CS194-24
Advanced Operating Systems Structures and Implementation Lecture 25

The Swarm
Extreme Distributed Storage
Quantum Computing

May 6th, 2013

Prof. John Kubiatowicz

http://inst.eecs.berkeley.edu/~cs194-24

Goals for Today

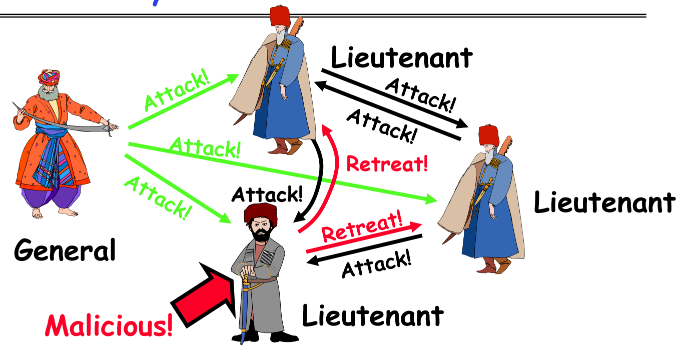
- Trusted Computing
- The Swarm Vision
- Extreme Distributed Storage (OceanStore)
- Quantum Computing

Interactive is important!
Ask Questions!

Review: Distributed Decision Making Discussion

- Why is distributed decision making desirable?
  - Fault Tolerance!
  - A group of machines can come to a decision even if one or more of them fail during the process
    - » Simple failure mode called "failstop" (different modes later)
  - After decision made, result recorded in multiple places
- Undesirable feature of Two-Phase Commit: Blocking
  - One machine can be stalled until another site recovers:
    - » Site B writes "prepared to commit" record to its log, sends a "yes" vote to the coordinator (site A) and crashes
    - » Site A crashes
    - » Site B wakes up, check its log, and realizes that it has voted "yes" on the update. It sends a message to site A asking what happened. At this point, B cannot decide to abort, because update may have committed
    - » B is blocked until A comes back
  - A blocked site holds resources (locks on updated items, pages pinned in memory, etc) until learns fate of update
- Alternative: There are alternatives such as "Three Phase Commit" which don't have this blocking problem
- What happens if one or more of the nodes is malicious?
  - Malicious: attempting to compromise the decision making

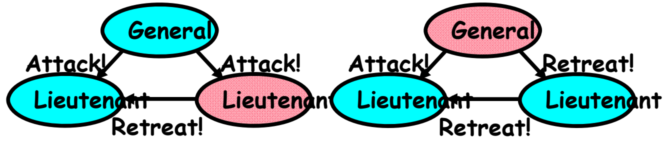
Recall: Byzantine General's Problem



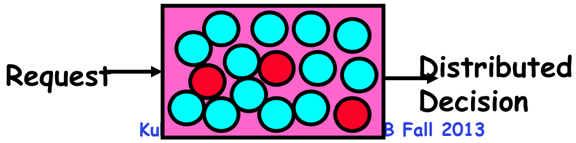
- Byazantine General's Problem (n players):
  - One General
  - n-1 Lieutenants
  - Some number of these (f) can be insane or malicious
- The commanding general must send an order to his n-1 lieutenants such that:
  - IC1: All loyal lieutenants obey the same order
  - IC2: If the commanding general is loyal, then all loyal lieutenants obey the order he sends

Recall: Byzantine General's Problem (con't)

- Impossibility Results:
  - Cannot solve Byzantine General's Problem with n=3 because one malicious player can mess up things



  - With f faults, need n > 3f to solve problem
- Various algorithms exist to solve problem
  - Original algorithm has #messages exponential in n
  - Newer algorithms have message complexity O(n^2)
    - » One from MIT, for instance (Castro and Liskov, 1999)
- Use of BFT (Byzantine Fault Tolerance) algorithm
  - Allow multiple machines to make a coordinated decision even if some subset of them (< n/3) are malicious



Trusted Computing

Trusted Computing

- Problem: Can't trust that software is correct
  - Viruses/Worms install themselves into kernel or system without users knowledge
  - Rootkit: software tools to conceal running processes, files or system data, which helps an intruder maintain access to a system without the user's knowledge
  - How do you know that software won't leak private information or further compromise user's access?
- A solution: What if there were a secure way to validate all software running on system?
  - Idea: Compute a cryptographic hash of BIOS, Kernel, crucial programs, etc.
  - Then, if hashes don't match, know have problem
- Further extension:
  - Secure attestation: ability to *prove* to a remote party that local machine is running correct software
  - Reason: allow remote user to avoid interacting with compromised system
- Challenge: How to do this in an unhackable way
  - Must have hardware components somewhere

TCPA: Trusted Computing Platform Alliance

- Idea: Add a Trusted Platform Module (TPM)
- Founded in 1999: Compaq, HP, IBM, Intel, Microsoft
- Currently more than 200 members
- Changes to platform
  - Extra: Trusted Platform Module (TPM)
  - Software changes: BIOS + OS
- Main properties
  - Secure bootstrap
  - Platform attestation
  - Protected storage
- Microsoft version:
  - Palladium
  - Note quite same: More extensive hardware/software system

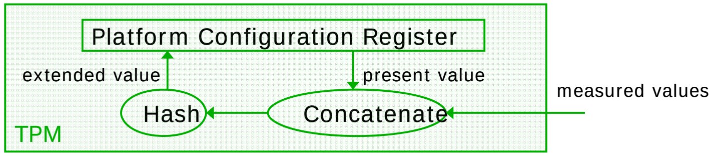
ATMEL TPM Chip
(Used in IBM equipment)

Trusted Platform Module

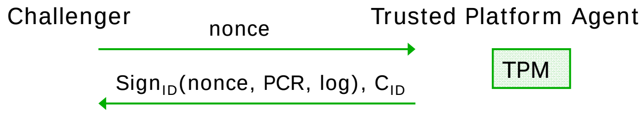
Functional Units	Non-volatile Memory	Volatile Memory
Random Num Generator	Endorsement Key (2048 Bits)	RSA Key Slot-0 … RSA Key Slot-9
SHA-1 Hash	Storage Root Key (2048 Bits)	PCR-0 PCR-15
HMAC	Owner Auth Secret(160 Bits)	Key Handles
RSA Encrypt/ Decrypt		Auth Session Handles
RSA Key Generation		

- Cryptographic operations
  - Hashing: SHA-1, HMAC
  - Random number generator
  - Asymmetric key generation: RSA (512, 1024, 2048)
  - Asymmetric encryption/ decryption: RSA
  - *Symmetric encryption/ decryption: DES, 3DES (AES)*
- Tamper resistant (hash and key) storage

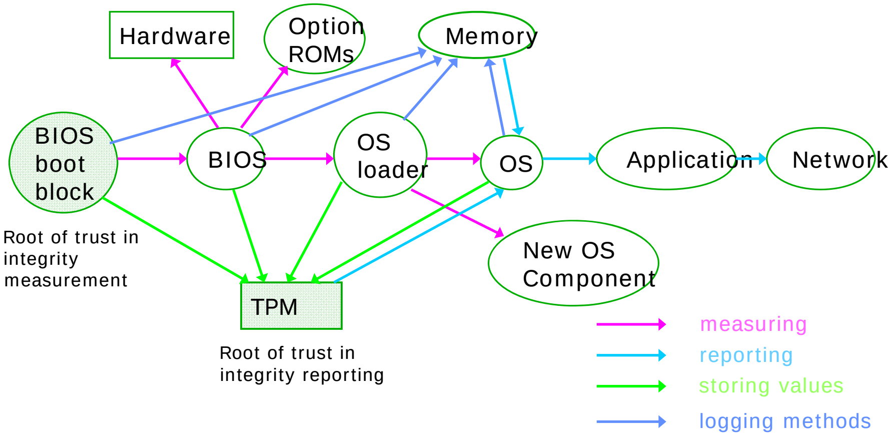
TCPA: PCR Reporting Value



- Platform Configuration Registers (PCR0-16)
  - Reset at boot time to well defined value
  - Only thing that software can do is give new measured value to TPM
    - » TPM takes new value, concatenates with old value, then hashes result together for new PCR
- Measuring involves hashing components of software
- Integrity reporting: report the value of the PCR
  - Challenge-response protocol:



TCPA: Secure bootstrap

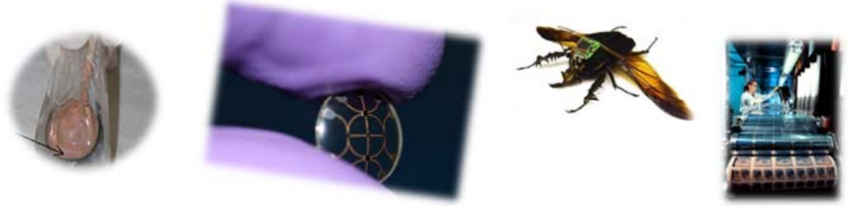


Implications of TPM Philosophy?

