TELE 301
Network Management

Lecture 22: Diagnostics & Ethics

Haibo Zhang

Computer Science, University of Otago

Fault Management

- Fault management
  - It means preventing, detecting, and correcting faults in the network circuits, hardware, and software.
  - Network monitoring based on network management software is essential for fault management.

- Fault detection
  - Dumb devices make fault detection difficult
  - Smart devices can record data about their status and make fault detection viable based on the data collected
  - Critical situations can be found early using alarm messages

Fault Management (cont.)

- Reports on faults are called *trouble tickets*
  - Trouble tickets are helpful to problem tracking, problems statistics, problem solving, and management reports.

- Elements of trouble tickets
  - Time and date of the report and the problem
  - Name and phone of the person who reported the problem
  - Location of the problem
  - Nature of the problem (definition of the problem)
  - Why and how the problem happened and solved

Fault Detection

- Diagnosis principle
  - Look for the most likely causes, such as power, cable
  - Start from simple tests
  - When a situation is confusing, cold mind is helpful.
  - Take notes of what was tried and the effect it had on the problem
  - Draw a conceptual map of the problem
  - Associate symptoms with previous experience

- Approaches to problem solving
  - Trial and error
  - Resolve by example
  - The replacement method
  - Step-by-step using the OSI model

Trial and Error

- This approach requires an assessment of the problem, an educated guess as to the solution, an implementation of the solution, and a test of the results. Repeat the process until the problem is solved
- This approach may be appropriate under the following conditions
  - New system, no data can be lost
  - System not attached to a live network
  - Easy to undo changes
  - Other approaches take considerably more time
  - There are a few possible causes
  - No other resources to help arrive at a scientific solution

Trial and Error (cont.)

- This approach is not advisable under the following conditions
  - A server or internetworking device is live on the network
  - You are instructing an untrained user over the phone
  - Unsure of the consequences of the solutions
  - No sure way to undo the changes
  - Other approaches take the same amount of time
- Guidelines when trial-and-error is used
  - Make only one change at a time before testing the results
  - Avoid making changes that may affect a live network
  - Document the original settings of hardware and software before making changes so that you can put the system back to its original settings
  - Avoid making changes that can destroy user data
  - Avoid making a change that you cannot undo

Trial and Error (cont.)

