CIS 700/003: Distributed Systems meet Social Networks

Instructor: Andreas Haeberlen

TTh 4:30-6pm
Chemistry 119

http://www.cis.upenn.edu/~ahae/teaching/cis700-s10/
About me

- Joined Penn in January 2010 as an assistant professor
  - PhD from Rice in 2009, short post-doc in Germany at MPI-SWS
- Affiliated with the new program in Market and Social Systems Engineering (MKSE)
- Interested in distributed systems, networking, security
- Coordinates: Levine 560, 6-6167
- Office hour: Thursdays 1:30-2:30pm
About CIS 700/003

- **My goal:** Introduce you to a new, 'hot' research area (distributed systems with multiple administrative domains)
  - Format: Seminar-style class with a term project

- **What you would have to do:**
  - **Read** one conference or workshop paper every week and submit a short review one hour before class
  - **Present** some of the papers and lead the discussion
  - **Participate** in the discussion
  - **Do a term project** (research). Deliverables: Progress report before spring break + final report at end of term + six-page project report (workshop-style)
What are we going to read?

- Research papers from top conferences:
  - SIGCOMM, NSDI, SOSP, OSDI, CCS, S&P, SIGMOD, ...

- "Do I have to be a systems (networking, security, social networks, databases) expert?"
  - No.
  - I will cover some basics (BGP, state machine replication) in two initial lectures. Papers are pretty accessible.
My plan for today

- Intro: What are MAD distributed systems?
- A tour of CIS700/003
  - Measurement and analysis
  - Building MAD systems
  - Faults and misbehavior
  - Internet crime
  - Privacy and confidentiality
  - Novel opportunities
  - Experience
- Course logistics
Example: The Internet

- System + control are distributed

Administrative domain
Routers

Alice
Example: The Internet

- System + control + information are distributed
- Social/economic connections

I have a good route

I have an okay route

Who knows how to get to YouTube?

© 2010 Andreas Haeberlen
- Distributed system + control + information
- Social/economic connections
Example: The Internet

- Distributed system + control + information
- Social/economic connections

I wish my route had been chosen.

Alice
Examples of MAD distributed systems

- The Internet
- P2P systems: Skype, BitTorrent, ...
- Cloud platforms: EC2, Azure, ...
- MMORPGs: World of Warcraft, ...
- CDNs; Akamai Download Manager
- Interbank networks
- Telephone systems
- Social networks: Facebook, ...
- ...

And this is just the beginning...
The trend towards MAD systems

- Social and technical systems are becoming increasingly connected
  - We will see more, bigger, and more complex MAD systems over the coming years
- My opinion: This will be one of the next 'grand challenges' for systems research
Characteristics of MAD systems

- **Technical**
  - Distributed state
  - True concurrency
  - Complex failure modes
  - Asynchrony
  - Large number of nodes
  - Resource limitations

- **Non-technical**
  - Distributed control
  - Incomplete information
  - Conflicting interests
  - Selfish/’crazy’ behavior
  - Privacy + confidentiality
  - Social/economic aspects
Research directions in this area

- **Study** existing MAD distributed systems
  - How do they form and evolve?
  - What is their structure?
  - How do the participants act, and why?

- **Design** new techniques to build better MAD distributed systems
  - How do we make them robust and reliable?
  - How do we encourage cooperation?

- **Experiment** with these techniques
  - Do they perform as expected in a real deployment?
  - If not, why not?
Outline

- Introduction
- A tour of CIS700/003
  - Measurement and analysis
  - Building MAD systems
  - Faults and misbehavior
  - Internet crime
  - Privacy and confidentiality
  - Novel opportunities
  - Experience
- Course logistics
What should we measure?

- What is the topology of the Internet?
- How does the Internet evolve?
- How does information flow in social networks?
- Where does spam come from?
- How do worms and viruses spread?
- What contracts do ISP have with each other?
- Which websites are sharing information?
- …
Why should we care?