- Could have great benefits
  - Prevent use of malicious software
  - Parts of OceanStore would benefit
- What does "trusted computing" really mean?
  - You are forced to trust hardware to be correct!
  - Could also mean that user is not trusted to install their own software
- Many in the security community have talked about potential abuses
  - These are only theoretical, but very possible
  - Software fixing
    - » What if companies prevent user from accessing their websites with non-Microsoft browser?
    - » Possible to encrypt data and only decrypt if software still matches ⇒ Could prevent display of .doc files except on Microsoft versions of software
  - Digital Rights Management (DRM):
    - » Prevent playing of music/video except on accepted players
    - » Selling of CDs that only play 3 times?

The Swarm at the Edge of the Cloud

Vision 2025



- Integrated components will be approaching molecular limits and/or may cover complete walls
- Every object will have a wireless connection, hence leading to trillions of connected devices,
- Opportunistically collaborating to present unique experiences or to fulfill common goals

What will it Enable?
The Birth of the Swarm

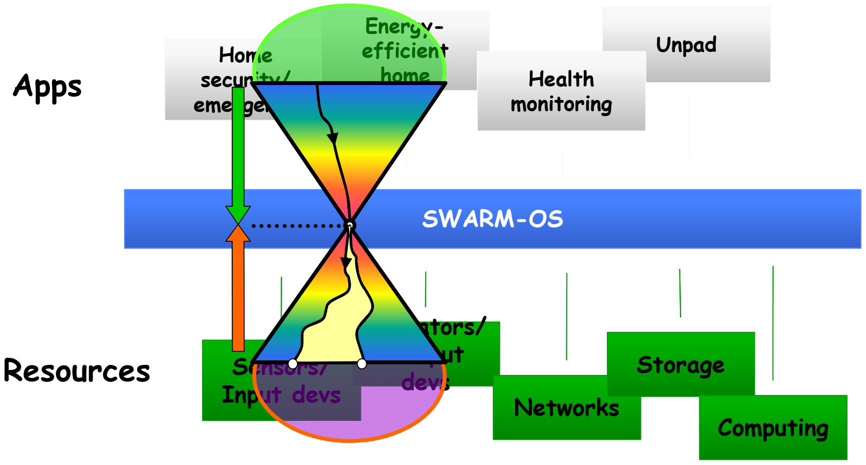
The Swarm at The Edge of the Cloud

Mobile
Access & Relay
The Swarm
TRILLIONS OF
CONNECTED DEVICES
The Cloud

[J. Rabaey, ASPDAC'08]

The Missing Link

An open platform accessible to everyone!



SWARM-OS: A mediation layer that discovers resources and connects them with applications

2010s Question:

"How to interact with information in world where enriched senses and interfaces are omnipresent?"

Mobiles to disappear or unravel! The unPad*

Blurring the boundaries between the physical and the cyber world

Towards (Human-)Aware Devices

➡ Desktops ➡ Laptops ➡ Handhelds ➡ unPads

- "Pad" goes away, but functionality (plus more) stays: enriched and unpackaged I/O, communication, computation, storage.
- People seamlessly interact with content, environment and one another through of collection of interconnected sensors and actuators.
  - Sensors and actuators opportunistically cluster as needed for a particular functionality.

unPads Coming to Life …

LCD contact lens Ghent

Google Glasses

Corning
A Day made of glass

From Interaction to Action Swarms

[C. Tomlin, UCB]

[V. Kumar, U.Penn]

[M. Maharbiz, UCB]

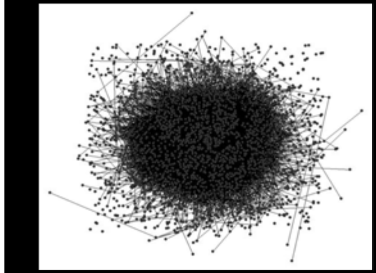
The Swarm … What does it take?

- Providing ubiquitous wireless connectivity at last
- Managing the swarm and its resources
- Maximizing experience, reliability, safety and security
- Seemless integration with cloud (and the "FOG")

A Hard and Complex Problem!

Distributed, many, heterogeneous, dynamic …

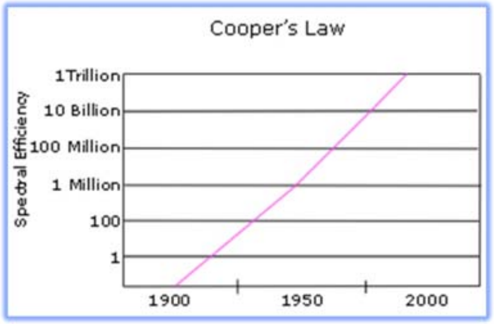
Adopt a "Swarm Perspective"



The function is in the swarm, not in the individual components
Use components opportunistically based on availability
Exploit the "power of numbers"

Get Better with Large Numbers

Wireless Capacity Doubled Every 30 Months Since 1900 *



Million-fold capacity increase since 1957

25x from wider spectrum,
5x by dividing spectrum into smaller slices,
5x by designing better modulation schemes,
1600x from reduced cell sizes and transmit distance.

Biggest gain in next decade to come from smaller cells!

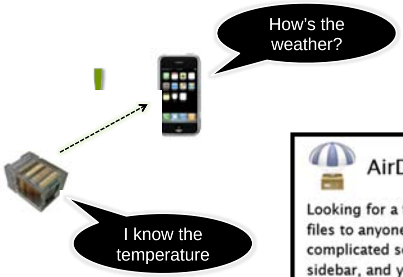
Message: The Swarm offers an unique opportunity

[M. Cooper, www.arraycom.com]

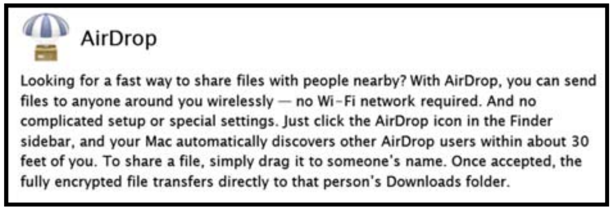
Exploiting Locality/Proximity

The peer-to-peer challenge

How to know if two nodes are even interested in talking?



Dedicated "stovepipe" solutions



AirDrop

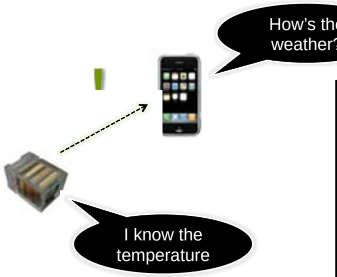
Looking for a fast way to share files with people nearby? With AirDrop, you can send files to anyone around you wirelessly — no Wi-Fi network required. And no complicated setup or special settings. Just click the AirDrop icon in the Finder sidebar, and your Mac automatically discovers other AirDrop users within about 30 feet of you. To share a file, simply drag it to someone's name. Once accepted, the fully encrypted file transfers directly to that person's Downloads folder.

Based on WiFi Direct

Exploiting Locality/Proximity

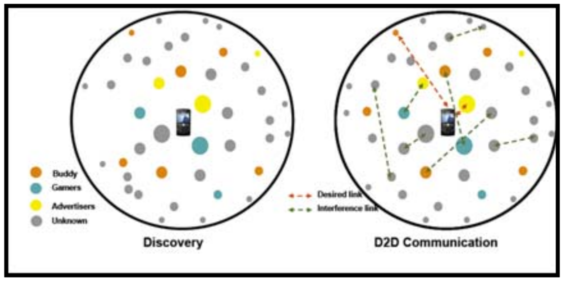
The peer-to-peer challenge

How to know if two nodes are even interested in talking?



