ECEN 5623 Exercise 6

Dhiraj Bennadi

April 2022

1 Problem 1

1.1 Paper Referred for system failures: Three Mile Island Nuclear Generating Station

Nuclear Power Stations are very critical systems as any untoward incidents at these stations results in great loss of life and property with the effects being perilous not just at the time of the incident but for many years to come. Nuclear incidents are very difficult to contain in case of a failure of any subsystems.

One such incident occurred at the Three Mile Island Nuclear Generating Station (TMI) in Pennsylvania. A cooling system malfunction caused a partial meltdown of the reactor core. This loss-of-coolant accident result in the release of an amount of radioactivity of significant measure. About 2 million people in proximity were exposed to radiation. The major causes identified were that a mechanical system breakdown was causing the cooling system to fail which caused the pressure to rise in the primary system. In order to arrest this pressure rise, the reactor's pressure valve opened which was supposed to be closed. The systems involved in the indication of the valve status did not indicate this incident to the control room until the alarms began to ring, by the time the reactor was already releasing the radioactive coolant into the atmosphere.

Design and Implementation Flaw

Critical Systems with high degree of risk to life and property should always have a backup system or a fail-safe system to completely eliminate the amount of risk and significantly reduce it. Another approach I would suggest over here is in such critical systems is when an abnormality is identified, the main systems should be shut down immediately which did not happen in this case.

1.2 Paper Referred for system failures: Therac 25

Design and Implementation Flaw

1. Software Code review was not done

2. The OS used was not tested and no bench marking was done
3. No Determination of Error situations and identification of error codes was not distinct
4. Adoption of previous versions of code, without determining the impact of the code on new systems.
5. No design in place to prevent interlocks
6. No bounds-checking for maximum value of a variable causing overflow

1.3 Paper Referred for system failures: The Crash of AT&T Network in 1990

Design and Implementation Flaw

1. The failure of the network systems happened due to a software improvement deployed into a live system without testing or anticipating the effect
2. The improvement was made to address the way the signalling of messages was handled between the switches
3. Switch B was expected to receive messages from Switch A for normal functioning upon which Switch B would reset its internal logic indicating Switch A is back in service. However during the reset of Switch B, it received messages for an anomaly which made the software of switch B to send messages to the network to avoid spreading of the problem.
4. This created a chain reaction bringing down the whole network

1.4 Ranking

I would rate the design flaws in this order with the impact of failure as my decision maker.

1. No Backup system in place
2. No Determination of Error situations
3. No Testing of Software before deployment

2 Problem 2

Key findings

1. USS Yorktown suffered a systems failure due to a bad data generated or fed into the database/memory sections of the computers using the Windows NT System

2. The bad data caused the propulsion system to fail
3. The Propulsion system failure was also caused possibly by the an invalid operation divide by 0
4. Another possible reason for failure was also suspected to be an OS issue, which was not immune from failure modes
5. Another design failure was not having a back up plan in place when the main system fails
6. Inter-operability between OS systems is very difficult task to achieve. The USS Yorktown has requirements such that different OS could have handled the systems in a better way.

Alternate Key findings

1. The system failure was rightly believed to be an arithmetic exception (divide by 0) which was not handled by the program causing the Remote Database Manager to fail
2. The USS Yorktown has a total of 27 client PC systems running which were connected on a LAN network over OFC cables with a server. Since the systems were connected on a common network, the failure of system created a domino effect causing the motor and propulsion machinery failure and halting of the ship.
3. There were contradicting claims on whether the failure was addressed using the emergency power units or by towing the ship back to a port
4. The accusations on the issues with Windows NT were always prevalent since UNIX was considered a better OS to handle the control of equipment and machinery
5. The issues of programs, databases, and code within the software was also blamed for the failure

Root cause analysis

1. One of the major issues highlighted was the arithmetic exception of divide by 0. This is an illegal instruction and if not handled well might cause the failure of the system This finding highlights the importance of validation of input data. If an invalid input is forced into the system then the system should have code to handle the invalid input
2. In general software systems should be able to handle exceptions. Exceptional handling is the process of detecting and reacting to computational anomalies, events or exceptional conditions that require a different flow than normal instruction execution.

3. Since the warship is a critical system, the OS used in the software systems has to be error free. Speculations of not handling failure modes in the OS was one of design failure.

4. Since the client systems were connected on a network, the failure in one of the systems should have been communicated to the other systems and total system failure should have been avoided.

5. No backup systems in place for support of failure

Would RedHawk Linux help for large scale mission critical systems like the USS Yorktown

Support

1. I would recommend using of Linux over Windows due to the fact the Windows was primarily developed for average computer users with ease of use as the main selling point. Considering the ease of use, it would always be a best effort service and not real-time. However the versions or flavors of Linux are specifically designed for real-time systems

2. Since Linux is open source, any updates to the kernel are open for the public and based upon this knowledge an user can upgrade the system or not opt to do so. Windows being proprietary, the updates are pushed onto the user. The consequences are addressed as the issues arise.

3. For critical systems, considering the performance of Linux on aspects of security and execution efficiency Linux would be superior to Windows Systems

4. The NightStar version would help the system to be more deterministic

5. The preemption in OS is reduced in Redhawk which would give more execution time to the system in place.

6. RedHawk Linux undergoes benchmarking testing, which is a good metric to evaluate systems

Against

1. Linux is heavy kernel to run specific applications. The system has a lot of preemption which makes it difficult to run applications of point of interest.

2. For a critical application, the Cyclic Executive would be better but as the application develops in terms of size and functionality, the CE is difficult to maintain.