- What is the topology of the Internet?
  - Better route selection
  - Resilience estimation
  - Informed application design

- How does information flow in a social network like Facebook or Twitter?
  - Targeted advertisement
  - Smart caching
  - Recommendations

- Where does spam come from?
  - Improved spam blocking
  - Identify spammers & hold them accountable
Measuring MAD systems is hard

- Data may be *distributed* across many domains
  - Challenges: Confidentiality, heterogeneity

- Data is often *not readily accessible*
  - May need to make creative use of existing, unrelated mechanisms to extract it (Examples: RR option, web frontend)
  - Countermeasures may be in place

- There may be a **huge amount** of data
  - Can use sampling, but must be done properly
  - Need to choose the proper metrics to extract any meaningful information

- Need a code of **ethics**
  - Measurement must not place undue burden on the system
  - Need to be careful with sensitive data (proper anonymization)
Paper #1: OSN measurement study

- Studies the structure of four online social networks: Flickr, YouTube, LiveJournal, Orkut
- Used crawling to collect data on over 11.3 million users, 328 million links
- Main findings:
  - Substantial differences to other networks, e.g. web graph
  - High fraction of symmetric links (reciprocity)
  - High degree of local clustering (cliques)

Mislove et al., "Measurement and Analysis of Online Social Networks", IMC 2007
Paper #2: DisCarte

- Builds a map of the Internet's router-level topology
  - Useful for diagnostics, modeling, route selection, overlay networks...

- Challenge: Techniques for collecting topology data (traceroute, IP RR) are error-prone
  - Router aliases, incorrect links, instabilities...

- Idea: Cross-validate maps that were collected using different techniques
  - Result: Accuracy of the map is much improved

Sherwood et al., "DisCarte: A Disjunctive Internet Cartographer", SIGCOMM 2008
Paper #3: Spamalytics

- **Goal:** Understand economic support for spam
  - Any structural weaknesses may help with fighting spam

- **Contribution:** Measurement and analysis of a spam campaign
  - Authors infiltrated an existing botnet's infrastructure (!)
  - Analyzed two spam campaigns, one that propagated a Trojan and one that advertised pharmaceuticals
  - Studied the 'conversion rate': How many spam mails actually reach their goal (purchase, infection of computer)
  - Finds that the conversion rate is tiny (<0.00001%)
Recap

- It is useful to study existing MAD systems
  - Examples: Find better routes, fight spam, less annoying ads
- Studying MAD systems is difficult
  - Scale, distribution, heterogeneity, confidentiality, ethics
- We will see examples of measurement studies that look at
  - the social aspect (OSN study)
  - the technical aspect (DisCarte)
  - the behavioral aspect (Spamalytics)
Outline

- Introduction
- A tour of CIS700/003
  - Measurement and analysis
  - Building MAD systems
  - Faults and misbehavior
  - Internet crime
  - Privacy and confidentiality
  - Novel opportunities
  - Experience
- Course logistics
Why building MAD systems is hard

Building any distributed system is hard
- Distributed state, synchronization, consistency, ...

But MAD systems present additional challenges:
- Large scale, distributed control, incomplete information, conflicting interests, ...

Insecure routing redirects YouTube to Pakistan
- Ars Technica, February 2008

Air Traffic is Delayed Nationwide - Facility Glitches In Georgia Ripple Through System
- The Wall Street Journal, August 2008

Cisco routers caused major outage in Japan
- Network World, May 2007

Google App Engine Goes Down and Stays Down
- TechCrunch, June 2008

Netflix: Faulty hardware caused massive outage
- USA Today, August 2008

LAX outage is blamed on a single computer
- Los Angeles Times, August 2007

Web startups crumble under Amazon S3 outage
- The Register, February 2008
Challenge #1: Large scale

