HELSINGIN YLIOPISTO
HELSINGFORS UNIVERSITET
UNIVERSITY OF HELSINKI

Peer-to-Peer and Grid Computing

Chapter 7: Other Issues

Chapter Outline

- Further issues in P2P systems
- Security (in DHTs)
  - Overview of problems
  - Sybil attack
- Privacy and anonymity
  - Can these be protected?
- Napster legal case
  - Why original Napster failed and what can we learn?
- Online music stores
  - Alternative to file sharing?

Security in DHTs

- DHT architectures assumes a trusted system
  - True in corporate environments, but not on the Internet
- One solution: Central certificate-granting authority
  - Used by Pastry and its related projects
  - Constrains membership in DHT
- One attack: Return incorrect data
  - Easy to avoid through cryptographic techniques
  - Detect and ignore non-authentic data
- Focus: Attacks that prevent participants from finding the data
  - Threatens the liveliness of the system

DHT Components

DHTs have following components:

1. Key identifier space
2. Node identifier space
3. Rules for associating keys to nodes
4. Per-node routing tables that refer to other nodes
5. Rules for updating routing tables as nodes join and leave

- Any of the above may be the target of the attack

Adversary Model

- Adversaries are participants in DHT that do not follow protocol correctly

Assumptions:

- Malicious node can generate arbitrary packets
  - Includes forged source IP address
- Can receive only packets addressed to itself
  - Not able to overhear communications between other nodes
- Malicious nodes can conspire together, but still limited as above

Types of Attacks

1. Routing attacks
2. Attack against data storage
3. Miscellaneous attacks

- First goal: Detect attack
  - Violation of invariants or contracts
- What to do when an attack is detected?
  - Is other node malicious?
  - Did other node simply not detect attack?

- Achieving verifiability is vital

Routing Attacks

- Routing is responsible for maintaining routing tables and sending messages to correct nodes
- Routing must function correctly
  - Define invariants and check them
- Attacker can forward messages incorrectly
  - But: Each hop should get “closer” to destination
  - Querying node should check this
  - Allow querying node to observe lookup process
    - For example, processing messages recursively hides this
- Attacker can claim wrong node is responsible node
  - Querying node is “far away”, cannot verify this
  - Assign keys to nodes in a verifiable way
  - Often: Assign node IDs in a verifiable way (e.g., IP address)
    - For example, CAN lets node pick its own ID…

More Routing Attacks

- Attacker sends incorrect routing updates
  - Blatantly wrong updates can be detected
  - If DHT allows several choices for next hop
    - Attacker can pick a “bad” node
    - Not necessarily a problem with correctness, only performance
    - Can be a problem for some applications (anonymity)
  - Server selection can be abused

More Routing Attacks 2

- Attacker can partition network
  - If new node contacts attacker first, attacker can partition network (can even hijack nodes from real network)
  - Parallel network is consistent and “looks OK”
    - Attacker can track nodes
  - Bootstrap from a trusted source: Hard to get in dynamic networks, public keys might help
  - Cross check routing tables with random queries
    - Assumes we were part of network earlier, still not totally safe

Storage and Retrieval Attacks

- Attacker can deny existence of data
  - Or return wrong data
- Must implement replication at storage layer
  - Who creates replicas?
  - Clients must be able to verify that all copies were created
- Avoid single points of responsibility
  - Replication with multiple hash functions is one good way
- Big problem if system does not verify IDs
  - Any node can become responsible for any data
  - For example, Chord allows virtual nodes

Miscellaneous Attacks

- Attacker can behave inconsistently
  - Some nodes see it as good, others as bad
  - Maintain good face to nearby nodes
  - How would a distant node convince neighbors of bad node?
    - Public keys and signatures could solve this
- Denial of service
  - Attacker floods a node with messages
  - Node appears failed to the rest of the network
  - Replication helps, but attacker may succeed if replication not sufficient
  - Replicas should be in physically different locations
    - DHT assigns keys to nodes randomly, should be OK
    - Large attacks require lot of resources

