Group 1 – Security (AFS)

You are an authorized user walking up to a workstation in CMU to access a file stored in Vice. Once you get access to the file you plan to modify it and store it back and logout.

Enumerate the steps from the beginning to the time you log out, clearly identifying the actions taken locally on your workstation, the communication between your workstation and Vice.

Identify the keys generated, exchanged, and discarded in the course of this activity.

Group 2 – Access rights (AFS)

Discuss the difference between the group and subgroup semantics of Andrew file system and those provided by the Unix file system.

Your discussion should include the granularity of protection domains,

and inheritance of rights, and types of rights supported in each.

Group 3 – DoS (AFS)

What are the weaknesses in the Andrew systems that makes it vulnerable to Denial of Service (DOS) attacks?

[Cautionary hint: Our discussion in class was only the tip of the iceberg!]

Group 4 - Saltzer

Discuss the security vulnerabilities identified in Saltzer's paper.

What are the technical challenges in addressing the above vulnerabilities?

Using GT campus as an example, discuss to what extent any or all of all these vulnerabilities addressed.

Group 5 – Giant scale services

Discuss the pros and cons of load management at different levels in giant scale services.

Discuss the tradeoffs between replication and partitioning in architecting the data repositories of giant scale services.

Group 5a – Giant scale services

Define the availability metrics: "uptime", "yield", and "harvest".

Explain the DQ principle and how it helps in handling graceful degradation of giant scale services.

Discuss the pros and cons of the three approaches to online evolution of giant-scale services: "fast reboot", "rolling upgrade", and "big flip".

Group 5b – Map-reduce

Write a Map-Reduce solution for ranking the pages of web pages. Be explicit in identifying the <key, value> of the map and reduce functions both for input and output.

Group 5c – Map-reduce

What "heavy lifting" is involved in large-scale distributed implementation of the map-reduce paradigm.

What is the work done by the "Reduce" worker thread in the implementation of the Map-Reduce paradigm.

How does the implementation deal with redundant Map and Reduce workers that could potentially be working on the same "split" of the data set for the map and reduce operations, respectively?

Group 6 – Coral CDN

A Coral node makes a web page request for a coralized URL through its web browser. Step through the process by which it gets the page it is requesting to appear on the browser.

Group 7 – Coral CDN

Distinguish between the storage and retrieval mechanisms in traditional DHT and DSHT.

[Use the link for DHT from the class presentation powerpoint if you need more information on traditional DHT.]

Group 8 – Coral CDN

Explain tree saturation. How is that experienced in CDNs?

Explain how the key-based routing layer of Coral mitigates the slashdot effect.

Group 9 – LRVM

LRVM lives above the operating system. The VM subsystem and the LRVM subsystem are thus independent of each other.

What implications does this have on functionality and performance?

In particular, how does this decoupling show up for applications running on top of LRVM?

Group 10 – LRVM

Develop the data structures and pseudo code for performing incremental truncation.

Group 11 – Rio Vista

What are the sources of problems in computer systems that leads to
failure?

How are these relaxed in Rio Vista?

The Vista RVM library is 700 lines of code as opposed to 10K lines for a comparable functionality implemented in LRVM. Explain why.

Group 12 - Quicksilver

Give examples with justifications of servers that fall into the "stateless", "volatile", and "recoverable" classes, respectively.

Group 13 - Quicksilver

Discuss with examples of situations that are appropriate for each type the four different commit protocols (one-phase immediate, one-phase standard, one-phase delayed, and two-phase)

Group 14 - Quicksilver

How does Quicksilver manage its log?

What are the optimizations used to reduce latency for log writing and dealing with memory pressure of excessive log generation?

Group 14a - TxOS

Give user level and system level examples of where transactional semantics will help in increasing concurrency while ensuring atomicity and isolation for the actions.

Discuss the semantics of transactional support in TxOS relative to Quicksilver.

Explain the system level race condition (with an example) caused by   "time of check to time of use" phenomenon in system code.

Group 15 – Quicksilver/LRVM

Identify points of similarity and differences between LRVM and Quicksilver.

Group 16 - LRVM

LRVM provides "bounded persistence". Explain.

Group 17 - LRVM

The authors claim that structuring the LRVM as a user-level library as opposed to a collection of tasks communicating via IPC led to better scalability as evidenced by lower CPU load for LRVM compared to Camelot.

While this may be true since Camelot is built on top of Mach, would you agree with this reasoning as ground truth?

[Hint: Recall some of the papers we read on OS structures.]

Group 18 - Coda

"Coda emulates Unix semantics": Explain what this means.

Explain "availability" and how does Coda provide availability. What guarantees does the user of Coda get with respect to availability.

Group 19 - Coda

Explain the mechanism in Coda for availability and scalability.

Explain the "callback" mechanism in Coda.

Explain the "optimisitic replication" mechanism of Coda. How does it affect the semantics of file usage by the client of the file system?

Group 20 – Treadmarks DSM

Consider a TreadMarks DSM system with 3 pages A, B, and C, and lock L. The following accesses happen in the order shown. Assume initially the lock L is available for N1 and that the pages A, B, and C are with nodes N1, N2, and N3, respectively.