- Systems can have millions of nodes
  - Asymptotic complexity becomes really important
  - Centralization is usually not an option
  - Need scalable algorithms + ways to make system fully decentralized
- Paper #1: SplitStream (SOSP 2003)
Challeng #2: Distributed control

- No single administrator has control over all the nodes (configure, repair, install, etc...)
  - Problem: Administrators must cooperate to make the system work well
  - Example: Admin A must convince admin B that the problem is on B's side, not on A's
  - No special paper; will be covered later (PeerReview)
Challenge #3: Incomplete information

- Nobody knows the state of the entire system
  - Domains may be unwilling to share information (confidential, business secret) or may neglect it unintentionally
  - Problem: How do we detect and diagnose problems?
  - Paper #2: Hubble (NSDI 2008)

Why can't I get to YouTube?
Challenge #4: Conflicting interests

- Sometimes a domain can do something that is good for itself, but bad for another domain
  - Example: Early-exit routing
  - Problem: Price of anarchy! How to coordinate?
  - Paper #3: Nexit (NSDI 2005)
Recap

- Building any distributed system is hard, but MAD systems are especially challenging
  - Scale, distributed control, incomplete information, conflicting interests, ...

- Special techniques are needed
  - Decentralized structure
  - Algorithms with good asymptotic complexity
  - Fault detection across domains
  - Negotiation to improve efficiency
  - ...

© 2010 Andreas Haeberlen
Outline

- Introduction
- A tour of CIS700/003
  - Measurement and analysis
  - Building MAD systems
  - Faults and misbehavior
  - Internet crime
  - Privacy and confidentiality
  - Novel opportunities
  - Experience
- Course logistics
Problem: Faults and misbehavior

- System designer assigns each node a well-defined function, its expected behavior.
- But in practice, not all nodes will follow the expected behavior:
  - **Faults**: Technical problem (node on fire, software bug, ...)
  - **Misbehavior**: Social problem (user has manipulated the node and installed different software, or is attacking the system).
Challenges specific to MAD systems

- Misbehavior does not come from outside attackers, but from within the system
  - Access restrictions, authentication etc. do not help

- Faults and misbehavior are the common case, not the exception
  - In a 1,000-node system, there will always be a few faulty nodes and misbehaving domains
  - Waiting for a fault-free system is not an option

- New forms of misbehavior to consider
  - Example: Selfish/rational behavior
Why do we need models?

- **Goal:** Build systems that have useful properties even in the presence of faults/misbehavior
  - No guarantees possible if all nodes can fail at the same time
  - But that is not realistic → Need to **model** the faults we expect
  - Model is always an assumption; can be wrong in practice!

- **Common types of models:**
  - **Type:** Only X, Y and Z can happen on affected nodes (e.g., data loss, but not data corruption)
  - **Pattern:** At most f nodes can be affected at the same time (or: nodes A and B can never be affected at the same time)
  - **Power:** Affected node cannot do X (e.g., factor large numbers)
  - Simple example: Crash fault model
Common models

- **Rational model:** Misbehaving domains have a specific, known goal (such as making money)
  - **Pro:** Good approximation for many systems
  - **Pro:** Powerful; gives us a lot of leverage
  - **Con:** Not everyone is perfectly rational, and it can be difficult to anticipate the domains' real goals

- **Byzantine model:** Misbehaving domains can behave arbitrarily (within their power)
  - **Pro:** Very pessimistic. Assumption almost always holds.
  - **Con:** Very pessimistic. Requires very expensive techniques.
Handling rational nodes

- **Approach #1: Incentives**
  - Rational nodes will do work if they get something in return
  - Example: Tit for tat
  - Paper #1: BitTorrent is an auction

- **Approach #2: Reputations**
  - Domains can rate their interactions with other domains as satisfactory or unsatisfactory
  - Similar to Ebay recommendations
  - If a domain acquires a bad reputation, others stop working with it
  - Paper #2: Credence (Best paper at NSDI 06)
Handling Byzantine nodes