Alternative approach:

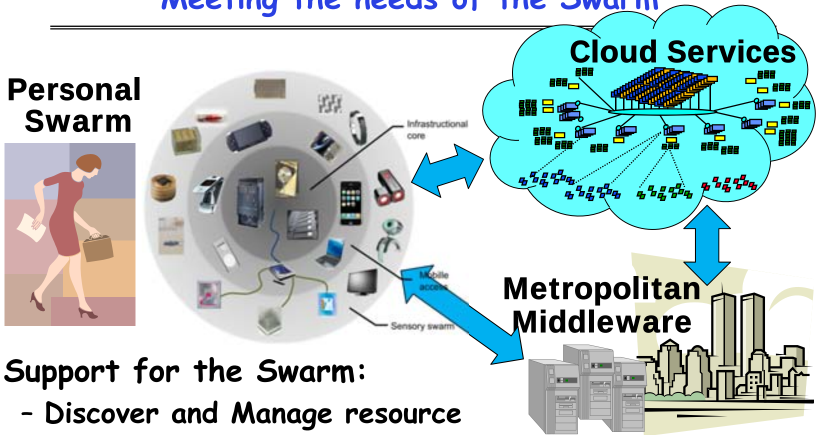
- Cut through the layers!



Example: Qualcomm FlashlinQ P2P protocol

Physical layer beaconing enables proximity and interest detection

Meeting the needs of the Swarm

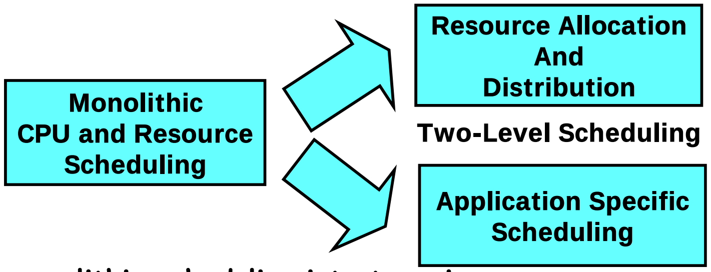


- Support for the Swarm:
  - Discover and Manage resource
  - Integrate sensors, portable devices, cloud components
  - Guarantee responsiveness, real-time behavior, throughput
  - Self-adapt to adjust for failure and provide performance predictability
  - Secure, high-performance, durable, available data

Examples

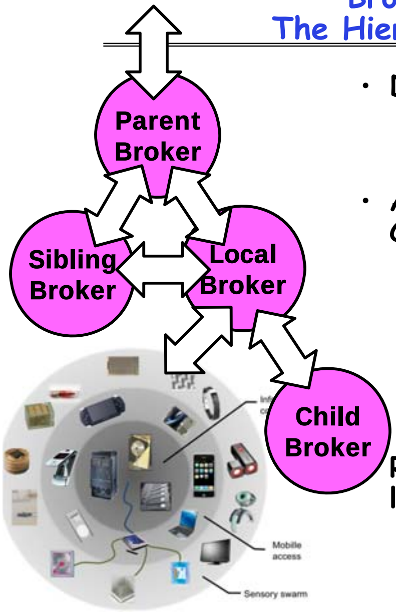
- eWallpaper:
  - Real-Time scheduling of resources
  - Secure loading of code
  - Privacy maintenance of collected information, communication
- Teleconference on nearest wall:
  - Automatic location of resources
    - » Display, Microphone, Camera, Routers
    - » Resources for transcoding, audio transcription
    - » Positional tracking
  - QoS-guaranteed network path to other side
- UnPad:
  - Resource location and allocation
    - » Displays, Microphones, Cameras, etc
    - » High-performance streaming of data from the network
  - ID-Based personalization
    - » RFID, Cellphone connection, other methods for root keys
    - » Targeted advertisement, personalized focus on
  - Deep archival storage ⇒ permanent digital history of activity

Separating Resource Allocation from Resource Usage


Monolithic CPU and Resource Scheduling
Resource Allocation And Distribution
Two-Level Scheduling
Application Specific Scheduling

- Split monolithic scheduling into two pieces:
  - Course-Grained Resource Allocation and Distribution
    - » Chunks of resources (CPUs, Memory Bandwidth, QoS to Services)
    - » Ultimately a hierarchical process negotiated with service providers
  - Fine-Grained (User-Level) Application-Specific Scheduling
    - » Applications allowed to utilize their resources in any way they see fit
    - » Performance Isolation: Other components of the system cannot interfere with Cells use of resources

Brokering Service: The Hierarchy of Ownership

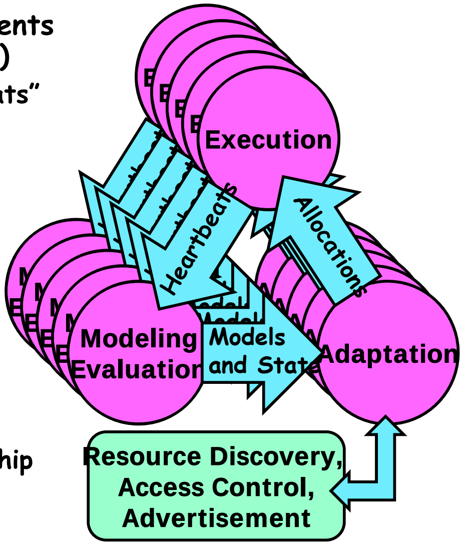


- Discover Resources in "Domain"
  - Devices, Services, Other Brokers
  - Constraints: Ownership, Access Control
- Allocate and Distribute Resources to Components that need them
  - Dynamically optimize execution
  - Hand out Service-Level Agreements (SLAs) to Software Components
  - Deny admission to application components when violate existing agreements

Resources described via declarative language: properties + requirements
  - Model of cyber-physical interactions
  - Requirements for usage
  - Constraints placed on other resources

Resource Allocation

- Goal: Meet the QoS requirements of a software component (Cell)
  - Application-specific "heartbeats" and system-level monitoring
  - Dynamic exploration of performance space to find operation points
  - Meet constraints imposed by other elements of system
- Complications:
  - Many components with conflicting requirements
  - Finite Resources
  - Hierarchy of resource ownership
  - Context-dependent resource availability
  - Stability, Efficiency, Rate of Convergence, Local Minima …


Execution

Heartbeats

Allocations

Modeling Evaluation

Models and State

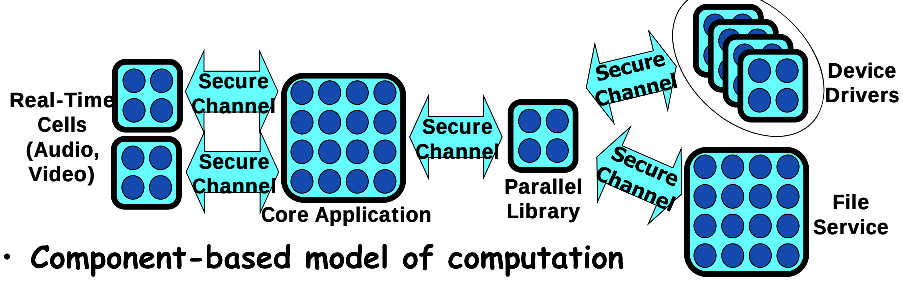
Adaptation

Resource Discovery, Access Control, Advertisement

Resource Container: the Cell

- Properties of a Cell
  - A user-level software component with guaranteed resources
  - Has full control over resources it owns ("Bare Metal")
  - Contains at least one memory protection domain (possibly more)
  - Contains a set of secured channel endpoints to other Cells
  - Hardware-enforced security context to protect the privacy of information and decrypt information (a Hardware TCB)
- Each Cell schedules its resources exclusively with application-specific user-level schedulers
  - Gang-scheduled hardware thread resources ("Harts")
  - Virtual Memory mapping and paging
  - Storage and Communication resources
    - » Cache partitions, memory bandwidth, power or energy
  - Use of Guaranteed fractions of system services

Applications are Interconnected Graphs of Services


Real-Time Cells (Audio, Video)

Secure Channel

Secure Channel

Core Application

Secure Channel

Parallel Library

Secure Channel

Device Drivers

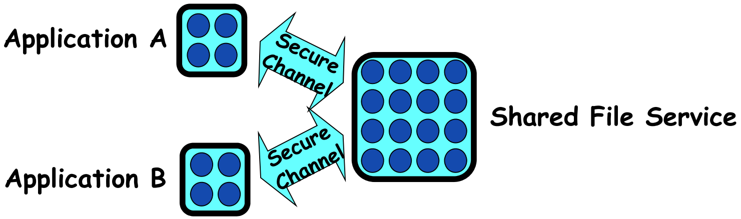
Secure Channel

File Service

- Component-based model of computation
  - Applications consist of interacting components
  - Explicitly asynchronous/non-blocking
  - Components may be local or remote
- Channel Interface ⇒ Service API, Security Boundary
  - Channels are points at which data may be compromised
  - Channels define points for QoS constraints
  - Fault tolerance and adaptation by evolving connections