Node N1: lock(L); modify A; unlock(L);

Node N2: lock(L); modify B; unlock(L);

Node N3: lock(L); read A; read B; modify C; unlock(L);

Show the "when" and "what" actions take place in the different nodes in terms of (a) messages exchanged between the nodes, (b) twin and diff creations at the nodes, (c) invalidations of pages at the nodes, and (d) propagation of diffs between the nodes.

Group 21 - GMS

Explain the functionality of each of the data structures (PFD, GCD, POD) in the global memory management scheme.

In the actual implementation, how does each node collect the age information for its pages?

Group 22 - xFS

xFS builds on three prior technologies: RAID, Zebra LFS, and multiprocessor cache consistency.

Discuss the limitations in each of these technologies with respect to realizing a scalable serverless NFS.

Group 23 - xFS

The paper claims:

"For instance, in a 32 node xFS system with 32 clients, each client receives nearly as much read or write bandwidth as it would see if it were the only active client."

Explain this statement.

Group 24 - xFS

In a centralized file system, the server performs the functions of managing the data blocks, metadata for the files, server-side file cache, and consistency of data blocks of files cached by multiple clients.

Discuss how these functions are carried out in xFS.

Group 25 - DSM

Explain the difference between eager and lazy release consistency with code samples.

What are the pros and cons of each approach?

Group 26 - DSM

Explain the following terms with code samples to illustrate the point:

a) data race

b) false sharing

Group 27 - TreadMarks

Develop the data structures and pseudo code for the TreadMarks DSM.

Take into account the distributed lock manager and the protocol actions needed commensurate with the memory consistency model and the coherence mechanism of TreadMarks.

Group 28 - GMS

Assume 4 nodes (N1 through N4) each with 16 physical page frames. Initially none of the nodes have any valid pages in their physical memory. N1 and N2 each make a sequence of page accesses to 32 distinct pages all from the disk. N3 and N4 are idle.

What is the state of the cluster (i.e. local and global caches at each node) at the end of the above set of accesses?

Group 29 - GMS

How is the minimum age of a page (cluster wide) determined? Explain the algorithm to determine the MinAge.

Group 30 - EJB

The paper describes 5 different implementation methods for an EJB application.

Discuss qualitatively the pros and cons of each method.

Group 31 – Java RMI

Explain with justification the difference in the semantics of parameters passing for remote Java objects compared to local ones.

Group 32 - Spring

Explain the basic abstractions supported by the nucleus of Spring.

Compare the microkernel in Spring with Liedtke's recommendation on microkernel construction.

Group 33 - Spring

What is the relation between the address space, pager, cache, and memory objects in the Spring system?

How do they work together to provide the virtual memory support in Spring?

Group 34 – RT Linux

Distinguish between "timer latency", "preemption latency", and "scheduling latency".

How do these terms affect a latency-sensitive application?

Group 35 – RT Linux

Discuss the pros and cons of "periodic timers", "one shot timers", and "soft timers" from the point of view of latency-sensitive applications.

How does the "firm timer" combine the advantages of all three above timers?

Group 36 – RT Linux

Discuss the implementation issues for "Firm timer".

What are potential ways to reduce "preemption latency"?

Group 37 – Time synchronization

What are the problems with network (feedback) based time synchronization? Under what conditions does it work well, and under what conditions it does not work well?

Differentiate between feedback-based and feed-forward based time synchronization.

Group 38 - PTS

What are "situation awareness“ applications? How do they differ from typical enterprise applications (such as DB transactional systems, online banking, stock trading, etc.)?

What are the pain points in building large-scale camera networks for situation awareness?

Group 39 - PTS

How does the PTS computational model (with PTS threads and channels) differ from a Unix distributed program written with sockets and processes?

What is the "heavy lifting" involved in implementing the get/put primitives of PTS?

Group 40 – Thekkath and Levy

As a designer of an RPC subsystem, what are the avenues available to you for shaving the marshaling and unmarshaling software overhead?

Explain your answer with respect to the expected benefit, potential protection concerns for each such avenue.

Group 41 – Active Networks

Consider an application wanting to send the same message to N of its peers. Let Tm be the time for the application to construct and send a message. Let Hc be the number of common hops that the message has to traverse. Let Hu be the average number of unique hops to get to each of the intended destinations. Let Ta be the time required to send a message over one hop.

Construct a simple mathematical model and show when an active network that implements a multicast service in the network may benefit over a vanilla Internet. Assume that a message fits in one IP packet.

Group 42 - Ensemble

Using Lamport's distributed mutual exclusion algorithm as an example, develop an abstract specification and a concrete specification.

You do not have to adhere to any specific syntax (such as IOA), and can invent your own so long as it is straightforward and understandable.

Group 43 - Ensemble

Ultimately this paper is about specializing of a subsystem (communication stack to be specific in this case). Other papers we have seen so far in this course (SPIN, Exokernel, and Thekkath and Levy) also deal with a similar theme.

What are the similarities and differences between Ensemble/Nuprl and these other works?