- **Approach #1: BFT**
  - **Goal:** Mask faults, so that they do not have any effect
  - **Idea:** Replicate the function of each node onto k nodes
  - **Challenge:** Faulty nodes can lie, equivocate, fall silent...
  - **Paper #3:** PBFT (OSDI 1999)
Handling Byzantine nodes

- Approach #2: Trusted hardware
  - Goal: Detect faults
  - Assume that there is a small, simple device in each node that can never fail (Example: TPM)
  - Use this device to detect when the node as a whole fails
  - Paper #4: TrInc (best paper at NSDI 2009)
Handling Byzantine nodes

- **Approach #3: Accountability**
  - **Goal:** Detect faults and create evidence
  - **Idea:** Each node has a special tamper-evident log in which it records everything it does.
  - Nodes can inspect each other's log and check for faults
  - If a fault is found, nodes can obtain evidence
  - **Paper:** PeerReview (SOSP 2007)
Recap

- System designer assigns each node a function, but not all nodes follow it
  - Faults: Technical problem with the node
  - Misbehavior: Node is manipulated by the domain that runs it

- Need to model what kinds of faults can occur
  - Rational: Domains are selfish, follow a known goal
  - Byzantine: Faulty nodes can behave arbitrarily

- For each model, several techniques are available for handling faults
  - We will study: Incentives, reputations, BFT/replication, trusted hardware, and accountability
Outline

- Introduction
- A tour of CIS700/003
  - Measurement and analysis
  - Building MAD systems
  - Faults and misbehavior
  - Internet crime
  - Privacy and confidentiality
  - Novel opportunities
  - Experience
- Course logistics
Goals for this section

- The techniques in the previous section are effective against entire classes of misbehavior
  - Good: Requires fewer assumptions, hard to circumvent
  - Not so good: Techniques tend to be expensive and complex

- In practice, we tend to see specific kinds of misbehavior, for example:
  - Spam
  - Denial of service
  - Extortion
  - Spoofing
  - Phishing
  - Espionage
  - Fraud
  - Cheating
  - Sybil

- Goal #1: See some examples of misbehavior
- Goal #2: Discuss some specialized defenses
Problem #1: Spoofing

- In the Internet today, the source IP address of a packet can be set arbitrarily
  - Some domains check it (egress filtering), but not all do
- On the other hand, the IP address is sometimes used like an identifier
  - Example: Complaints about offensive traffic (portscans, …)
  - Example: Tracking copyright violations
- Problem: Innocent users can find themselves accused of misbehavior
- Paper: DMCA take-down (HotSec'08)
Problem #2: Sybil attacks

- Many distributed systems use weak identities
  - Example: Free user accounts on a web page (IMDb)
  - A single user can create many accounts and control them all

- Problem: Malicious user can gain considerable power over the system
  - Analogy: Create 10,000 IMDb accounts and have them all give 'Avatar 3D' a rating of 10.
  - Example: Eclipsing a node in a DHT

- Several possible countermeasures, e.g.:
  - Centralized authority links online identity to offline identity
  - Distributed techniques for Sybil detection

- Paper: SybilGuard (SIGCOMM'06)
Problem #3: Denial of service

- Attacker tries to hurt his victim by consuming resources (CPU, bandwidth, storage, ...)
- Problem: Difficult to distinguish attackers from legitimate clients
- What can we do? Examples:
  - Over-provision massively
  - Detect and block attacker's requests
  - Charge for service
- Paper: Defense by offense (SIGCOMM'06)
Paper #4: Motivation of Internet crime

- Many types of online criminal activity:
  - Credit card fraud, identity theft, spamming, phishing, ...

