UKCMG Seminar

Systems Engineering for Availability and Performance

Colin Butcher

Agenda

09:30 – 09:55	**Registration**
09:55 – 11:00	**Session**
11:00 – 11:30	**Break - Foyer**
11:20 – 12:20	**Session**
12:20 – 13:20	**Lunch – Plum Bar & Grill**
13:20 – 14:20	**Session**
14:20 – 14:40	**Break - Foyer**
14:40 – 15:40	**Session**
15:40 – 16:00	**Break - Foyer**
16:00 – 17:00	**Session**

Topics

- Performance
- Availability
- Data network technologies
- Storage and storage network technologies
- System platforms and processors
- Software, compilers and operating systems
- Virtualisation
- System engineering and design
- Examples and discussion
- Troubleshooting and analysis

Requirements:

- What is the minimum we have to do?
- How well do we need to do it?

Availability:

- Disaster Tolerance = ability to survive major disruption
- High Availability = ability to survive failures

Performance:

- Performance issues are often the cause of transient system failures and disruption
- The systems have to have sufficient capacity and performance to deal with the workload in an acceptable period of time

Principles of performance:

Understanding throughput, response times and the underlying mechanisms that determine the overall performance characteristics of a system.

The performance envelope

- What does the business require the systems to do?
- What happens to the business if the systems fail?
- What happens if you push beyond the limits?
- How far from the edge are you?
- How do you know?

How can we characterise it?

- **What is “fast” or “slow”?**
- **What are the performance requirements of the system we’re designing or investigating?**
- **What can we measure?**
- **What comparisons can we make?**
- **What “footprint” can we look for?**

Throughput & Response Times

- **Bandwidth – determines throughput**
- **Latency – determines response time**
- **Jitter ("div latency" or variation of latency with time) – determines predictability of response**
- **Think about a transport system:**
  - **What is throughput?**
  - **What is latency?**
  - **What is jitter?**

Bandwidth – determines throughput

- **Worst case is the maximum capacity of the smallest path in the system (the “bottleneck”)**
- **It’s not just “speed”, it’s throughput in terms of “units of stuff per second”**

Latency – determines response time

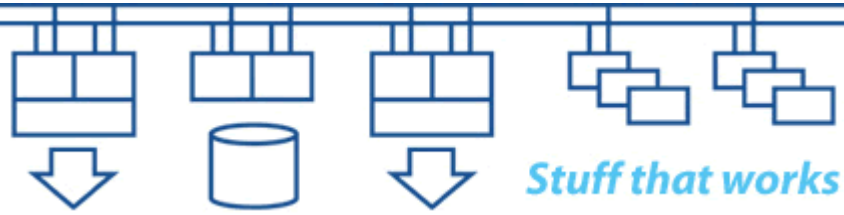
- Worst case is the sum of all the delays in the system
- Latency determines how much “stuff” is in transit through the system at any given instant
- “Stuff in transit” is the data at risk if there is a failure

Jitter ("div latency" or variation of latency with time)

- Determines predictability of response, ie: timeouts
- Ideally zero
- Wildly fluctuating latency is a "bad thing"
- Latency fluctuations can cause system failures under peak load

Principles of high availability

An overview of availability analysis, state transitions and failure detection.



Availability in context

- **Availability is more important than performance**
- **Business continuity:**
  - **It's not just the systems – it's everything!**
- **Disaster tolerance:**
  - **Surviving major site outages without loss of data**
- **High availability:**
  - **Surviving equipment and software failures**

- **We're looking for single points of failure**
- **We're looking for critical components**
- **We need to understand how systems fail and what failure looks like when it happens**
- **We're paranoid about our data**

Availability assessment

- How can we look for points of failure?
- How can we assess the impact of failure?
- Which parts of the system are mission-critical?
- What kind of failure do we prefer?
- What happens to our data when things go wrong?

Reliability and Availability

- **Reliability = Probability of failure at a given point in time, usually expressed as MTBF (Mean Time Between Failures)**

- **Availability = Probability of system being up and running at the instant when need it. Function of MTBF and MTTR (Mean Time To Repair), usually expressed as a % uptime , eg: 99.999%**

- **99.999% uptime (five nines) is equivalent to 5.26 minutes loss of service in a year for a 24x365 system. For a 12hour x 5day operational window it's only 1.87 minutes permissible outage in a year. That's difficult to achieve.**

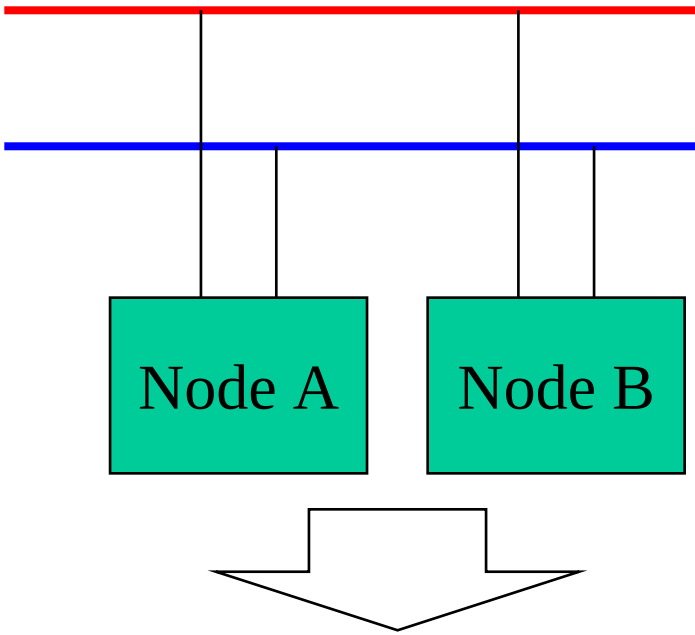
Techniques

There are many techniques which have evolved over the years and there are tools to help you apply them. For example:

