Panel on Uncertainty

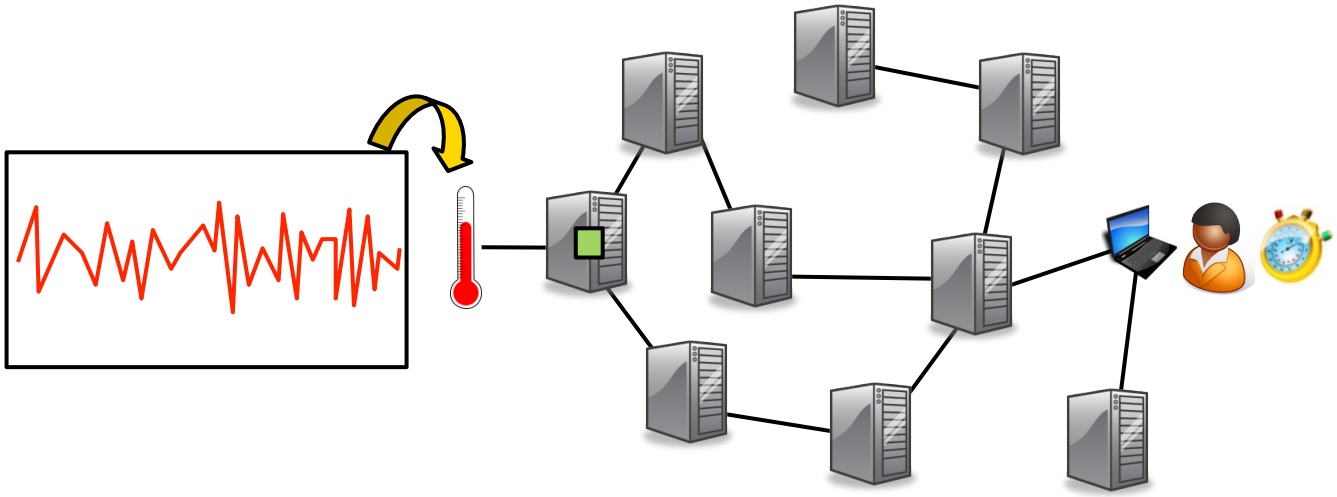
Peter M. Chen Andreas Haeberlen Z. Morley Mao Ratul Mahajan

Opening statements

- Andreas 

- Peter
- Morley
- Ratul

Uncertainty: An old "friend"



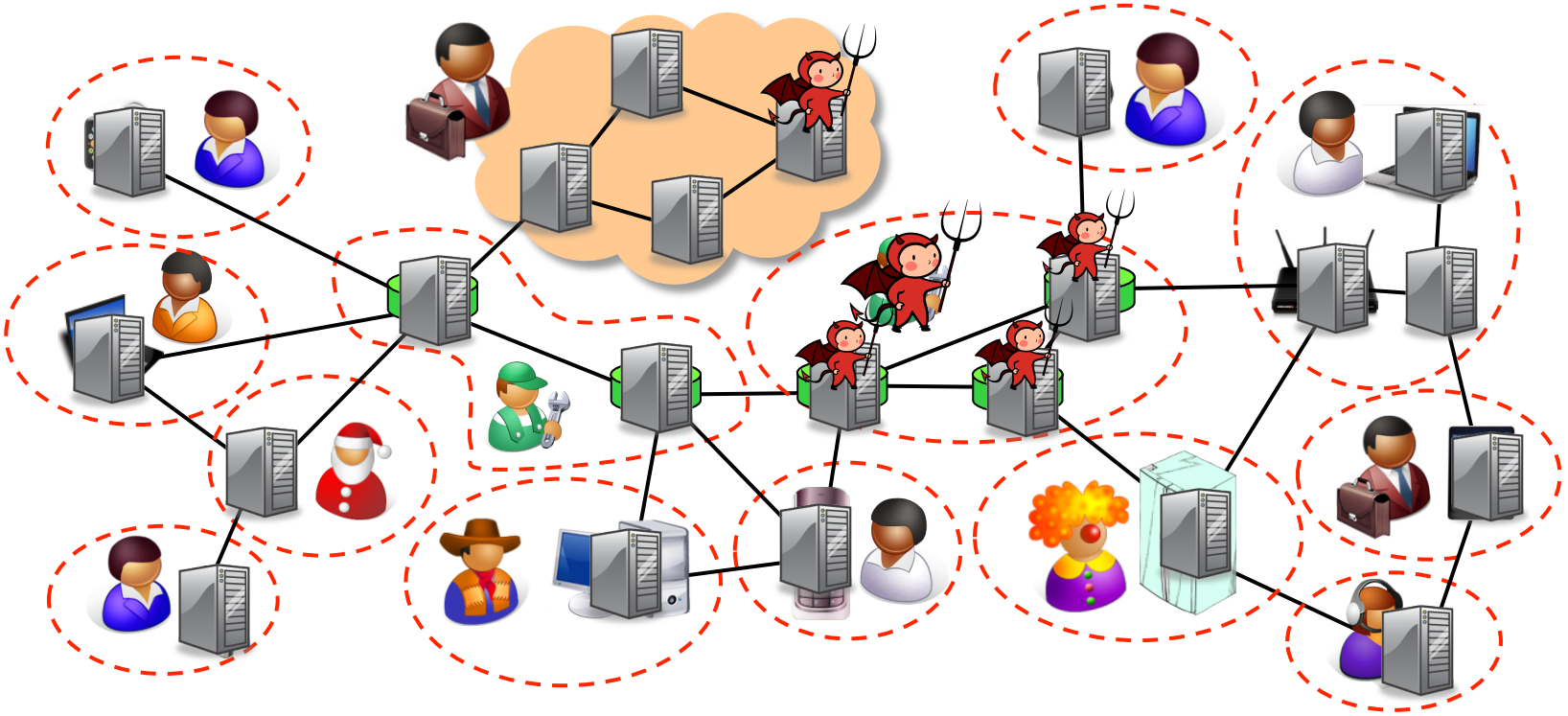
- Distributed systems have always had to deal with some uncertainty

What is changing?