- What motivates the attackers?
  - Profit-driven: Attacker can make money
  - There is an active and diverse on-line market economy that trades in illegal digital goods and services
  - If we understand how this market works, we may be able to discover vulnerabilities or devise countermeasures

- Paper: Wealth of Miscreants (CCS'07)
  - First systematic study of the activity on this underground market, based on >13 million IRC messages
Recap

- Many deployed MAD systems need defenses against specific types of misbehavior
  - Spoofing, Sybil attacks, DoS, phishing, fraud...
- Specialized defenses can be more efficient and/or rely on weaker assumptions
- We will
  - look at some specific examples of misbehavior that occur in 'real' MAD systems
  - see some specialized defenses
Outline

- Introduction
- A tour of CIS700/003
  - Measurement and analysis
  - Building MAD systems
  - Faults and misbehavior
  - Internet crime
  - Privacy and confidentiality
  - Novel opportunities
  - Experience
- Course logistics
Why should we care?

- Some examples of privacy challenges:
  - System may store highly sensitive information (religious and political views, sexual preferences, employment history...)
  - Controlling access is difficult
  - Data may leak out accidently or through attacks
  - Once sensitive data is out, it potentially stays around forever
Paper #1: Persona

- OSNs store sensitive information
  - Examples: Sexual preferences, political/religious views

- But: Difficult to restrict access
  - Example: Name, current city, networks, fan pages are now 'public information' on Facebook
  - Controls may be insufficient to implement desired policy

- Persona: New OSN that protects user privacy
  - Attribute-based encryption
  - Decentralized, persistent storage

In 2006, Netflix released a large, anonymized sample of their ratings database

- Purpose: Aid contestants of 'Netflix Prize'
- 100,480,507 movie ratings from 480,189 Netflix subscribers

Claim: Privacy was preserved

- Q: "Is there any customer information in the dataset that should be kept private?" A: "No, all customer identifying information has been removed; all that remains are ratings and dates. This follows our privacy policy. [...]"

Paper shows that, with some outside information (IMDb), it is possible to identify users in the dataset

- With only 8 movie ratings, 99% of records can be uniquely identified. For 68%, two ratings and dates are already sufficient.

Narayanan and Shmatikov, "Robust De-anonymization of Large Sparse Datasets", S&P 2008
How can we ask queries over a data set without compromising privacy?

- Example: Database of all patients in the U.S. with HIV
- How many of them live in ZIP code 19104?
- Problem: 87% of the U.S. population can be uniquely identified by gender, ZIP code, and full DoB

Idea: Give users a 'privacy budget'

- 'Privacy cost' of each query is debited from the budget
- Once budget is exhausted, users can no longer ask queries
- Hard guarantees based on differential privacy

McSherry, "Privacy Integrated Queries", SIGMOD 2009
Paper #4: Vanish

Problem: Personal data stays around forever

- Examples: Old emails, Facebook messages, Flickr photos...
- Data is cached, copied, archived
- Privacy is at risk to accidental, malicious, and legal attacks

Idea: Can we ensure that all copies of the data will 'self-destruct' after a certain time?

- No explicit deletion required
- Data is unreadable even if an attacker can retroactively obtain a pristine copy
- No secure hardware required

Privacy and confidentiality are important challenges in MAD systems
- Anonymization may not offer sufficient protection
- Restricting access to sensitive data can be difficult
- Revocation problem
- Difficult to limit information leakage when data is queried

Considerable advances in the past few years
- Often based on novel crypto techniques
- Interesting systems aspects
Outline

■ Introduction

■ A tour of CIS700/003
  ■ Measurement and analysis
  ■ Building MAD systems
  ■ Faults and misbehavior
  ■ Internet crime
  ■ Privacy and confidentiality
  ■ Novel opportunities
  ■ Experience

■ Course logistics
MAD systems present new challenges, but also new opportunities for building better systems.