- **Reliability Block Diagrams (RBD)**
- **Failure Modes, Effects and Criticality Analysis (FMECA)**
- **Fault Tree Analysis (FTA)**

These techniques can be applied with software tools available from a number of vendors.

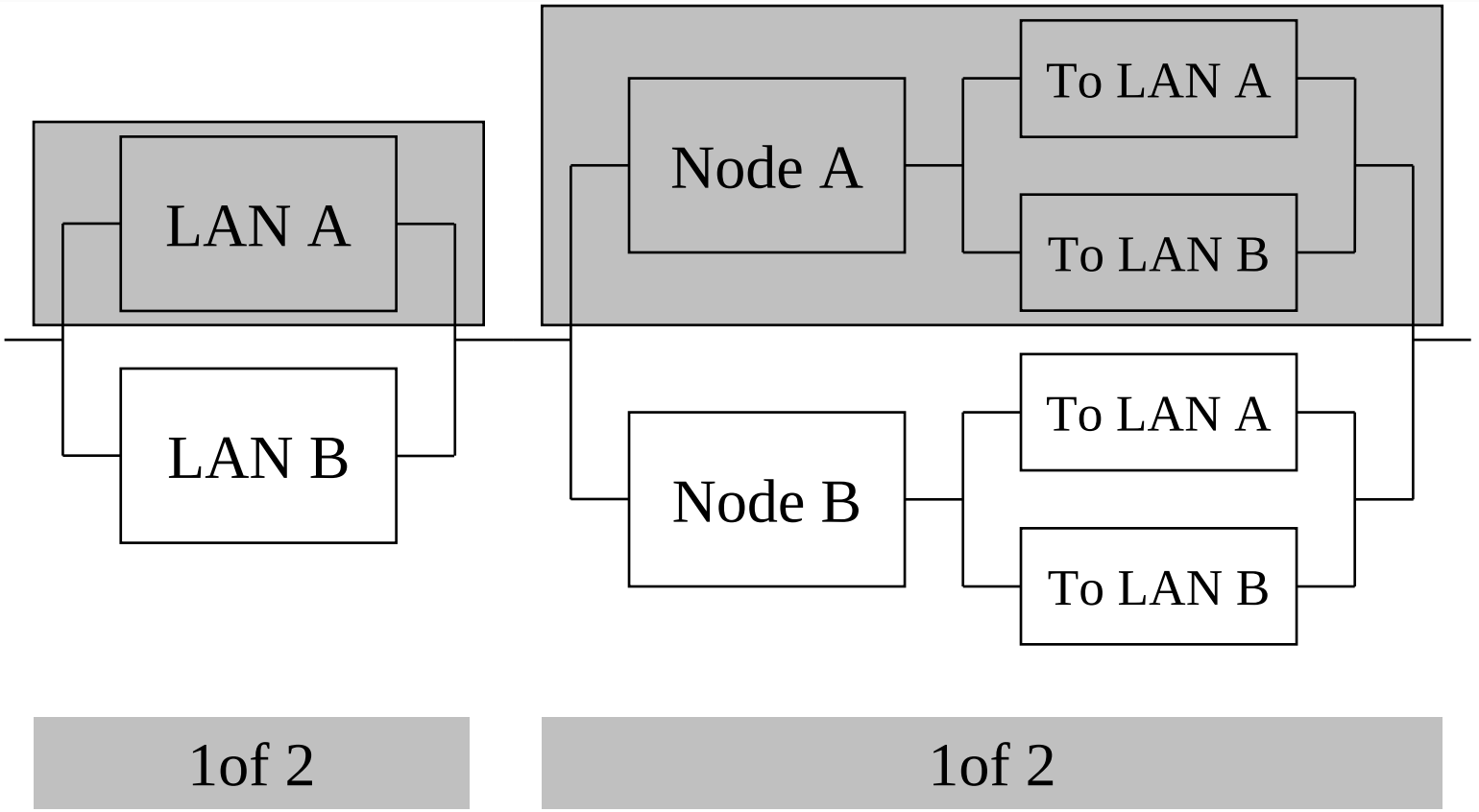
Reliability block diagram (1)



For this to be operational, we need to have:

- **Either LAN, and**
- **Either node, which in turn needs either connection to either LAN**

Reliability block diagram (2)



Reliability block diagram (3)

- **Used to identify single points of failure (SPOF)**
- **Can be used to derive an overall theoretical probability of failure for the system by assigning probabilities of failure to individual items**
- **Be aware that theoretical probability of failure calculations are based on statistics and assumptions – however the process is invaluable in understanding the issues**

- **Failure Modes, Effects and Criticality Analysis (FMECA)**

- **Failure Mode and Effects Analysis (FMEA)**

These are techniques used to:

- **identify potential failure modes**
- **assess the risk associated with the failure modes**
- **sort the issues in terms of importance**
- **identify possible recovery actions**

Fault Tree Analysis

- **Fault Tree Analysis (FTA)**
- **Identify the way that failures can ripple through a system**
- **The "inverse" of a fault tree can help to identify "common mode" failure events and guide fault-finding, eg: loss of one phase can cause loss of power to many devices**

State transitions (1)

- **Take the example used in the Reliability Block Diagram where we have two machines in hot-standby operation**
- **What states can a pair of machines be in?**
- **We need to identify all possible states and ensure that "invalid states" do not occur (or are handled appropriately) and that the state information is propagated to all other participating machines in the overall system**

State transitions (2)

A	B
Off to Master	Off
Master	Off to Standby
Master to Off	Standby to Master
Master to Off	Off to Master
Master to Standby	Standby to Master
Master to Hung	Standby to Confused
And so on…	

State transitions (3)

- **Nothing happens instantaneously**
- **How long do state transitions last for?**
- **What can we do while a state transition is in progress?**
- **Can we ensure that there are no timing windows / flaws?**
- **Can we ensure that there are no “race conditions”?**
- **How can we test it?**

Dealing with failures

- How can we get good information?
- What does “failure” look like?
- How quickly do we need to react to a failure?
- Should we automate decision making?
- How can we recover from failure?

Network connectivity and protocols

An overview of networking and connectivity to systems.

Connectivity

- **Network protocols**
- **Node naming, addressing schemes and routing mechanisms**
- **Multiple NICs and multiple LANs**
- **Map functions to NICs:**
  - **Management (hardware consoles, etc.)**
  - **Clustering**
  - **Network backups**
  - **Data transfers (eg: FTP, NFS etc.)**
  - **Interactive users**

Commonly used network protocols

Typical network protocols:

- **TCP/IP V4 (and all it's components)**
- **TCP/IP V6**
- **Spanning tree (bridges and switches)**
- **LLDP (Link Layer Discovery Protocol)**
- **SCS (OpenVMS clustering)**
- **DECnet**
- **LAT (DECserver terminal access etc.)**

See http://www.wireshark.org/ as an example of a network packet capture and analysis tool

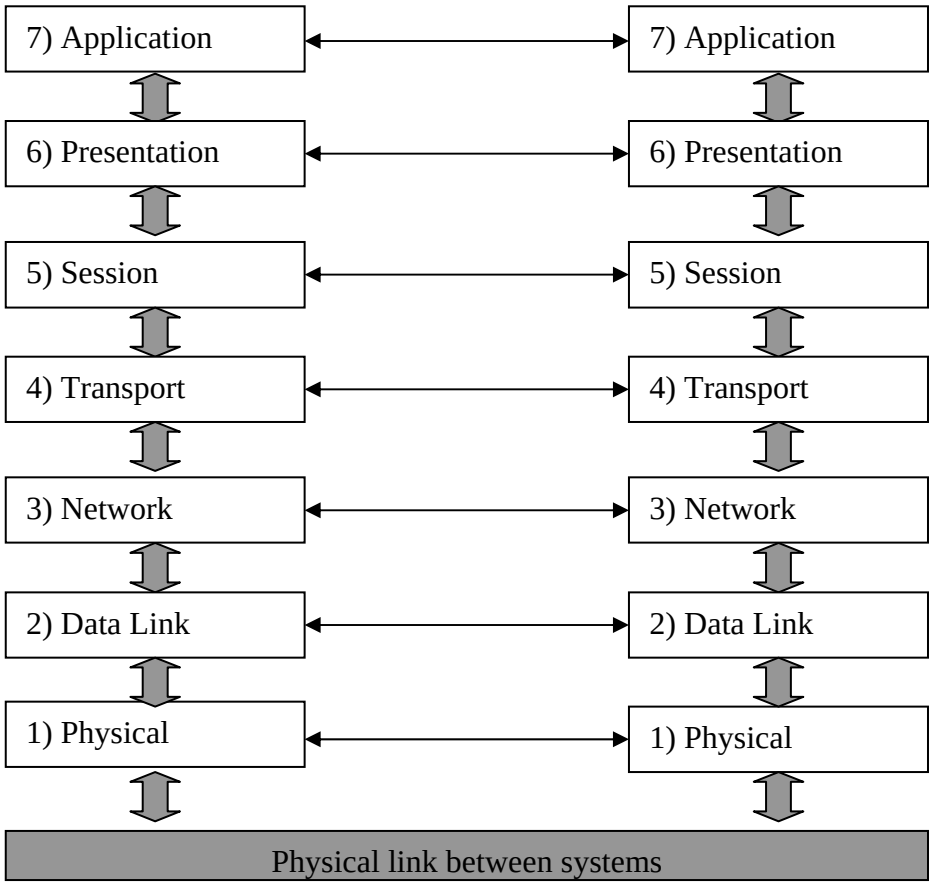
Data networks

- **Ethernet technologies**
- **Physical components and cabling**
- **Protocols and addressing**
- **Network Interfaces**
- **Network Segmentation**
- **Wide-Area networks (WANs)**