More Miscellaneous Attacks

- Attacker can join and leave the network rapidly
  - Causes lot of stabilization traffic in network
  - Loss of performance, maybe loss of correctness
  - Works well if stabilization requires lot of data transfer
    - For example, copying of large objects from node to node
  - DHT must handle this case anyway
- Attacker can send unsolicited messages
  - *Q* asks *E* and gets referred to *A*
  - *E* knows *Q* expects an answer from *A*
  - *E* forges message from *A* to *Q*
  - Public keys and signatures (heavy solution)
  - Random nonce in a message works also

Design Principles

Summary of design principles for secure DHT:

1. Define verifiable system invariants (and verify them!)
2. Allow querying node to observe lookup process
3. Assign keys to nodes in a verifiable way
4. Server selection in routing may be abused
5. Cross-check routing tables with random queries
6. Avoid single points of responsibility

Sybil Attack

- Sybil?
  - From book/movie telling the story of Sybil Isabel Dorsett who suffered from multiple personality disorder
- How to protect against malicious peers?
- For example, data replication
  - A single copy might be on a malicious peer
  - But several copies on different peers are safe, right?
- How can we know that the “different” peers are really different and distinct physical entities?
- Answer: We need a centralized, trusted entity (e.g., CA)
- Without central authority, the problem is *unsolvable*
  - Can be proven mathematically to be unsolvable

What Is The Problem?

- Entity: Real-world entity, e.g., one user
- Identity: Representation of an entity in system
- Redundancy requires resources to be spread across several entities
  - Peer-to-peer systems work only with identities
- How to ensure one entity does not create multiple identities and attack the system that way?
- This is called the Sybil Attack
- Only solution is a (logically) centralized authority for managing entity-identity mappings

Examples of Solutions

- Actually centralized authorities:
  - Certification Authorities, e.g., VeriSign
- Logically centralized authorities:
  - Hashing IP address to get DHT identifier (e.g., CFS)
  - Add host identifiers to DNS names (SFS)
  - Cryptographic keys in hardware (EMBASSY)
  - These appear distributed, but they all rely on some centralized authority (e.g., ICANN gives out IP addresses and DNS names)
- Identities vouching for other identities
  - For example, PGP web of trust for humans
  - NOT a solution!
  - Attacker can attack the system early and compromise generation of identities and break chain of vouchers

Results

- Entity should accept identities only if they have been validated by central authority, itself, or others
  - In a fully distributed system, only entity itself and others
- Following can be shown under reasonably realistic assumptions for direct validation:
  1. Even when severely resource constrained, a faulty entity can counterfeit a constant number of multiple identities
  2. Each correct entity must simultaneously validate all the identities it is presented; otherwise, a faulty entity can counterfeit an unbounded number of entities
  - Similar results hold for indirect validation by others
- What resources can be used in identification?
  - Communication, CPU, storage

Resources as Proof

- Communication
  - Broadcast request for others to identify themselves and accept only responses which come within a certain time interval
  - Model had assumed broadcast communications
- CPU
  - Require other peer to perform some computationally intensive, but easily verifiable, task
  - This requires simultaneous identification (point 2 from above)
- Storage
  - Have others store some uncompressible data and periodically ask them to give back a small piece
  - Would eventually catch a Sybil attack
  - Problem: No storage space left for doing any real work…

Implications of Sybil Attack

- Need centralized authority for managing identities
- Logically centralized systems should be aware of their potential (future) vulnerabilities
  - For example, privacy extensions for IPv6 might break CFS
- Sybil attack can be avoided under the assumptions:
  - All entities operate under identical resource constraints
  - All presented identities are validated simultaneously by all entities, coordinated over the whole system
  - For indirect validation, the number of vouchers must exceed the number of failures in system
- Are these assumptions feasible or practical for a large-scale distributed system?
  - Answer would seem to be no

Privacy

- Privacy is freedom from unauthorized intrusion (M-W)
- In physical world, privacy is easy to define and maintain
  - “Close the door”, “Send letter in envelope”, …
- What about the digital world?
  - What kind of privacy is “reasonable” to expect?
  - What kind of privacy corresponds to the “classical” privacy?
- Encryption can be used to protect personal data
- What about personal information stored by others?
  - Store needs to keep customer registry to function
  - How should that information be kept and protected?