Impact on the Programmer

- Connected graph of Cells ⇔ Object-Oriented Programming
  - Lowest-Impact: Wrap a functional interface around channel
    - » Cells hold "Objects", Secure channels carry RPCs for "method calls"
  - Greater Parallelism: Event triggered programming
- Applications compiled from abstract graph description
  - Independent of location or identity of services
- Shared services complicate resource isolation:
  - How to ensure each client gets well-defined fraction of service?
  - Distributed resource attribution (application as distributed graph)


Application A

Secure Channel

Shared File Service

Secure Channel

Application B

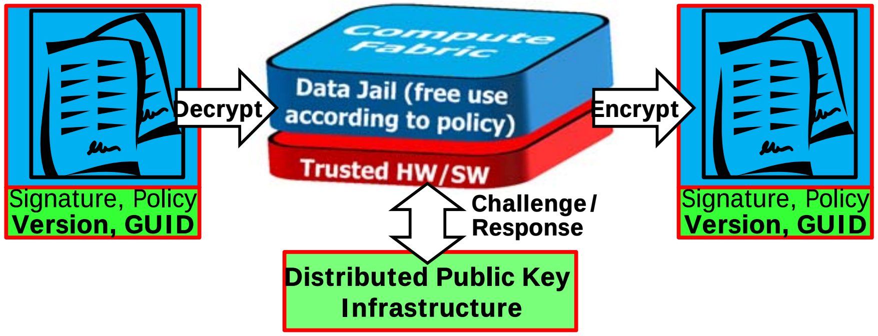
Security and Privacy

Open architectures with dynamically recruitable sensors open enormous security and privacy concerns. But recent innovations show that data aggregation and networking can be used to *enhance* security and privacy.

E.g., Differential privacy [Dwork et al., 2006] provides a framework for removing side-channel information that can be derived by cross-correlating data sets.

In another example, tighter coupling of time bases in distributed systems (time synchronization) provides a framework for detecting and countering denial of service attacks.

Secure Cell: Portable Secure Data



Signature, Policy
Version, GUID

Decrypt

Compute Fabric

Data Jail (free use according to policy)

Trusted HW/SW

Encrypt

Signature, Policy
Version, GUID

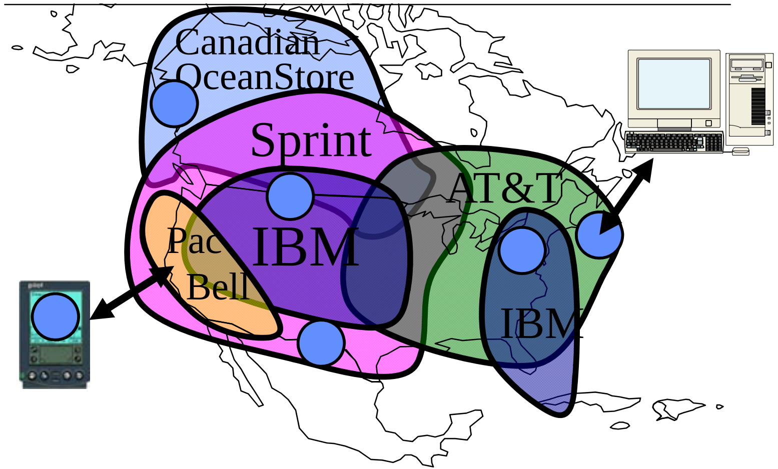
Challenge/
Response

Distributed Public Key Infrastructure

- Secure Cell: Security Context as a resource
  - Data is signed, has attached policy, Optionally encrypted
  - Unwrappable only in correct trusted environment
  - Data automatically reencrypted on exit
  - Hardware TCB: guarantees against faulty/malicious software
- What about durability? Performance? Availability?

Oceanstore

Utility-based Infrastructure



- Data service provided by storage federation
- Cross-administrative domain
- Contractual Quality of Service ("someone to sue")

OceanStore: Everyone's Data, One Big Utility

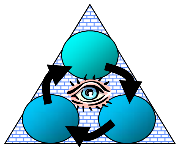
"The data is just out there"

- How many files in the OceanStore?
  - Assume 10^{10} people in world
  - Say 10,000 files/person (very conservative?)
  - So 10^{14} files in OceanStore!
  - If 1 gig files (ok, a stretch), get 1 mole of bytes! (or a Yotta-Byte if you are a computer person)

Truly impressive number of elements…
… but small relative to physical constants

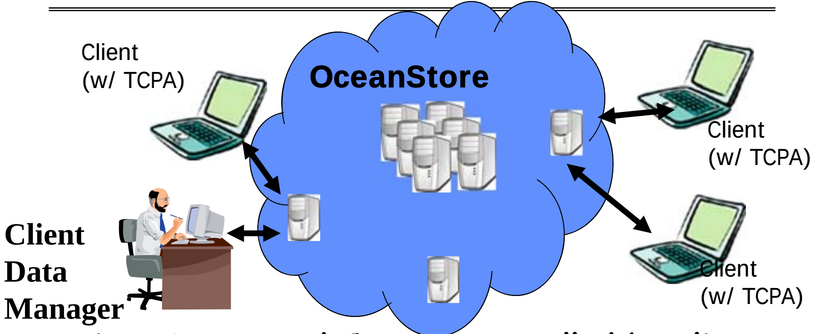
Key Observation: Want Automatic Maintenance

- Can't possibly manage billions of servers by hand!
- System should automatically:
  - Adapt to failure
  - Exclude malicious elements
  - Repair itself
  - Incorporate new elements



- System should be secure and private
  - Encryption, authentication
- System should preserve data over the long term (*accessible* for 1000 years):
  - Geographic distribution of information
  - New servers added from time to time
  - Old servers removed from time to time
  - Everything just works

Example: Secure Object Storage



- Security: Access and Content controlled by client
  - Privacy through data encryption
  - Optional use of cryptographic hardware for revocation
  - Authenticity through hashing and active integrity checking
- Flexible self-management and optimization:
  - Performance and durability
  - Efficient sharing

OceanStore Assumptions

Peer-to-peer

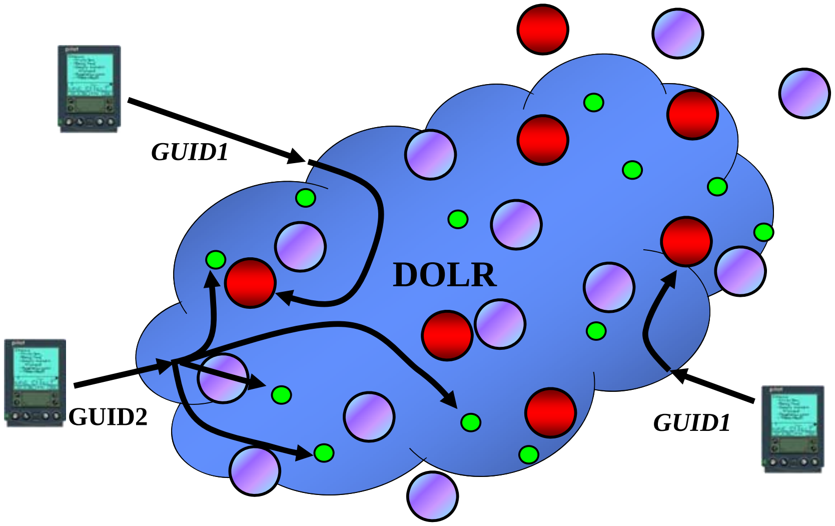
- Untrusted Infrastructure:
  - The OceanStore is comprised of untrusted components
  - Individual hardware has finite lifetimes
  - All data encrypted within the infrastructure
- Mostly Well-Connected:
  - Data producers and consumers are connected to a high-bandwidth network most of the time
  - Exploit multicast for quicker consistency when possible
- Promiscuous Caching:
  - Data may be cached anywhere, anytime