OSI 7 layer model

7	**Application**	**Provides for distributed processing and access, contains application programs and supporting protocols (eg FTAM)**
6	**Presentation**	**Coordinates conversion of data and data formats to meet the needs of the individual applications**
5	**Session**	**Organises and structures the interactions between pairs of communicating applications**
4	**Transport**	**Provides reliable transparent transfer of data between end systems with error recover and flow control**
3	**Network**	**Permits communication between network entities**
2	**Data link**	**Specifies the technique for moving data along network links between defined points on the network, and how to detect and correct errors in the Physical layer (layer 1)**
1	**Physical**	**Connects systems to the physical communications media**

Data flow between the layers



TCP/IP V4 layers

- **Layer 4 – port or socket layer (eg: HTTP = port 80, “well known” ports allocated by convention)**
- **Layer 3 – IP addressing and routing layer (eg: 192.168.0.n/24, DNS/BIND resolver user to convert IP hostnames to interface IP addresses)**
- **Layer 2 – MAC address layer (ARP used to convert IP interface addresses to MAC addresses, cached locally)**
- **Layer 1 – Physical layer (transmission media)**

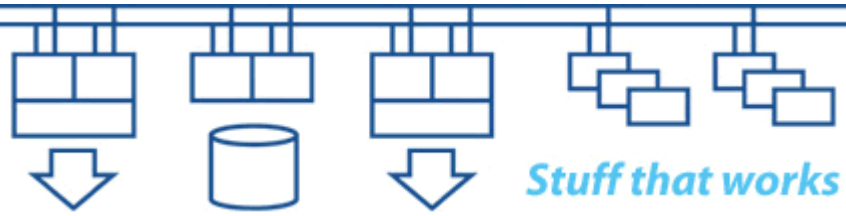
LAN - Ethernet and derivatives

- 10 Mbit/sec
- 100 Mbit/sec (Fast ethernet)
- 1,000 Mbit/sec (Gigabit ethernet)
- 10,000 Mbit/sec (10Gigabit ethernet)
- Copper / fibre (different transmission characteristics)

- Wireless ethernet (2Mbit / 11Mbit / 54Mbit / 108Mbit)
  - Note: WAP, GPRS, HSCSD, Bluetooth etc. are not wireless ethernet.

NICs (Network Interface Cards)

- Provide connection between IO subsystem and network
- Copper / fibre / wireless
- On-NIC processing:
  - Packet creation
  - Address filtering
  - Encryption
  - Protocol processing (TCP/IP offload)



Ethernet addressing

- Hardware MAC address
- Physical MAC address
- Broadcast address
- Multicast addresses
- Point to point addresses
- Ethernet packet format v IEEE802.3 packet format

Why segment a network?

- NICs (network interface cards)
- “Flat” network
- Repeaters
- Bridges
- Switches
- VLANs

Repeaters

- Layer 1 devices
- Provide electrical fault isolation
- Simply re-time and re-transmit signal
- No control of bandwidth
- Beware of cumulative end to end delay exceeding maximum permissible frame timing

Bridges

- Packet content based (Layer 2)
- Store and Forward
- Easy to use and configure
- Poor control of bandwidth
- Spanning tree algorithm
- Provides an extended LAN
- Not all protocols can tolerate the inherent delays in working over an extended LAN
- Remote booting (MOP, BOOTP etc.) will absorb bandwidth

Switches

- Introduce parallelism
- Speed of chipsets (latency & bandwidth)
- Full duplex operation on a single device per port basis
- Traffic monitoring (mirror ports)
- Link aggregation
- Bandwidth control
- “Store and forward” versus “Cut through” switching
- Layer 2, Layer 3 etc. switching
  - Layer 3 generally refers to TCP/IP routing layer
  - Layer 4 generally refers to TCP/IP port numbers, eg: HTTP port 80 traffic)

Virtual LANs (VLANs)

VLANs are another way to segment a network for performance and security

- **Implemented within core switches**
- **Layer 2, Layer 3 etc.**
- **Port based VLANs**
- **Protocol based VLANs**
- **Connectivity between VLANs**
- **VLAN tagging of packets (802.1Q)**
- **VLAN tagging of packets out of NICs**
- **QoS (Quality of Service) and bandwidth reservation**