Anonymity

- Anonymity seen as a way to protect privacy
- Pseudonyms (e.g., user-picked ID) provides a simple form of protection
- But pseudonyms are not enough
  - Record company knows IP address
  - IP address reveals ISP
  - ISP has logs to tell who used the IP address
  - Lawsuit follows
- Pseudonyms also allow for user tracking
- How to provide true anonymity on a P2P network?
- Several solutions: FreeNet, Achord, Tarzan, Herbivore

Achord: Basics

- Achord is a censorship resistant Chord
  - Note: Censorship resistance not quite same as anonymity
- Analysis about which Chord functionality is vulnerable to revealing the identities of nodes
- Chord (or any DHT) is suitable for storage networks
  - Guarantees that data will be found
  - Bounds on the number of messages needed
- Other anonymous networks (e.g., FreeNet) have no guarantees
  - In FreeNet, less popular data may disappear
  - No guarantees about finding any content
  - No guarantees about number of messages
  - But FreeNet provides more anonymity than Achord

Key Properties of Censorship Resistance

1. Possible to insert data without revealing the identity of the inserter
   - Cannot censor by attacking those who insert information
2. Possible to retrieve data without revealing the identity of the retriever
   - Cannot censor by attacking those who want information
3. Difficult to introduce a new node such that it will be responsible for a given document
   - Cannot censor by deleting documents
4. Difficult to identify node which is responsible for a given document
   - Cannot censor by attacking the responsible node

- (Especially) last point not fulfilled by Chord
  - Chord returns address of responsible node
  - Problem with implementation, not a fundamental weakness

Achord and Chord

- Node identity is SHA-1 hash of IP address
  - Virtual nodes numbered and hashed
  - Fulfills property 3
- Each node knows *O(log N)* other nodes (finger table)
  - Achord attempts to limit knowledge to this
  - Attempts to fulfill property 4
- Finding successor is Chord's fundamental operation
  - Iterative and recursive methods
  - *Find_successor* lets node find out what keys other node is responsible for
  - Achord never returns *find_successor* to requesting node
  - Achord maps keys to values
    - Chord maps keys to nodes

Achord: Finding Successor

- No *find_successor* returned in Achord
  - *Find_successor* is used, but the actual successor is not revealed to the requesting node
- Instead, *connect_to_successor*
  - Value is tunneled back to the requesting node
  - Same for inserting a value
- Provides anonymity
  - Tunnel node cannot know who is requesting
    - Could be immediate requester or someone else
  - Identity of the node storing a key is not shown
- Above takes care of retrieving and inserting keys
- Overlay maintenance requires new procedures

Overlay Maintenance

- Recall: To join, new node must find its successor
  - Call *find_successor* with own ID
- Achord restricts use of successor and predecessor
  - Only needed in a few cases, easily identified
- Node *n* calls *find_successor(n)* to join network
  - Benign call, anyone can verify that this is OK (needs IP address)
  - In fact, a node must know its successor
- Rule 1: Only node with ID *n* is allowed to call *find_successor(n)*
  - Implies recursive processing of join is not possible
- Rule 2: Only iterative processing of *find_successor* possible
  - *O(log N)* nodes learn about a new node

Predecessors

- Node needs to access predecessor field on other nodes in a single case
  - Periodic stabilization and ring maintenance
- Possible to determine if access to predecessor field is valid
- If *n'* is successor of node *n*, then:
  - *n* has called *find_successor(n)* which ended up at *n'*
  - *n'* sets predecessor to *n*
  - *n'* keeps list of predecessors, only most recent can access it
- Rule 3: A node can access predecessor field on another node only if it was previously the predecessor and has not accessed the field since the value changed