Quality-of-Service

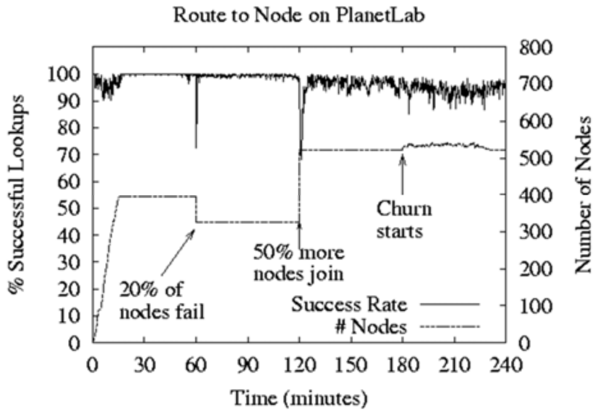
- Responsible Party:
  - Some organization (*i.e. service provider)* guarantees that your data is consistent and durable
  - Not trusted with *content* of data, merely its *integrity*

Peer-to-Peer for Data Location

Peer-to-Peer in OceanStore: DOLR
(Decentralized Object Location and Routing)



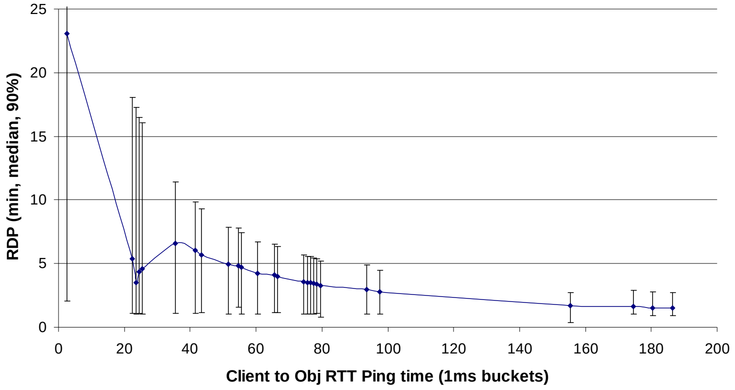
Stability under extreme circumstances



(May 2003: 1.5 TB over 4 hours)

DOLR Model generalizes to many simultaneous apps

Object Location with Tapestry DOLR

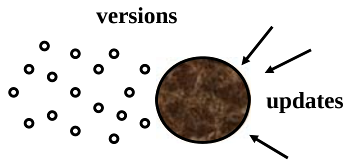


Peak at Oceanstore Mechanisms

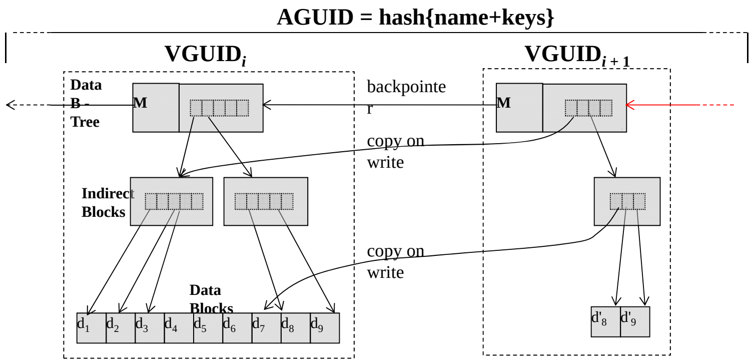
OceanStore Data Model

- Versioned Objects
  - Every update generates a new version
  - Can always go back in time (Time Travel)
- Each Version is Read-Only
  - Can have permanent name
  - Much easier to repair
- An Object is a signed mapping between permanent name and latest version
  - Write access control/integrity involves managing these mappings

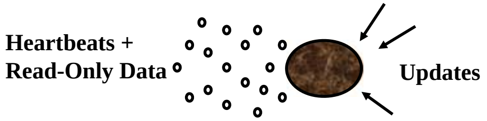
Comet Analogy



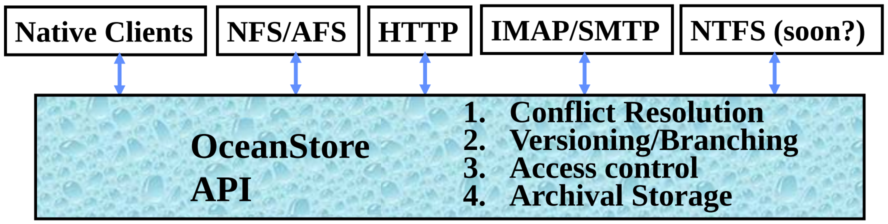
Self-Verifying Objects



♥Heartbeat: {AGUID,VGUID, Timestamp}$_{\text{signed}}$



OceanStore API: Universal Conflict Resolution



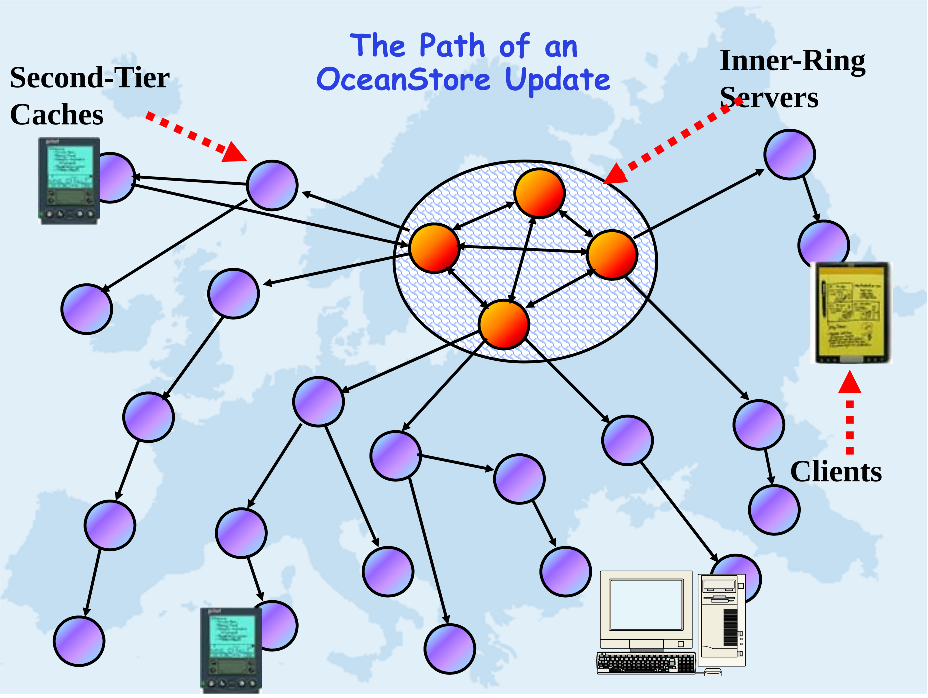
1. Conflict Resolution
2. Versioning/Branching
3. Access control
4. Archival Storage

- Consistency is form of optimistic concurrency
  - Updates contain predicate-action pairs
  - Each predicate tried in turn:
    - » If none match, the update is aborted
    - » Otherwise, action of first true predicate is applied
- Role of Responsible Party (RP):
  - Updates submitted to RP which chooses total order
- This is powerful enough to synthesize:
  - ACID database semantics
  - release consistency (build and use MCS-style locks)
  - Extremely loose (weak) consistency

Two Types of OceanStore Data

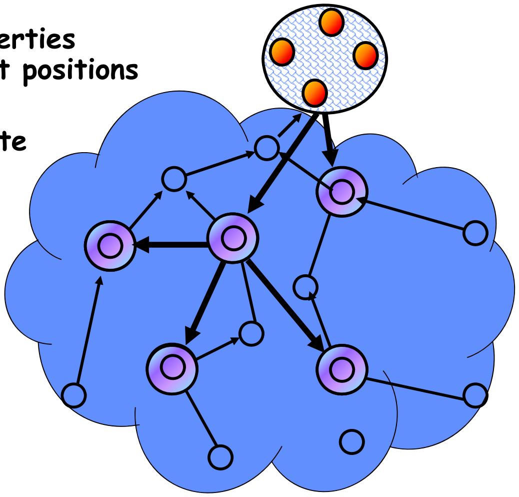
- *Active Data:* "Floating Replicas"
  - Per object virtual server
  - Interaction with other replicas for consistency
  - May appear and disappear like bubbles
- *Archival Data:* OceanStore's Stable Store
  - m-of-n coding: Like hologram
    - » Data coded into *n* fragments, any *m* of which are sufficient to reconstruct (e.g m=16, n=64)
    - » Coding overhead is proportional to n÷m (e.g 4)
    - » Other parameter, *rate*, is 1/overhead
  - Fragments are cryptographically self-verifying
- Most data in the OceanStore is archival!