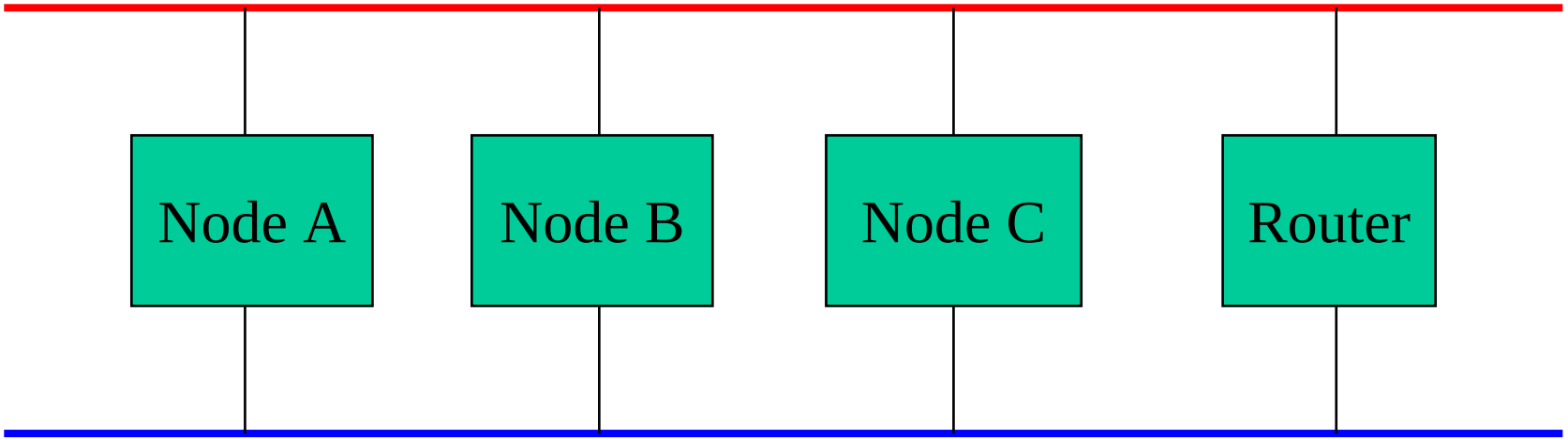
Wireless ethernet

- Shared bandwidth (“flat” network)
- Security (access control, data encryption)
- Roaming (multiple Access Points)
- Antennas (coverage and beam patterns)
- Wireless bridges

Multiple LAN interface behaviours

- Operating system dependent
- Network configuration dependent
- What's possible is determined by the protocol addressing behaviour
- Load balancing requires an "out of order packet cache"
- May wish to allocate specific protocols to different LAN adapters

Dual LAN behaviour



- **TCP/IP – multiple NICs per subnet, dynamic routing**
- **DECnet - multi-homed ES or IS, load balancing**

Wide Area Networking (WAN)

- **ISDN**
- **Leased Line (KiloStream, MegaStream etc.)**
- **Frame Relay**
- **ATM**
- **MPLS**
- **“Dark fibre” and Wave Division Multiplexing**
- **ADSL / SDSL**
- **VPN over Internet connection**
- **Satcom**

Routing

- Routers do not need to be involved in the normal inter-node traffic within a LAN, other than keeping track of who's where and making themselves known
- Routers build knowledge of address (node or interface) reachability on a per-protocol basis
- Protocol address based (Layer 3)
- Need to design addressing scheme
- Bandwidth control
- Design routing paths
- Routing table updates are propagated between routers

Storage subsystems and SANs

An overview of storage subsystems and storage area networks.

Storage networks

- Storage arrays and devices
- Fibrechannel technologies (1 / 2 / 4 Gbps)
- FC and FC-AL
- Physical components
- WWIDs
- Host Bus Adapters (HBAs)
- Storage subsystems and device presentation
- SAN Segmentation (switching, routing, Zones etc.)
- SAN extension (eg: FC over IP)
- Booting systems from SAN devices

Storage arrays (1)

- Array controllers interface to the SAN
- Mirrored cache in the paired array controller
- Paired array controller drives the discs
- Array controllers presents devices to the fabric once virtual discs are created in the array
- Array controllers map storage to physical discs
- Array controllers need to be configured and managed
- Systems can interact with the array controllers (eg: snapshots, snapclones, mirrorclones)
- Details are generally manufacturer specific

Storage arrays (2)

- **Behaviour of physical discs is hidden from the systems by the array controller**
- **Individual disc failures are hidden from the systems and dealt with by the array controller**
- **Array controllers map storage requirement to physical disc devices with appropriate levels of striping and mirroring**
- **Presented devices can grow in size – which requires support in the operating system file system**
- **Throughput only becomes disc limited if you saturate the array controller cache**

Tape libraries

- Backups need to be made in a file structured manner, thus the host operating system needs to write the data
- Tape libraries generally have a large number of cartridges, several drives and a robot to move the cartridges
- Keeping track of cartridges requires software to manage the libraries
- Use separate FC paths to avoid contention for bandwidth in the SAN
- Virtual tape libraries allow a large number of cheap discs (typically FATA) to be accessed as if they were tape – useful for off-site bulk storage

Fibrechannel

- A switch based network optimised for shifting large quantities of data with high throughput and low latency
- All endpoints uniquely identified with a WWID (World-Wide ID)
- Multiple switches can be interconnected
- Inter-switch links can be trunked
- The network between the storage devices and the systems is known as a fabric
- High availability typically uses a dual-fabric SAN
- FC (or FC-SF) is a switched fabric network
- FC-AL (arbitrated loop) is used inside storage arrays

Device addressing and presentation

- **WWIDs are unique**
- **Systems and storage controllers scan the fabric to build a list of paths between devices**
- **Storage devices (eg: EVA Vdiscs) are presented to the fabric by the array controller (HSV controllers for HP's EVAs)**
- **Device presentation can be controlled to limit access to specific paths (by WWID)**
- **Devices are presented to the fabric by the storage controller with a LUN (logical unit number) and (required by some operating systems) a device identifier**

SAN segmentation

- **Device paths and visibility can be controlled by zoning in the switches**
- **Zones can be port based, or WWID based – WWID based generally a better solution**
- **Zones can overlap (think Venn diagrams) so that devices like tape libraries can be accessed by more than one system**