Examples:
- Relationships between domains can be used to estimate node reliability
- Social networks can be used to establish trust and to raise the bar for attacks
- Variety of domains tends to increase heterogeneity, which can help with reliability
Paper #1: F2F

- **Goal:** Build a P2P storage system

- **Problem:** High churn
  - If data is placed randomly, nodes end up storing data for people they don't know
  - Little incentive to remain in the system → Churn

- **Idea:** Leverage friendship relations
  - People tend to behave cooperatively towards their friends
  - Store data on machines that belong to friends

---

Li and Dabek, "F2F: Reliable Storage in Open Networks", IPTPS 2006
Goal: Reduce unwanted communication

Idea: Leverage trust relationships, e.g., from a social network

Mislove et al., "Ostra: Leveraging trust to thwart unwanted communication", NSDI 2008
Paper #3: PlanetLab

- PlanetLab: A global platform for deploying and evaluating network services
  - 1070 nodes, 494 sites, dozens of countries

- Interesting case study of a MAD system
  - Has evolved considerably since its first deployment in 2002
  - Challenges: Maintaining trust relationships, resource allocation, management, stability, autonomy, ...
  - Paper reports experiences of the PlanetLab designers

Peterson et al., "Experiences Building PlanetLab", OSDI 2006
Recap

- MAD systems not only present new challenges, but also offer new opportunities
  - Example #1: Using social relationships to fight spam
  - Example #2: Using trust to increase reliability of storage

- Building and running a MAD system is a complex task
  - Case study: PlanetLab
Outline

- Introduction
- A tour of CIS700/003
  - Measurement and analysis
  - Building MAD systems
  - Faults and misbehavior
  - Internet crime
  - Privacy and confidentiality
  - Novel opportunities
  - Experience
- Course logistics
How the course works

- This course has two main components:
  - Paper reading, paper presentations, in-class discussions
  - Term project

- In addition,
  - I will conclude each class by saying a few words to put the papers in context
  - I will occasionally give short lectures about fundamentals that will help you understand the papers better
  - Example: Next Tuesday I will give you an introduction to interdomain routing
Part #1: Reading + discussion

- We will read one conference/workshop paper per class
  - Short summaries (1-2 paragraphs) due one hour before class
    - What is the key problem the paper addresses?
    - Why is this an important problem?
    - How does the paper solve the problem?
    - What are strengths and weaknesses of the paper? (At least one each)
  - One student briefly presents each paper (15-20 minutes) and leads the in-class discussion
  - At the end of each class, I will say a few words to put the paper in context
Part #2: Term project

- Two options:
  - Research project (strongly preferred)
  - Survey paper

- Research projects can be done in small (!) teams or individually

- Timeline (tentative):
  - Feb 4: Project proposals due (one page)
  - Mar 18: Status reports due (two pages)
  - Apr 22: In-class presentations (15 minutes incl. Q&A)
  - Apr 27: Final reports (six pages, workshop paper format)
Part #2: Term project

How do I pick a research project?

- **Option #1:** Work on an aspect of your current research that is related to this course
  - No 'extra' work for you!
  - Results can become part of your thesis

- **Option #2:** Pick an entirely new project in this area that you find exciting
  - Opportunity to try something new
  - To maximize chances of success, please come see me and get some feedback before submitting your proposal

- **Option #3:** Come talk to me during office hours and get some suggestions for projects
  - Several options, depending on your background and interests
Todo for next week

- Please send me a short introductory email to ahae (at) cis (dot) upenn (another dot) edu
  - Program (MS/PhD) and year
  - Any topics you want to hear more (or less) about

- Please pick at least one paper you'd like to present before spring break, and send me an email to reserve it
  - Papers are assigned on a first come, first served basis

- Please read the Gao/Rexford paper
  - You are encouraged but not (yet) required to submit a summary
Preview: Next week

- Short lecture on fundamentals:
  - Internet routing, BGP, ...

- And a paper that shows:
  - that there is no technical reason why Internet routing should ever stabilize,
  - that in the real world, it almost always stabilizes, and
  - that the reason is a subtle property of the economic relationships between ISPs
See you on Tuesday!

MAD distributed systems need YOU!