- Trial-and-error flowchart

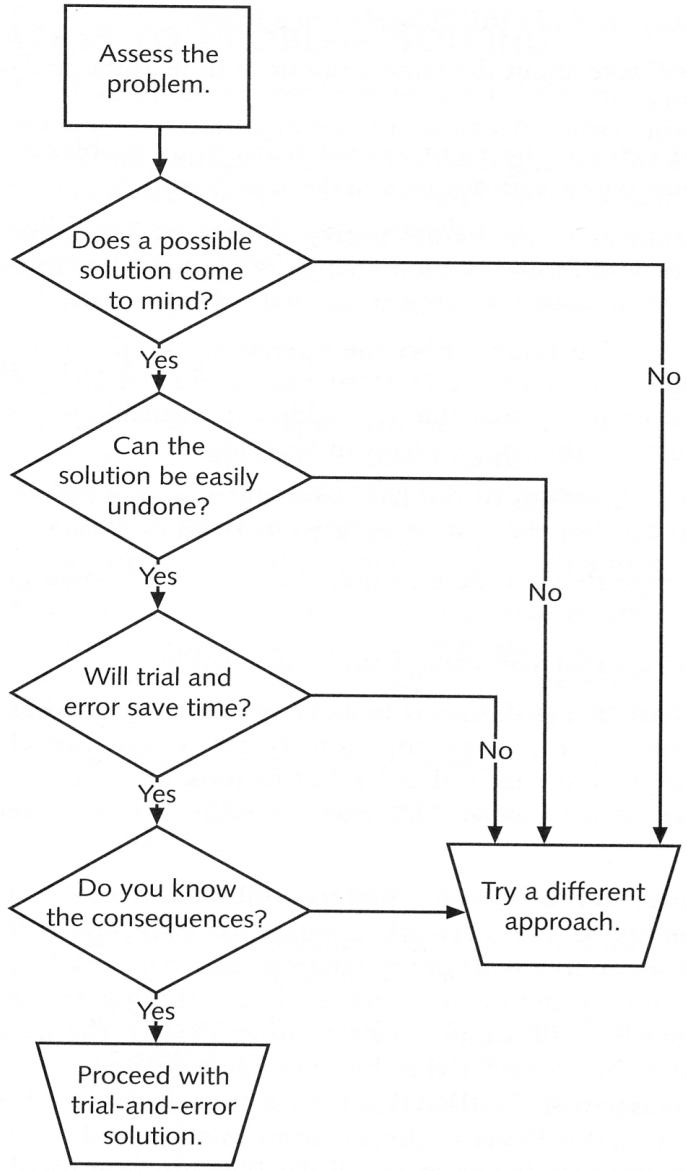

Figure 2-1 Trial-and-error flowchart

Resolve by Example

- Resolving by example occurs when you use something that works, compare it to something that does not work, and then make the one that does not work look like the one that does
- Fastest and easiest way
- Caveats to using the resolve-by-example method
  - Use it only when the working sample has a similar environment as the problem machine
  - Don't make configuration changes that will cause conflicts
  - Don't make any changes that can destroy data that cannot be restored (backup your data first)

The Replacement Method

- Use a working part to replace the possible faulty one
- A simple approach for the following conditions
  - Source of problem can be determined
  - The problem is a defective part
- Rules to follow
  - Narrow the list of potentially defective parts down to one or two possibilities
  - Make sure you have the exact part replacement on hand
  - Replace only one part at a time
  - If your first replacement does not fix the problem, reinstall the original part before replacing another part

Step-by-Step Using the OSI Model

- Based on OSI model
- Test a problem starting at application layer or physical layer, and keep testing at each layer until you have a successful test
- You may start at a middle layer as well
- Most people use it for network support
- You need to understand how networks really work and where you are able to use troubleshooting tools
- Example: network services become unavailable
  - Possible reasons
    - The server died on the server host
    - The line of communication has been broken
    - The latency of the connection is so long that the client has timed out

Diagnosis Based on OSI

- Diagnosis steps
  - Use ping to see if the host is alive
  - If there is any problem with **ping**, there are two further possibilities: the line of communication is broken, or the DNS lookup server is not responding
  - To check if DNS is a problem, use the IP address instead of IP name
  - If the **ping** with IP address fails again we know the problem is not caused by DNS and try next; otherwise DNS has a problem
  - Now the problem suggests a broken line of communication
  - **Traceroute** command can be used to follow a network connection through various routers to its destination

Diagnosis Based on OSI (cont.)

- Diagnosis steps (continued)
  - If a message is responded "No route to host" then we know there is connectivity problem
  - You should at least check your local router and your configuration file **rc.inet1**
  - If there is no problem with ping, we should verify that the service is running on the server host
  - Logon to the server host and use the commands **ps auxlgrep *daemon-name***
  - Check the log files to find out when the service is stopped and the error messages
  - Possible reasons: bug in the program, mis-configured
  - If the service process has not died, then there might be something more fundamental.

Diagnosis Based on OSI (cont.)

- Diagnosis steps (continued)
  - You should check TCP wrapper and firewall. If there is no problem, you should test other services as well...
  - The host may not be able to respond (over-loaded), or unwilling (access denied to your host)
  - We can check if the host is overloaded by checking the process table
  - The host may be under denial of service attack. Use **netstat** to check the number of connection directed to it
  - If there is no attack, there are lots of other reasons
  - Is DNS on the host working?
  - TCP connection close time in the kernel is too long?
  - Disk problems?