- **Systems (HBAs) need to have BIOS type support for booting from SAN devices**

SAN extension between sites

- **Inter-site links can be "trunked" (as with data networks) to provide sufficient bandwidth**
- **Link "glitches" will cause fabric resets and rescans, so use FC routing in large extended SANs to minimise disruption**
- **Zones can extend across multiple switches (as with VLANs)**
- **WDM (DWDM, CWDM etc.) can be used for extended distance inter-switch links**
- **"FC over IP" can be used to link SANs over an IP data network (beware latency issues – use QoS techniques)**

System platforms and processors

- History and development
- What do we mean by “hardware”?
- CPUs (Central Processing Units)
- Memory subsystem and memory management
- IO subsystem
- Interconnects and bus structures

History (1)

- **The stored program computer concept:**
  - **Babbage's difference engines**

- **Early days:**
  - **Bletchley Park & Colossus**
  - **Valves**
  - **Mercury delay lines**
  - **Paper tape**
  - **Core memory**

History (2)

Computers are only a tool to do a job:

- **Repeatable mathematics quickly (eg: generating tables for logarithms, gunnery etc.)**
- **Solving equations (eg: calculus)**
- **Warfare (eg: signals decoding – Colossus)**
- **Storing and processing data (eg: LEO)**
- **Control systems (eg: fly-by-wire aircraft)**
- **Mathematical modelling (eg: stress analysis)**
- **Data collection and analysis (eg: pharmaceuticals)**
- **Cross-correlating large amounts of information (EG: data warehousing)**
- **Trading systems (eg: data providers, on-line trading)**

**There are 10 kinds of people in the world –
those who understand binary arithmetic and those who don't.**

Boolean logic:

- **basic gates: AND, OR, NOT**
- **derived gates: NAND, NOR, XOR, XNOR**

Logic gates – truth table

Input A	Input B	AND	NOT AND	OR	NOT OR	XOR	NOT XOR
0	0	0	1	0	1	0	1
0	1	0	1	1	0	1	0
1	0	0	1	1	0	1	0
1	1	1	0	1	0	0	1

Basic limits to performance

- Speed of light
- Feature size in “chip”
- Signal degradation
- Power consumption
- Heat dissipation
- Chemical impurities
- Mechanical contacts
- Manufacturing processes
- Software design

Components of a computer system

- CPU (Central Processing Unit)
- Memory subsystem
- IO subsystem
- Storage subsystem
- User access

- Power
- Cooling
- EMI and RFI protection
- Mechanical protection

CPUs (1)

- Processes instructions (I stream)
- Manipulates data (D stream)
- Instruction Set (processor architecture)
- Protection mechanisms (processor modes)
- Flow of control (logic)
- Integer arithmetic
- Floating point arithmetic
- Clocking
- “bit width” (16bit, 32bit, 64bit …)
- Memory Management

CPUs (2)

- **Performance:**
  - **Registers**
  - **Cache**
  - **Pipelines**
  - **Parallel execution**
  - **Multiple cores**
  - **Hyperthreading**
- **CISC (Complex Instruction Set Computer)**
- **RISC (Reduced Instruction Set Computer)**
- **EPIC (Explicitly Parallel Instruction Computing)**
- **Compilers are the key to performance**

Multiprocessing

- Asymmetric Multiprocessing (attached processor)
- Symmetric Multiprocessing (SMP)
- Interconnections (switch, mesh, toroid etc.)
- Latency and Bandwidth
- Synchronisation and Serialisation
- IO processing
- Partitioning
- “Locality of resources”

Memory subsystem

- Used to be small, slow and expensive, so tried hard to minimise memory usage
- Now plentiful, fast and relatively cheap
- Error detection and correction
- Can use memory to gain performance
- Cache and synchronisation across multiple processors
- Memory management and Translation Buffers
- Memory interconnects to CPUs and “Locality” to CPUs (NUMA effects)

IO (Input / Output) subsystem

- **Provides interface with outside world (physical, electrical, logical …)**
- **Connects to storage subsystems**
- **Access times (memory, cache etc.)**
- **“Locality” to CPUs**
- **Device detection (ACPI)**
- **IO device operation (interrupts, registers)**
- **IO interconnects to CPU and memory subsystems**

- **Devices are “block structured”**
- **Volumes are “file structured”**
- **Data are “record structured”**
- **Operating System arbitrates access to control data structures (file system) and the file content (record level locking)**

6: Operating systems and software

- **Operating systems and software are the things we interact with and which turn a pile of hardware into something we can use**
- **Application design for performance and availability is very similar to operating system design**
- **Clear understanding of the different functional elements and the level of interconnection is essential**

Designing for performance

- **Size systems to cope with peaks in workload**
- **Minimise "wait states" (caches, parallelism)**
- **Minimise contention for resources and data structures**
- **Understand the need for synchronisation and serialisation of access to data structures**
- **Maximise "User Mode", minimise the other modes:**
  - **The fastest IO is the IO you don't do**
  - **The fastest code is the code you don't execute**

Parallelism and scalability

- **Understand scalability – do as much as possible once only, do little as possible every time**

- **Understand how the applications could break down into parallel streams of execution:**
  - **Some will be capable of being split into many small elements**
  - **Some will require high-throughput single-stream processing**
  - **Some will require very high interconnectivity between the parallel streams of execution**