- Prediction: The next generation of distributed systems will need to grapple with a greater degree of uncertainty than systems today

- Where will this uncertainty come from?
  - Trend: New models that sacrifice accuracy or determinism to get better speed, or to save energy
  - Trend: More sensors, but cheaper ones → Input quality degrades
  - Trend: Mobility and wireless links are becoming the common case rather than the exception

Trend: MAD distributed systems



- The very notion of what constitutes a distributed system is changing
  - Result: Uncertainty in the very fabric of the system
  - Parts of the system might misbehave, collude, wander off, …

Challenge: Interfaces

- Should systems expose uncertainty?
  - Option: Work hard to mask uncertainty whenever possible
  - But: In the 'offline world', we deal with uncertainty every day
  - Analogy to Butler Lampson's "Computer Security in the Real World"?
- If so, how?
  - Simplest approach: Show confidence intervals
  - Could (some) systems be allowed to 'be wrong' sometimes, especially if they fix problems quickly?

Challenge: Good abstractions

- So far, solutions tend to be very specific
  - Example: Confidence intervals for timing
  - Doesn't work for many apps, or for other kinds of uncertainty
- Can we find abstractions that work for larger classes of distributed systems?
- Can we have a toolbox of useful mechanisms for dealing with uncertainty?
  - Examples: Speculation, secure provenance, accountability, ...

Opportunity Challenge: New systems + tradeoffs

- If we treat uncertainty as a 1st-class citizen...
- can we build better systems?
  - Example: Google's Spanner system [OSDI'12]
  - Exposes timing uncertainty → global, consistent database
- can we enable interesting tradeoffs?
  - Example: Approximate computing
  - Maybe also less restrictive/expensive security?
- can we obtain surprising new properties?
  - Example: Use of uncertainty in differential privacy

Opening statements

- Andreas 
- Peter 

- Morley
- Ratul

Uncertainty is a certainty

- Old sources of uncertainty: imprecise knowledge of the future
  - Failures
  - Intrusions
  - Bugs
  - Timing of concurrent accesses
- New sources of uncertainty: imprecise knowledge of the present
  - Approximate hardware
  - Fuzzy sensors
  - Approximate queries

Case study: concurrency control

- Prevent/reduce uncertainty
  - Acquire mutex
- Mask uncertainty
  - Optimistic concurrency control
  - Detect bad events and recover
- Expose uncertainty
  - Conflict exceptions

Some research questions

- How to estimate uncertainty as it enters the system?
- How to track uncertainty as it propagates through the system?
- How to estimate uncertainty as it is computed on?
- How to represent uncertainty and uncertainty bounds?
- How to cheaply detect when uncertainty exceeds allowable bound?
- How to support distributed rollback due to uncertainty errors?
- Who is responsible for handling uncertainty?
- How to communicate uncertainty in the interface between system and applications?

Opening statements

- Andreas 
- Peter 
- Morley 

- Ratul

Uncertainty is inevitable

- Proprietary systems, unknown configurations
  - E.g., Load balancing algorithms in a middlebox are proprietary
  - E.g., Network configurations are not exposed
- Specifications (RFCs) leaving out details
  - Many variants of the TCP protocol
- Difficulties in modeling user behavior
  - Unknown workload

How to address uncertainty?

- Our models can be inaccurate
  - It's OK, we don't rely on them for correctness, but only for performance optimization
  - For security, we need to be conservative, i.e., not make assumptions that may not hold.
- Conservatively assume the worst scenario
  - Build into the system ways to deal with uncertainty, e.g., variable network conditions
  - Do not rely on modeled behavior
- Explore the tradeoffs of directly addressing the uncertainty or optimistically assuming the most common behavior

Examples

- Radio resource allocation behavior differs across cellular carriers
  - Need an abstraction to identify these differences to support efficient data scheduling.
- Malicious mobile apps may exploit side channels to violate user privacy
  - Impose the policy to actively prevent un-trusted apps from running in the background.

Questions for discussion

- What are the current approaches to deal with uncertainty? What are their limitations?
- How do we model uncertainty as a part of our design? (e.g., leverage robust optimization)
- Can security solutions for dealing with broad attack surfaces be used to address uncertainties?
- How do we measure/characterize/quantify uncertainty?

Opening statements

- Andreas 
- Peter 
- Morley 
- Ratul 

Uncertainty

Ratul Mahajan
Microsoft Research

Failed attempt at a taxonomy

Type of uncertainty	Applicable techniques
Input (workload)	Build models
Output	
Processing	
Control	
System state	
…..	

Control uncertainty

Delays or failures in devices implementing commands

Two general (?) techniques for fast, robust control

- Absolute commands
- Planned transitions (optimization)

Uncertainty in system state

What the system state is right now or will be in the future

- Changes from “below”
- Application interaction and conflicts

Two general (?) techniques for reducing uncertainty

- Continuous polling of complete state
- “Don’t care” state or multiple acceptable outputs

Chaining uncertain inferences

Applications can handle uncertainty in domain-specific manner but chaining is problematic

General (?) techniques for reducing uncertainty

Discussion: Help fill this

Type of uncertainty	Applicable techniques
Input (workload)	Build models
Output	
Processing	
Control	
System state	
…..	