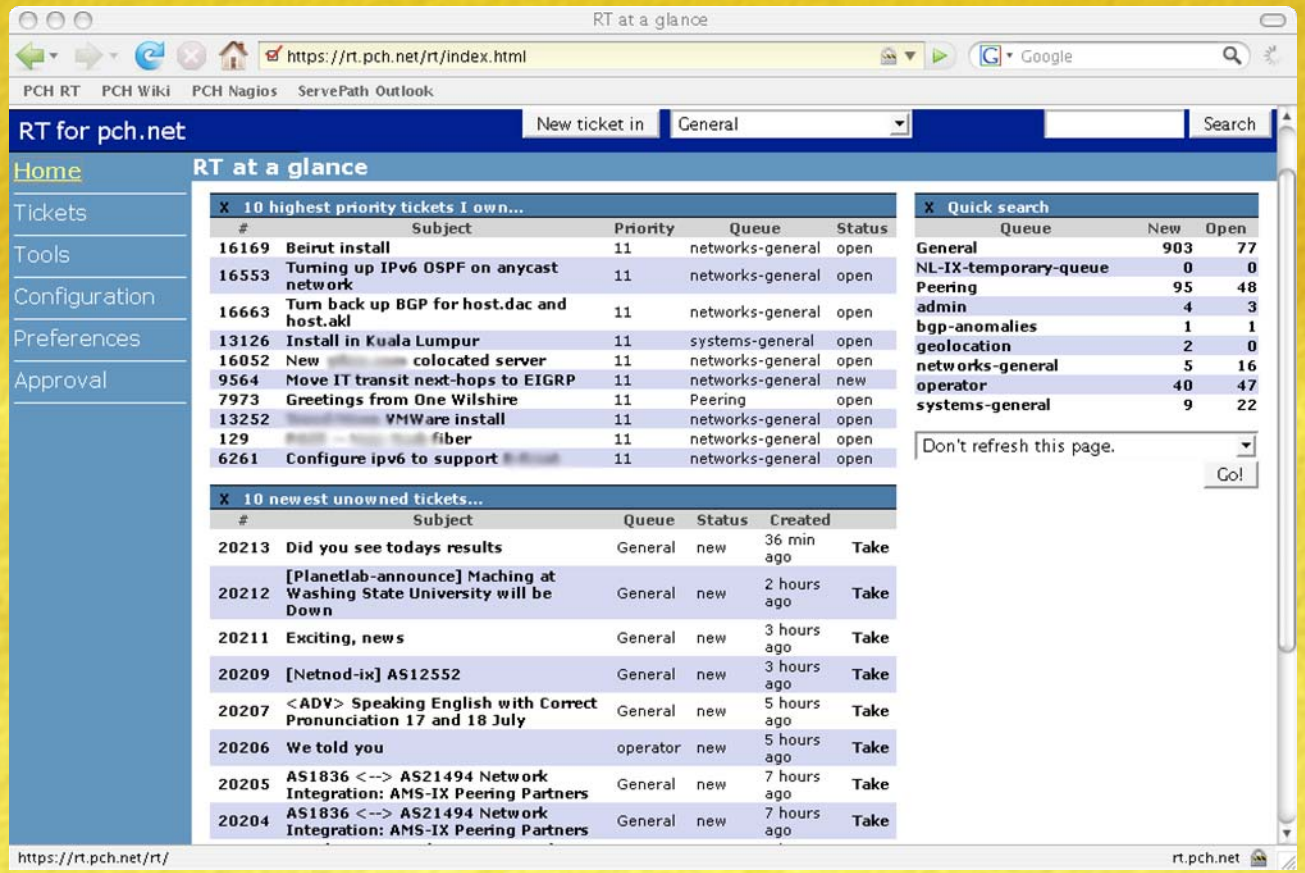
Dealing with customers/peers

- Ticket systems
  - Track communications in a ticket system. Your co-workers will know why a customer is calling.
- Maintenance announcements
  - Let people know what's going on.

Ticket systems

- Gathers customer communications in one place.
- Sets up an audit trail. You know what was done, what was said.
- Makes it easy to take over projects started by other people.
- Maintains “to do” lists.
- “RT” is a good open-source ticket system.

Ticket systems



Maintenance announcements

- Tell your customers and peers before causing outages.
  - Avoid surprises.
  - Don’t make them waste time troubleshooting.
- Don’t overdo it.
  - Sending too many maintenance notices makes people ignore them.
  - Don’t send notices for things people don’t need to know about.
  - Finding the right balance is sometimes tricky.

Sample maintenance notice

Dear NL-ix customers,

We will be performing maintenance work in the following datacenter:

- NIKHEF

This work will be carried out on Wednesday 23 January 2008 starting at 02:00.

When

The work will be carried out on:

Wednesday, January 23, 2008 between 02:00 and 06:00 CET, during the regular scheduled maintenance window.

A brief outage on the connections to the NIKHEF backbone switch will be experienced as the switch is reloaded to activate the current supported Foundry OS release.


Questions?
Further discussion?
Steve Gibbard
scg@stevegibbard.com
http://www.stevegibbard.com