Compilers

- Generate the Instruction and Data streams for processing by the system
- Different types of instructions and data are split out into separate sections (shared data, read-only data, local read-write data etc.) for use by the linker
- Generate code for a specific machine architecture
- Optimisation re-orders the code to take advantage of hardware parallelism and processing efficiencies
- Generate debug information
- Linker lays out the image address space and provides hooks for the image activator

Java Virtual Machines

- Java is an object-oriented platform-independent programming language
- Java is not a compiled language running native instructions
- The Java run-time environment interprets the “byte codes” on-the-fly
- The Java run-time environment uses significant amounts of memory
- The Java run-time environment performs “garbage collection” intermittently to remove objects that are no longer in use
- Tuning a system to run Java well can be “interesting”

Limits to software performance

- Ability to make use of hardware parallelism
- Granularity of data structures
- Synchronisation techniques
- Serialisation techniques
- Scalability techniques
- Compilers
- Application design
- Designing and writing very good code requires very good programmers

7: Virtualisation

- **Virtualisation generally refers to running multiple copies of guest operating systems under a primary operating system**
- **It's not necessarily appropriate under all circumstances**
- **Understand the behaviours and implications**

CPU Virtualisation

- **Typically CPU virtualisation decreases overall performance by introducing additional latency and consuming some of the available processing capacity**
- **Techniques such as "VMotion" can allow applications to be moved between processing elements, but again with increased latency and brief pauses in service**
- **Overall system complexity is increased**
- **Understand your workload and the performance you need**

Storage Virtualisation

- **Typically storage virtualisation increases storage subsystem performance by hiding the behaviour of the underlying hardware behind large caches**
- **Techniques such as striping and mirroring are done by the storage subsystem, not by the operating system**
- **Failures and recovery are handled by the storage subsystem, not by the operating system**

Systems need to be able to:

- **Survive failures (resilience and failover)**
- **Survive changes (adapt and evolve)**
- **Survive people (simplify and automate)**
- **Never corrupt or lose critical data (data integrity)**
- **Requirements never remain static over an extended period of time, so we need to be able to make changes during the operational lifetime of the system**
- **Circumstances change, so we often need to be able to extend the operational lifetime and scope of a system**

Problem solving concepts

Borrowed from TRIZ

	Before	Now	After
Environment			
System			
Component			

Design decisions

- **Big decisions which have long-term implications and constraints, eg: disc block size = 512bytes**
- **Small decisions which seem big at the time, eg: memory page size = 512 bytes, now variable**
- **Requirements you don't yet understand or know about**
- **Design for change**

Do we really need 100% uptime?

- **Safety-critical systems (especially safety-critical real-time monitoring and control systems such as air traffic control) require exceedingly high levels of availability. They also have to be fail-safe in order not to endanger lives.**

- **True 24x365 mission-critical systems are fairly rare. With these there is no "downtime window" to take backups, fix faults or to make changes. So, whatever you do has to be done "live" – and very carefully!**

- **The closer you get to 100% uptime the more expensive a satisfactory solution will become.**

"Survivability test"

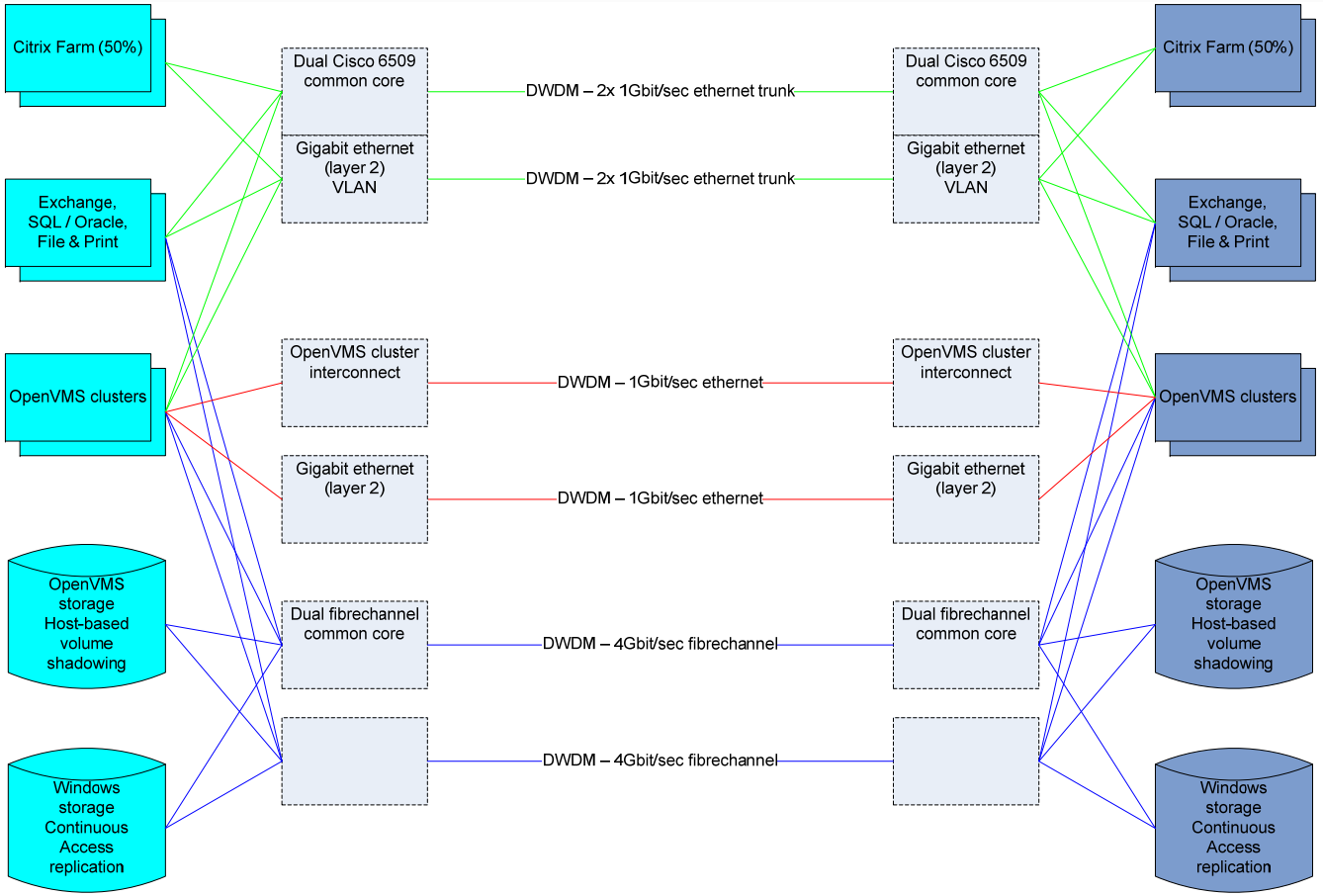
Cause of Outage:	Planned (Maintenance)	Unplanned (Failure)
Hardware	?	?
Operating System	?	?
Network Layer	?	?
Layered Products	?	?
Application Software	?	?
Application Data	?	?
Environment	?	?
Staff	?	?