Practical Guidance for Diagnosis

- Pay attention to all the facts available about the problem (verify the facts from the user reports)
- Read documentation carefully to clarify some misunderstandings in configuration
- Talk to experts (short cut)
- Read old bug and problem reports (FAQ)
- Examine system log files
- Simple tests and experiments
- Use an experimental environment if possible
- Keep in mind the belief: no problem should be mysterious!

More Examples for Diagnosis

- Example 2
  - System is running slower than it should
  - Possible reasons: disk full, too many processes, memory usage
- Example 3
  - Disks suddenly become full
  - When a disk is full, we should clear enough space to make the system working again
  - However we should also investigate the possible reasons
    - User disk full: big temp files, junk files, normal data growth
    - System disk full: core files, temp files, log files

Problem-solving Process

- Determine the problem definition and scope
- Gather information
- Consider possible causes
- Devise a solution
- Implement the solution
- Test the solution
- Document the solution
- Devise preventive measures

Problem-solving

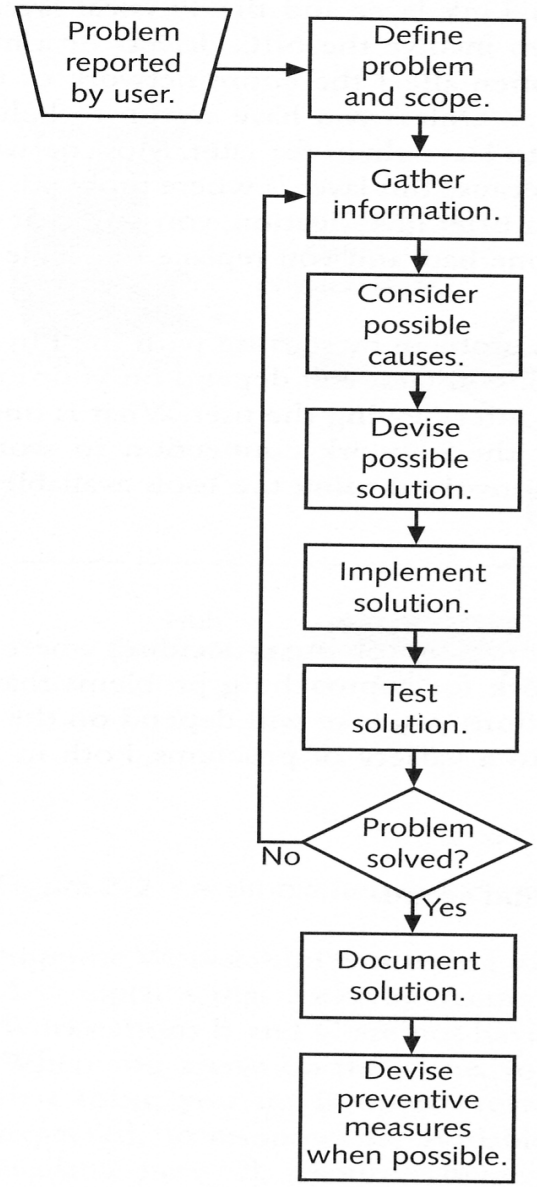

Figure 2-3 The problem-solving process

Problem-solving Process (cont.)

- Step 1: determine the problem definition and scope
  - Is anyone else near you having the same problem?
  - What about other areas of the building?
  - Is the problem occurring with all applications or just one?
  - If you move to a different computer, does the problem occur there as well?
- Step 2: gather information
  - Did it ever worked? Check log files.
  - When did it stop working?
    - Does the problem occur all the time or only intermittently?
    - Are there particular times of the day when the problem occurs?
    - Are other applications running when the problem occurs?
  - Has anything changed?
  - Never ignore the obvious
  - Define how it is supposed to work

Problem-solving Process (cont.)

- Step 3: consider possible causes
  - List possible causes
  - Remove some causes with more information
- Step 4: devise a solution
  - Is the identified cause of the problem truly the cause, or is it just another symptom of the true cause of the problem?
  - Is there a way to adequately test the proposed solution?
  - What results should the proposed solution produce?
  - What are the ramifications of the proposed solution for the rest of the network?
  - Do you need additional help to answer some of these questions?