The Path of an OceanStore Update

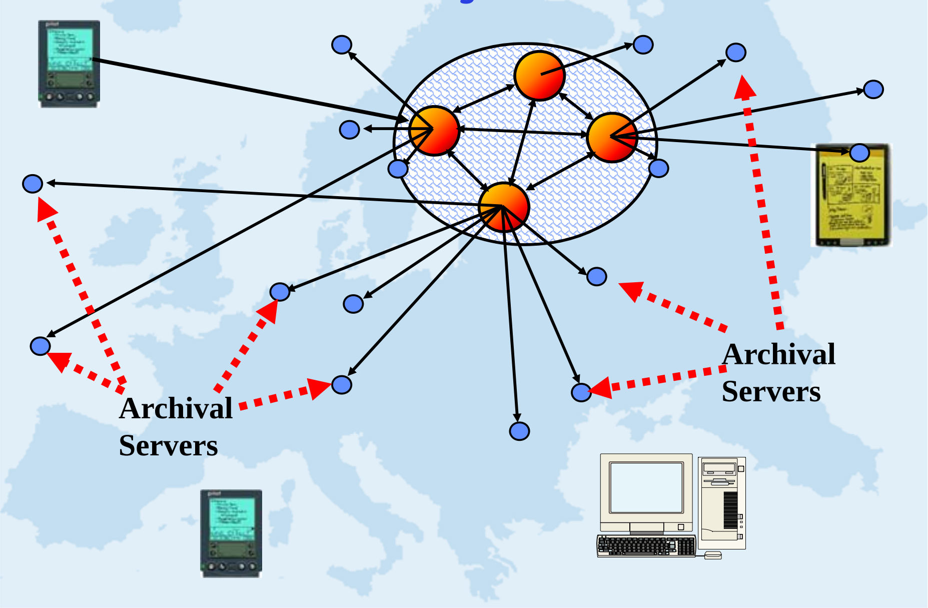


Self-Organizing Soft-State Replication

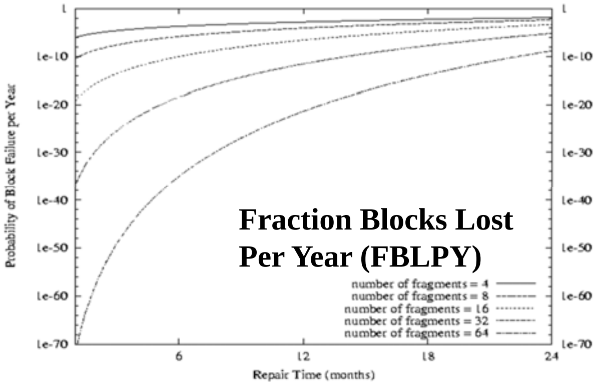
- Simple algorithms for placing replicas on nodes in the interior
  - Intuition: locality properties of Tapestry help select positions for replicas
  - Tapestry helps associate parents and children to build multicast tree
- Preliminary results encouraging
- Current Investigations:
  - Game Theory
  - Thermodynamics



Archival Dissemination of Fragments



Aside: Why erasure coding? High Durability/overhead ratio!



Fraction Blocks Lost Per Year (FBLPY)

- Exploit law of large numbers for durability!
- 6 month repair, FBLPY:
  - Replication: 0.03
  - Fragmentation: 10-35

Extreme Durability?

- Exploiting Infrastructure for Repair
  - DOLR permits efficient heartbeat mechanism to notice:
    - » Servers going away for a while
    - » Or, going away forever!
  - Continuous sweep through data also possible
  - Erasure Code provides Flexibility in Timing
- Data transferred from physical medium to physical medium
  - No "tapes decaying in basement"
  - Information becomes fully Virtualized
- Thermodynamic Analogy: Use of Energy (supplied by servers) to Suppress Entropy

Differing Degrees of Responsibility

- Inner-ring provides quality of service
  - Handles of live data and write access control
  - Focus utility resources on this vital service
  - Compromised servers must be detected quickly
- Caching service can be provided by anyone
  - Data encrypted and self-verifying
  - Pay for service "Caching Kiosks"?
- Archival Storage and Repair
  - Read-only data: easier to authenticate and repair
  - Tradeoff redundancy for responsiveness
- Could be provided by different companies!

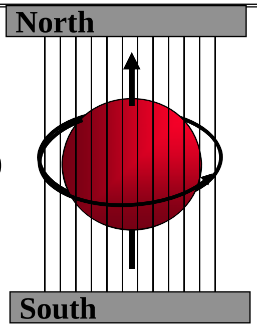
Quantum Computing

Can we Use Quantum Mechanics to Compute?

- Weird properties of quantum mechanics?
  - Quantization: Only certain values or orbits are good
    - » Remember orbitals from chemistry???
  - Superposition: Schizophrenic physical elements don't quite know whether they are one thing or another
- All existing digital abstractions try to eliminate QM
  - Transistors/Gates designed with classical behavior
  - Binary abstraction: a "1" is a "1" and a "0" is a "0"
- Quantum Computing:
  Use of Quantization and Superposition to compute.
- Interesting results:
  - Shor's algorithm: factors in polynomial time!
  - Grover's algorithm: Finds items in unsorted database in time proportional to square-root of n.

Quantization: Use of "Spin"

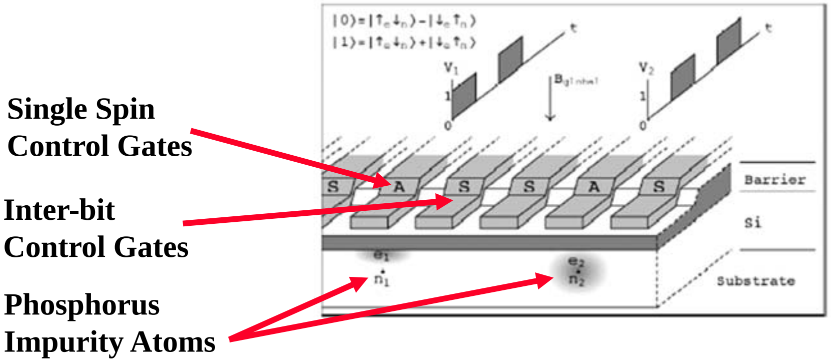
Spin ½ particle:
(Proton/Electron)



Representation:
|0> or |1>

- Particles like Protons have an intrinsic "Spin" when defined with respect to an external magnetic field
- Quantum effect gives "1" and "0":
  - Either spin is "UP" or "DOWN" nothing between

Kane Proposal II (First one didn't quite work)



Single Spin
Control Gates

Inter-bit
Control Gates

Phosphorus
Impurity Atoms

- Bits Represented by combination of proton/electron spin
- Operations performed by manipulating control gates
  - Complex sequences of pulses perform NMR-like operations
- Temperature < 1° Kelvin!

Now add Superposition!

- The bit can be in a combination of "1" and "0":
  - Written as: $\Psi = C_0|0> + C_1|1>$
  - The C's are *complex numbers!*
  - Important Constraint: $|C_0|^2 + |C_1|^2 = 1$
- If *measure* bit to see what looks like,
  - With probability $|C_0|^2$ we will find |0> (say "UP")
  - With probability $|C_1|^2$ we will find |1> (say "DOWN")
- Is this a real effect? Options:
  - This is just statistical – given a large number of protons, a fraction of them ($|C_0|^2$) are "UP" and the rest are down.
  - This is a real effect, and the proton is really both things until you try to look at it
- Reality: second choice!
  - There are experiments to prove it!