9: Examples and discussion

- **Safety-critical and mission-critical system:**
  - **Port from Alpha to Integrity**
  - **Move from 3x clusters to single Integrity cluster**
  - **Move from HSG80s to EVA4100s**
  - **Move to multiple NIC connectivity**

- **Similar principles apply in many other cases, eg:**
  - **Satellite flight control**
  - **Finance data processing**
  - **Process control**

Example overall infrastructure



System configurations

- System configuration – the art is to select components that work well together and which provide the bulk of what you need with minimal additional work
- Establish the minimum requirements that have to be met – and do it as well as possible
- Availability and performance have to be designed in to the application
- Monitoring and automation are key components
- Understand the typical behaviour of your systems and be aware of changes
- Configuration control – hardware, settings, software etc.

10: Performance investigation

- What “footprint” can we look for within the system?
- What effects does a performance problem cause?
- What has changed, not necessarily within the system?
- Most performance problems are transient
- Is there a problem at all?

How do we go looking?

- Follow the data flow
- Time series data sets allow us to make comparisons
- The sample rate must let us see the transient peaks and spikes in workload
- You must understand your workload and know what it looks like over time
- There is no substitute for experience to interpret and analyse the data

Relative time is more useful than absolute time

- Need to be able to order events across the network based on timestamps
- UTC Timestamp format
  - Time value
  - Inaccuracy component
- External reference clocks
- NTP

11: Summary and conclusion

- **Take a holistic view**
- **Design in performance and availability**
- **Design in monitoring and logging**
- **Design in debugging and problem tracing**
- **Follow the data flow and view it from a user's perspective**

Systems infrastructure

- **Use multiple systems within a site – and have appropriate physical separation between them**
- **Use multiple sites with appropriate geographical separation**
- **Understand what happens when a failure occurs and how the overall system configuration is likely to respond**
- **Avoid the risk of more than one system thinking that it's in charge when a failure has happened**
- **Ensure that the surrounding infrastructure and environment is appropriate to the needs of your systems**
- **Configure the individual components of the systems in a manner that maximises interchangeability**

Storage infrastructure

- **Destroying or corrupting data is the ultimate disaster**
- **Availability is more important than performance**
- **Size storage subsystem based on minimum components and maximum estimated throughput**
- **Segment storage subsystem to provide gradual degradation rather than wholesale failure**
- **Need adequate backup capacity and throughput in order to meet permissible backup windows**
- **Understand application behaviour and storage performance requirements (bandwidth and latency)**

Network infrastructure

- Scale network so that overall performance is based on minimum essential number of paths and maximum estimated traffic
- May wish to take advantage of installed bandwidth capacity to provide additional functionality when everything is working
- There is no such thing as a single protocol network
- Understand the behaviours of the different protocols under failure conditions
- Segment the network to provide gradual degradation rather than wholesale failure

Software considerations

- Operating system design
- System configuration
- Performance tuning
- Application design (scalability, data integrity, availability)
- Compilers generate code better than you can
- Use the OS features
- Portability
- Reliability
- Supportability
- Testing and documentation

Design and implementation

- Think big, implement small
- Documentation
- Portability
- Modularity
- Reliability
- Scalability
- Software build process
- Software installation process
- Configuration control
- Testing (development, pre-deployment, post-deployment)
- Support and planning

Thank you for your participation

Q & A