Midterm
Fall 2001 15-712

Please write your name on the following line:

____

This midterm is open book and open notes.

You should write brief, precise, and legible answers. Rambling brain-dump answers may be punished twice — you run out of time writing them and we run out of patience reading them. :–)

There are eight questions but we will only grade **six**. List the **six** questions we should grade:

____

Name: ______________________________

1. The Virtual Log paper shows how a disk with eager writing can reduce write service times for small synchronous writes by more than 90%. Would a file system using track-aligned extents see benefits of that magnitude from eager writing? Explain.

   No. A small synchronous write requires a seek, a rotational latency delay, and then a small transfer time. A track-sized request requires a seek and a full rotation of transfer. Thus, the room for improvement is much smaller.

   For a normal synchronous disk write, seek + rotational latency + transfer = 6 ms + 3 ms + 0.2 ms = 9.2 ms. The real work is only 0.2 ms, thus eager writes have lots of room in which to make improvements.

   Compare this to a traxtent write, seek + transfer = 6 ms + 6 ms = 12 ms. The real work is 6 ms, thus write times can only be reduced by a maximum of 50%.

   Even without zero-latency disks, the analysis is, seek + rotational latency + transfer = 6 ms + 3 ms + 6 ms = 15 ms, leading to a maximum possible reduction of 60%.

   Note, the results for traxtents would be even worse when the free space for eager writing is fragmented. In this situation, writing a track's worth of data may require several seeks/rotation delays.

Name: ____________________________________

2. Assume that a log-structured file system sits atop a disk with eager writing support. Explain one workload under which the eager writing would help performance and one under which it does not.

   Eager writing improves the performance of synchronous writes. Log structured file systems attempts to avoid synchronous writes by coalescing writes into a log. So long as the LFS is not forced into performing synchronous writes, eager writing does not help. Applications can force LFS into performing synchronous writes by calling fsync() after writes. Workloads could do this because they require their writes to be synchronous (e.g., an ATM machine does not give out cash before persistently logging a commit record for the transaction).

Name: ______________________________

3. Consider the *Better Security via Smarter Devices* proposal with regard to Saltzer's principles of secure systems design. Identify one principle that the proposal follows and one that the proposal violates. Explain.

   Follows Economy of Mechanism. Each device should have an interface and an implementation that is much smaller and cleaner than those of the OS. As such, correctly designing, fully analyzing, and correctly implementing each device's security perimeter should be easier than an entire OS.

   No impact on Fail-Safe Defaults.

   Non-obvious impact on Complete Mediation. Although, it is likely that the OS would cache items and thus prevent smart devices from mediating access those items.

   No impact on Open Design.

   Could enable Separation of Privilege. More security perimeters could lead to better separation of privilege, but it is not clear how an OS/system could take advanatage of this.

   Could enable Least Privilege. Smaller, finer security perimeters could facilitate least privilege. However, it is again not clear how an OS/system could take advantage of this.

   Violates Least Common Mechanism. By distributing trust amongst components, there is, in the aggregate, more to trust. Consider "Reflections on Trusting Trust", - you could hide a fair bit of dastardly code in a smart device. Just consider the mechanisms the RIAA hopes to deploy in hardware - availability of local data could become dependent on remote servers being accessible.

   Violates Psychological Acceptibility. It is an open research question to demonstrate how security distributed amongst smart devices can be packaged as a self-maintaining form, or in a form that system adminstrators don't get seriously worked up about.

   Non-obvious impact on Work Factor. This principle can only be considered for a complete design.

Name: ____________________________________

4. System crashes are “yucky” [Greg’s wording, not Jay’s—Ed.]. Explain what data is lost (if any) and what recovery mechanism is used for each of these scenarios:

   (a) A client crash on a file sytem using soft-updates.

   Modified data in the cache that has not been written back to disk is lost. The data on the disk is consistent and can thus be used immediatly. Further recovery can be done in the background, but it is only necessary to reclaim unclaimed resources and ensure reference counts are accurate.

   (b) A server crash in NFS File Server Appliance.