Implications: A register can have many values

- Implications of superposition:
  - An *n*-bit register can have 2^n values simultaneously!
  - 3-bit example:

$$\Psi = C_{000}|000> + C_{001}|001> + C_{010}|010> + C_{011}|011> + C_{100}|100> + C_{101}|101> + C_{110}|110> + C_{111}|111>$$

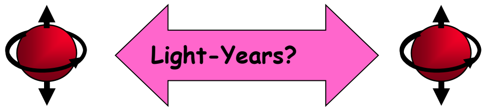
- Probabilities of measuring all bits are set by coefficients:
  - So, prob of getting |000> is $|C_{000}|^2$, etc.
  - Suppose we measure only one bit (first):
    - » We get "0" with probability: $P_0 = |C_{000}|^2 + |C_{001}|^2 + |C_{010}|^2 + |C_{011}|^2$ Result: $\Psi = (C_{000}|000> + C_{001}|001> + C_{010}|010> + C_{011}|011>)$
    - » We get "1" with probability: $P_1 = |C_{100}|^2 + |C_{101}|^2 + |C_{110}|^2 + |C_{111}|^2$ Result: $\Psi = (C_{100}|100> + C_{101}|101> + C_{110}|110> + C_{111}|111>)$
- Problem: Don't want environment to *measure* before ready!
  - Solution: Quantum Error Correction Codes!

Spooky action at a distance

- Consider the following simple 2-bit state:

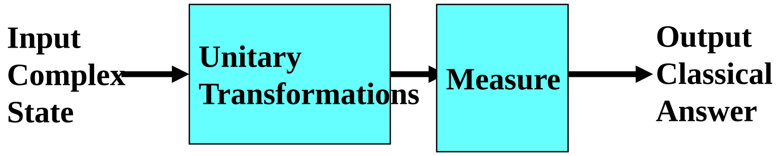
$$\Psi = C_{00}|00> + C_{11}|11>$$

  - Called an "EPR" pair for "Einstein, Podolsky, Rosen"
- Now, separate the two bits:



- If we measure one of them, it instantaneously sets other one!
  - Einstein called this a "spooky action at a distance"
  - In particular, if we measure a |0> at one side, we get a |0> at the other (and vice versa)
- Teleportation
  - Can "pre-transport" an EPR pair (say bits X and Y)
  - Later to transport bit A from one side to the other we:
    - » Perform operation between A and X, yielding two classical bits
    - » Send the two bits to the other side
    - » Use the two bits to operate on Y
    - » Poof! State of bit A appears in place of Y

Model? Operations on coefficients + measurements



Input Complex State → Unitary Transformations → Measure → Output Classical Answer

- Basic Computing Paradigm:
  - Input is a register with superposition of many values
    - » Possibly all 2^n inputs equally probable!
  - Unitary transformations compute on coefficients
    - » Must maintain probability property (sum of squares = 1)
    - » Looks like doing computation on all 2^n inputs simultaneously!
  - Output is one result attained by measurement
- If do this poorly, just like probabilistic computation:
  - If 2^n inputs equally probable, may be 2^n outputs equally probable.
  - After measure, like picked random input to classical function!
  - All interesting results have some form of "fourier transform" computation being done in unitary transformation

Security of Factoring

- The Security of RSA Public-key cryptosystems depends on the difficult of factoring a number N=pq (product of two primes)
  - Classical computer: sub-exponential time factoring
  - Quantum computer: polynomial time factoring
- Shor's Factoring Algorithm (for a quantum computer)

Easy 1) Choose random x : $2 \le x \le N\text{-}1$.

Easy 2) If gcd(x,N) ≠ 1, Bingo!

Hard 3) Find smallest integer r : $x^r \equiv 1 \pmod{N}$

Easy 4) If r is odd, GOTO 1

Easy 5) If r is even, $a = x^{r/2} \pmod{N} \Rightarrow (a-1)\times(a+1) = kN$

Easy 6) If $a = N\text{-}1$ GOTO 1

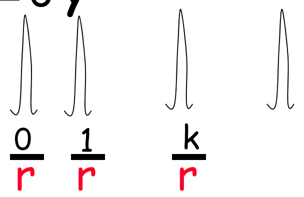
Easy 7) ELSE gcd($a \pm 1, N$) is a non trivial factor of N.

Shor's Factoring Algorithm

$$\sum_k |k\rangle|1\rangle \rightarrow \sum_k |k\rangle|x^k\rangle$$

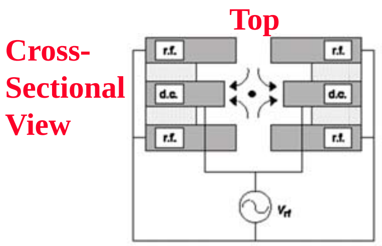
$$= \sum_{w=0}^{r-1}\sum_y |w+ry\rangle|x^w\rangle$$

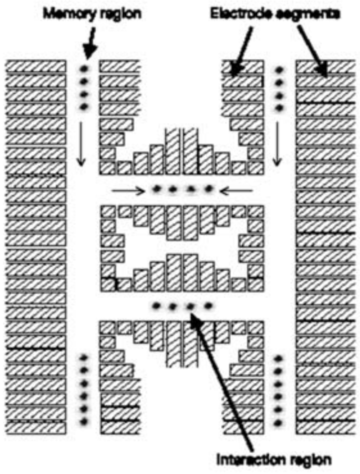
$$\xrightarrow{\text{Quantum Fourier Transform}} \sum_{w=0}^{r-1}\Big(\qquad \Big)|x^w\rangle$$



- Finally: Perform measurement
  - Find out r with high probability
  - Get |y>|a^w'> where y is of form k/r and w' is related

ION Trap Quantum Computer: Promising technology



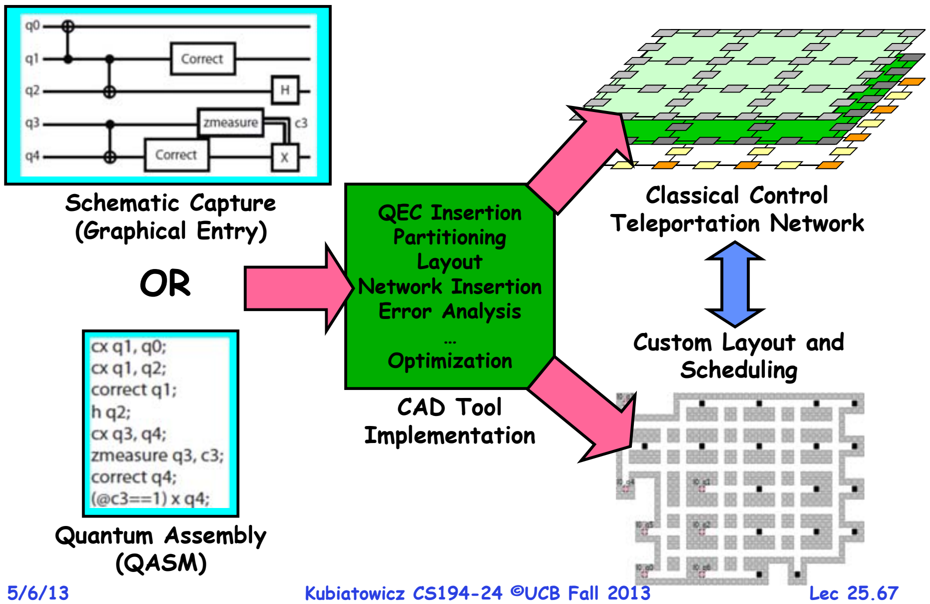


Top View

Proposal: NIST Group

- IONS of Be+ trapped in oscillating quadrature field
  - Internal electronic modes of IONS used for quantum bits
  - MEMs technology
  - Target? 50,000 ions
  - ROOM Temperature!
- Ions moved to interaction regions
  - Ions interactions with one another moderated by lasers

Vision of Quantum Circuit Design



Schematic Capture (Graphical Entry)

OR

```
cx q1, q0;
cx q1, q2;
correct q1;
h q2;
cx q3, q4;
zmeasure q3, c3;
correct q4;
(@c3==1) x q4;
```

Quantum Assembly (QASM)

QEC Insertion
Partitioning
Layout
Network Insertion
Error Analysis
…
Optimization

CAD Tool Implementation

Classical Control
Teleportation Network

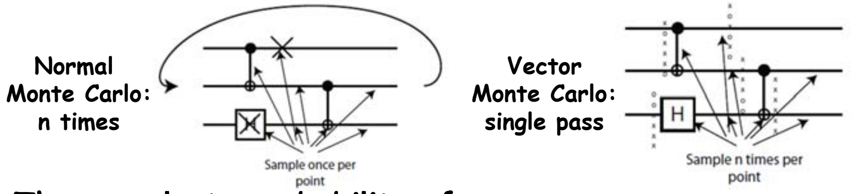
Custom Layout and Scheduling

Important Measurement Metrics

- Traditional CAD Metrics:
  - Area
    - » What is the total area of a circuit?
    - » Measured in macroblocks (ultimately μm^2 or similar)
  - Latency ($\text{Latency}_{\text{single}}$)
    - » What is the total latency to compute circuit *once*
    - » Measured in seconds (or μs)
  - Probability of Success (P_{success})
    - » Not common metric for classical circuits
    - » Account for occurrence of errors and error correction
- Quantum Circuit Metric: ADCR
  - Area-Delay to Correct Result: Probabilistic Area-Delay metric
  - $\text{ADCR} = \text{Area} \times E(\text{Latency}) = \dfrac{\text{Area} \times \text{Latency}_{\text{single}}}{P_{\text{success}}}$
  - $\text{ADCR}_{\text{optimal}}$: Best ADCR over all configurations
- Optimization potential: Equipotential designs
  - Trade Area for lower latency
  - Trade lower probability of success for lower latency

How to evaluate a circuit?