Problem-solving Process (cont.)

  - Make sure you will be able to put things back the way they were before you implement the solution, just in case the solution doesn't work.
    - Save all network device configuration files
    - Document and back up workstation configurations
    - Document wiring closet configurations
- Step 5: implement the solution
  - Create intermediate testing opportunities
  - Inform your users
  - Put the plan into action
- Step 6: test the solution
- Step 7: document the solution
- Step 8: devise preventive measures

Problem-solving Tools

- Experience
- Colleagues
- Manufacturer's technical support (phone number)
- World wide web (google)
- Network documentation
  - Network topology and internetworking devices
- Network testing and monitoring tools
  - ifconfig, ping, traceroute
  - Protocol analyzer: tcpdump
  - Network monitoring tools: wireshark
  - Cable tester

Service-Level Agreement

- Most organisations establish SLAs with their common carriers and Internet service providers
- SLA specifies the exact type of performance and fault conditions that the organisation will accept
  - Availability: e.g. 99.5%, measured over a month
  - Average round-trip delay, e.g. 110 ms
  - Throughput
  - Mean time to respond: 4 hours or less
  - Mean time to fix: 4 hours or less
  - Mean time between failure: 120 days or more for T1 carrier

Performance Tuning

- Performance and user perception of efficiency are related but two different things
  - User's expectation of efficiency grows
- Performance can be tuned to satisfy one type of users but not all
  - Kernel configuration
  - Specialised hosts
- When the system is running slowly, we should check
  - What processes are running
  - How much available memory
  - Disk space/swap space
  - Network traffic
  - Software dependency

Performance tuning (cont.)

- Performance problems can be identified using **ps aux**
  - %CPU is large. The process is computation intensive or in an endless loop
  - Running time is large. CPU intensive or stuck
  - %MEM is large. SZ is large. If RSS may be small, it indicates swap space is intensively used. If %MEM is growing steadily, it indicates a memory leak
- The command **vmstat** can be used to identify the system bottlenecks
  - Memory statistics: swap, free, buffer, cache
    - Can use **free** to get the same data
  - Swap in/out rates
  - Block in/out rates
  - Interrupts and context switch rates

Performance tuning (cont.)

- Basic network performance tuning
  - One DNS server on each large subnet to avoid sending unnecessary queries through a router
  - Use loop back address 127.0.0.1 as primary name server for name servers themselves
  - Avoid distributed file access on a different subnet. File servers and clients should be on the same subnet
  - Avoid X traffic being sent to the same host via the network. Don't set DISPLAY to hostname:0.0
- Network statistics can be viewed using **netstat** or other monitoring tools
  - Transmission error rate indicates collision rate

Performance tuning (cont.)

- Options for performance improvement
  - Optimising choice of hardware
  - Optimising chosen hardware
  - Optimising kernel behaviour
  - Optimising software configuration
  - Optimising service availability
- Software/kernel tuning
  - Software tuning is complex since it depends on each particular software
  - You have to understand the software before you can tune it
  - OS kernels, servers should be carefully configured, e.g. time for purging socket connections, started by **inetd** or standalone
  - Modules can be loaded on demand

Performance tuning (cont.)

- Data efficiency
  - Parameters for disks and network affect data storage and transmission
  - Block/inode size: If too big it will waste disk space, especially when there are many small files; if too small it will affect transfer efficiency
  - Service response: some services are multithreaded so that multiple clients can be served at the same time
    - Server load balancing
  - Network latency: use policy-based management to prioritise packets so that the quality of data stream of applications such as tele-medicine can be guaranteed

Ethics

- What SAs can do is different from what they should do.
  - Professional ethics
- Principles
  - Others' wellbeing and interests
  - Justice
  - Privacy
  - Love your neighbor (user) as yourself