   Data is not lost since all requests that are cached on the server side are in NVRAM. To recover, the requests cached in the NVRAM are played forward against the last consistency checkpoint. Consistency checkpoints are done every 10 seconds. An NFS File Server Appliance can reboot and recover from an unclean shutdown in around a minute.

Name: ____________________________________

5. When a server supporting callbacks crashes, it must ensure that it knows about any valid callbacks before continuing to serve requests. One approach to doing this is to treat callbacks as leases, having them expire after a given amount of time. Explain how a server using this approach can achieve a known callback state after a crash. What trade-off is inherent in the lease expiration time?

   Servers that grant leases can achieve a known callback state after a crash by waiting for the length of a lease. At this point in time, all client leases will have expired. During this period, old leases can be renewed, but new leases cannot be granted (this would require client requests for renewing leases to be distinct from asking for leases).

   The trade-off inherent in the lease expiration time is between the time until a crashed server can start granting new leases after it has rebooted, and how often clients must renew leases (i.e., between recovery time and server load during normal operation).

Name: ___________________________

6. The xFS file system offers significant scalability. Identify three security dilemmas created by its architecture. (Think about ways that an attacker could exploit it.)

   xFS explicitly assumed trusted peers. Thus, from a security perspective, it has many features that can be attacked.

   - *Non-cooperative caching. Read in things you want to corrupt. Cache them. When the item is requested from your cache, lie about its contents. This could make it look like other peers are corrupting data.*
   - *Inconsistent caching. For portions of the namespace a compromised peer owns, it can lie about who caches copies. This is an availability / performance attack. By refering peers to other peers that do not have cached copies, you force the requesting peer into expensive timeout / error recovery.*
   - *Error prone storage. Once a peer is corrupted, all of the data it owns can be corrupted. Thus the portion of the namespace that the compromised peer manages can be corrupted. This is possible since stripe groups that could provide integrity are below the Imap which is the local portion of the global namespace.*
   - *Unauthorized access (read & write). Access control, although it may be part of the FS, is not completely mediated. The manager map allows any peer to see which machine manages which part of the file index space. Any peer can request any file index from the appropriate manager - without presenting any proof to the manager that it is allowed to read/write the file. For reads, this is a breach of confidentiality, for writes this is a breach of integrity. In the paper they state that this type of security could be achieved if you can trust other kernels, but it is not reasonable to trust the kernel of a compromised machine.*
   - *Just imagine the fun of compromising the distributed cleaner, which has write access to everything...*

   Note, a particularly good answer to this question broke the answer into three parts: availability flaws, confidentiality flaws, and integrity flaws.

Name: ________________________________

7. The River system focuses on dynamically balancing load across servers and across clients to avoid bottlenecks created by slow components. Abacus dynamically shifts functions between a client and a server. Is Abacus's goal to balance load? Explain how it does so or give an example of when it does not.

   The main focus of Abacus is improving the performance of applications that manipulate large data sets by deciding where it is currently best to perform some function (at the client or at the server). The decision is made based on the cost of CPU cycles, network bandwidth, etc. In some cases this may balance the load between a client and a server because as either becomes loaded its CPU cycles become scarce and performance may be improved by migrating work off of the loaded machine.

   However, consider running an extremely selective filter, which must look at the majority of the stored data, on a client/server system with a slow network. In this case, all of the work is going to be performed on the server because the cost of shipping data to the clients to parallelize the work outweighs the benefit. Abacus will heavily load the server while leaving the client idle. Thus, balancing load between the client and the server is a side-effect of some actions of Abacus, not its main goal.

Name: ______________________________

8. AFS was designed to scale to a large number of clients. The NASD architecture was also designed to scale. What kind of scalability does NASD offer that AFS does not? Ignoring cost, how could AFS offer the same kind of scalability?

   NASD enables bandwidth to scale. AFS does not directly offer this type of scalability.

   Without a cost constraint, AFS could offer similar bandwidth scalability to NASD by limiting each AFS server to a single disk, and increasing the number of volumes in the AFS system. This system would resemble (approximately) the architecture from the NASD paper that enabled bandwidth to scale. However, the hardware and administration costs of deploying such an AFS system are prohibitive compared to NASD.