- First, generate a physical instance of circuit
  - Encode the circuit in one or more QEC codes
  - Partition and layout circuit: Dependant of layout heuristics!
    - » Create a physical layout and scheduling of bits
    - » Yields area and communication cost

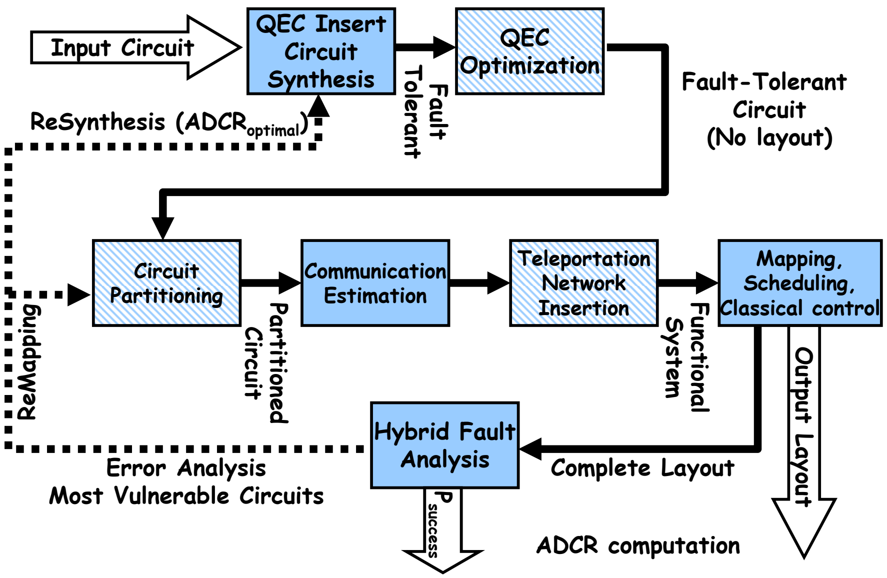


Normal Monte Carlo: n times

Vector Monte Carlo: single pass

- Then, evaluate probability of success
  - Technique that works well for depolarizing errors: Monte Carlo
    - » Possible error points: Operations, Idle Bits, Communications
  - Vectorized Monte Carlo: n experiments with one pass
  - Need to perform hybrid error analysis for larger circuits
- Finally – Compute ADCR for particular result
  - Repeat as necessary by varying parameters to generate $\text{ADCR}_{\text{optimal}}$

Quantum CAD flow



Input Circuit

QEC Insert Circuit Synthesis

Fault Tolerant

QEC Optimization

Fault-Tolerant Circuit (No layout)

ReSynthesis ($\text{ADCR}_{\text{optimal}}$)

Circuit Partitioning

Partitioned Circuit

Communication Estimation

Teleportation Network Insertion

Functional System

Mapping, Scheduling, Classical control

ReMapping

Error Analysis Most Vulnerable Circuits

Hybrid Fault Analysis

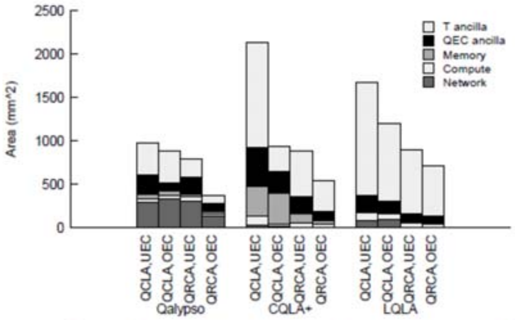
Complete Layout

Output Layout

P_{success}

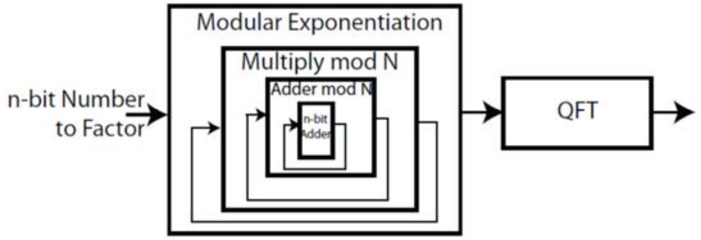
ADCR computation

Area Breakdown for Adders



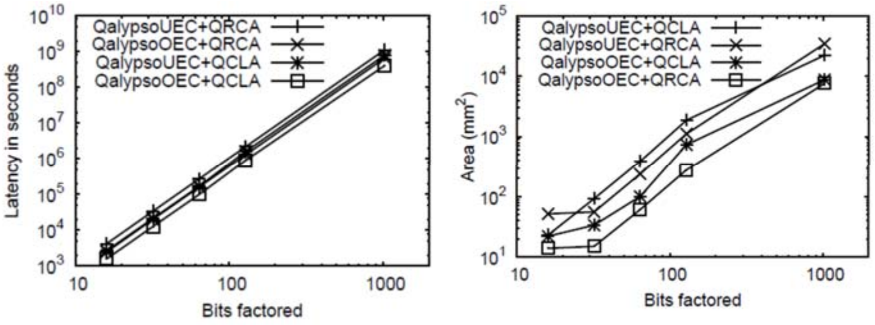
- Error Correction is *not* predominant use of area
  - Only 20-40% of area devoted to QEC ancilla
  - For Optimized Qalypso QCLA, 70% of operations for QEC ancilla generation, but only about 20% of area
- T-Ancilla generation is major component
  - Often overlooked
- Networking is significant portion of area when allowed to optimize for ADCR (30%)
  - CQLA and QLA variants didn't really allow for much flexibility

Investigating 1024-bit Shor's



- Full Layout of all Elements
  - Use of 1024-bit Quantum Adders
  - Optimized error correction
  - Ancilla optimization and Custom Network Layout
- Statistics:
  - Unoptimized version: 1.35×10^{15} operations
  - Optimized Version 1000X smaller
  - QFT is only 1% of total execution time

1024-bit Shor's Continued



- Circuits too big to compute $P_{success}$
  - Working on this problem
- Fastest Circuit: 6×10^8 seconds ~ 19 years
  - Speedup by classically computing recursive squares?
- Smallest Circuit: 7659 mm^2
  - Compare to previous *estimate* of 0.9 m^2 = 9×10^5 mm^2

Conclusion (1/2)

- Trusted Hardware
  - A secure layer of hardware that can:
    - » Generate proofs about software running on the machine
    - » Allow secure access to information without revealing keys to (potentially) compromised layers of software
  - Cannonical example: TPM
- Services for the Swarm:
  - Use of Resources negotiated hierarchically
  - Underlying Execution environment guarantees QoS
  - New Resources constructed from Old ones:
    - » Aggregate resources in combination with QoS-Aware Scheduler
    - » Result is a *new* resource that can be negotiated for
  - Continual adaptation and optimization

Conclusion (2/2)

- OceanStore properties:
  - Provides security, privacy, and integrity
  - Provides extreme durability
  - Lower maintenance cost through redundancy, continuous adaptation, self-diagnosis and repair
- Quantum Computing
  - Using interesting properties of physics to compute
  - Noise is one of the most complex aspects
  - At a stage where Computer Aided Design (CAD) makes sense
  - Quantum Circuit Metric: ADCR
    - » Area-Delay to Correct Result: Probabilistic Area-Delay metric
    - » ADCR = Area × E(Latency)
      $\text{ADCR}_{\text{optimal}}$: Best ADCR over all configurations
- Let's give a hand to Palmer – the labs wouldn't exist without him!

Good Bye!