Finger Tables

- Achord replaces Chord's finger table maintenance
  - Chord calls *find_successor* for each finger table entry
- Node updates its finger tables by picking a random node *n'* from its current finger table
  - Call *n'.find_best_match(i)*, where *i* is index to *n*''s finger table
  - *n'* knows IP of *n,* can calculate the best match for *n*'s finger table slot *i*th position
- Rule 4: Finger tables updated with *find_best_match* which returns a new IP address only if that node is a better match than the current node
- Nodes can collect IP addresses of others
  - Can get *O(k log N)* addresses

Achord: Issues

- Possible to attack Achord if you have access to a large number of IP addresses
  - Higher probability to be responsible for a given document
  - Must limit number of virtual nodes?
- Achord maybe not as anonymous as FreeNet
  - Key and node IDs can be used to guess if a node sent a message
- Nodes can learn about others during stabilization
  - Extent is still unclear

Achord: Summary

- Achord adds censorship resistance to Chord
- 4 basic properties of censorship resistant systems
- Basic idea:
  - Provide anonymity
  - Limit a node's knowledge about other nodes
- Hard to provide total anonymity and good performance
  - Tradeoff between the two
  - Need more investigation
- What is required from an anonymous system?
- What is acceptable performance?

P2P and Copyright

- What did Napster do wrong?
  - First lawsuits against Napster after only a few months
  - Eventually, Napster had to shut down
- Reason for lawsuits: Copyright violations
  - Users on Napster were sharing files without permission
  - Copyright holders (= record companies) have the right to protect their rights
- What can we learn from this case?
  - Especially from the point of view of P2P software developer
  - How should you build your system?
  - What kinds of mechanisms can you use to avoid liability?
- Recent rulings have gone against file sharing
  - Most networks being shut down

What is Copyright?

- Copyright is:

  “A form of intellectual property that grants its holder the legal right to restrict the copying and use of an original, creative expression for a defined period of time.”
- Copyright holder has exclusive rights to:
  - Make and sell copies of the work (including electronic copies)
  - Import or export the work
  - Make derivative works
  - Publicly perform the work
  - *Sell or assign the rights to others (e.g., artist to record company)*
- Only the copyright holder can do these things
  - Everyone else is prohibited from doing them

Copyright and File Sharing

- Copyright applies also to file sharing

1. Digital file is fixed
   - Files being shared qualify as copyrighted works
2. Transmission of a file is reproduction
   - Only copyright holder can reproduce the work

- Any unauthorized reproduction of a copyrighted work is possibly copyright infringement
- Our discussion concerns the Napster case and American copyright law
  - European law similar, but varies from country to country
  - New EU directives about copyright enforcement

Direct Infringement

- Direct infringer is someone who is directly violating copyright law
  - User who shares an unauthorized file
- Direct infringer can be sued
  - Record companies have sued many individual users who were sharing large number of files
- In modern P2P file sharing networks, the presence of direct infringers is “guaranteed”
- File sharing network would need to implement special mechanisms to prevent unauthorized sharing
- Direct infringement does not (directly) concern the P2P software developer

What About The Developer?

- Software developer not (usually) involved in creation or transmission of unauthorized copies
  - Easy to avoid this in a P2P system
- Copyright law can hold you accountable for the actions of others
  - Also applies to other areas of law

Two kinds of secondary liability:

1. Contributory
2. Vicarious

Contributory Infringement

- *"One who, with knowledge of infringing activity, contributes to the infringing may be held liable."*

Copyright owner must prove:

1. Direct infringement
   - Direct infringement must have happened by someone
2. Knowledge
   - Accused knew of infringement
   - Actually, "should have known" is enough
   - Must have specific knowledge, "system is capable of infringement" is not enough
3. Material contribution
   - Accused must have contributed
   - Providing "site and facilities" (e.g., search) is enough

Vicarious Infringement

- Employer is responsible for actions of employees
  - Right and ability to supervise and financial benefit

Copyright holder must prove:

1. Direct infringement
2. Right and ability to control
   - Must show that accused has right and ability to control the direct infringement
   - *Napster: Ability to block user accounts is control*
3. Direct financial benefit
   - Accused must get direct financial benefit from infringement
   - Actually: "direct" and "financial" not important, any benefit is enough
   - *Napster: Infringing material brings more users, makes company more attractive to investors*

Vicarious Infringement: Note

- Vicarious infringement has no requirement of knowledge
- Possible to be completely unaware of infringing activity and still be liable
- Strong incentive to monitor your users
  - If you do not monitor, you take a big risk

Possible Defenses

- No direct infringement
  - No direct infringement, no indirect liability
  - Hard to prove in a P2P file sharing network
- Betamax defense: "Capable of substantial non-infringing uses"
  - Originally from Sony Betamax VCR case
    - Device capable of "substantial non-infringing uses"
    - No indirect liability
    - Actual use does not matter, "capability" is enough
  - *Napster: Betamax does not apply to vicarious infringement*
  - *Napster: Betamax defense applies only until you are notified of infringement*

More On Betamax Defense

- Recent interpretations have two implications

1. Betamax does not apply to vicarious liability
   - Control and benefit are dangerous
   - “Service” or “community-building” models are dangerous
     - These usually include some form of control
2. When you are notified, you must do “something”
   - What is “something”?
   - *Napster: “Something” may be limited by the P2P technology*
     - In a fully decentralized network, not possible to do much
   - Copyright owners argue designers should design for this case
     - This point not accepted by courts

- Extent and applicability of Betamax defense still unclear

One More Defense

- DMCA Section 512 “Safe Harbors”
  - Similar new copyright directives in Europe too
- Only apply to “online service providers” if infringement involves any of:
  - Transitory network transmission
  - Certain kinds of caching
  - Storage for others (e.g., web hosting)
  - Information location tools (e.g, search engine)
- Safe harbors very tightly defined
  - Consult a lawyer
- This defense (also) failed for Napster

Lessons and Guidelines

- Make and store no copies
  - Even a copy in RAM can be considered a copy!
  - Creating copies makes you a *direct infringer*
  - Not really a problem for P2P developer (except caching?)
- Total control or total anarchy
  - Contributory infringement: Knowledge and contribution
    - Hard to avoid contribution (software is contribution)
    - When you “know”, you must “do something”
    - “Something” depends on architecture
      - Either full control over users or no possibility to do anything
  - Vicarious infringement: Control and benefit
    - Again, benefit hard to avoid (defined very loosely)
    - What is “control”?
    - Either monitor users or make monitoring impossible

Lessons and Guidelines

- Sell software, not services
  - Vicarious liability maybe biggest threat to P2P developer
  - Service model usually has possibility for “control”
  - Stand-alone software is out of developer’s control
    - For example, VCR manufacturer has no control over users
    - Remember: No automatic updates, etc.
- Can you deny knowledge about user activities?
  - Contributory liability depends on knowledge
  - Can you plausibly deny knowledge?
    - Rememeber: “Should have known” may be enough!
  - Don’t promote infringing uses
    - May mean no customer support
  - Again, total control or total anarchy

Lessons and Guidelines

- What are your "substantial, non-infringing uses"?
  - P2P systems very general purpose, don't think too small
- Don't promote infringing uses
  - No screen shots with Beatles songs in marketing material :-)
- Disaggregate functions
  - P2P system needs several components: search, management, …
  - Split them over several entities (companies)
  - Responsibility of each entity limited to what it controls
  - Some entities may be better protected
    - For example, search entity may fall under DMCA safe harbor
- Don't make money out of infringing activities

Lessons and Guidelines

- Give up end-user license agreement (EULA)
  - EULA is a contract, may imply control
- No “auto-updates”
  - Auto-updates are “control over users”
- No customer support
  - Present no evidence that you have helped a direct infringer
  - Even reading a message from customer may be “knowledge”
    - For example, user asking about problems downloading “Matrix”
- Be open source
  - Hard to show “control” or “financial benefit”
  - But: “Benefit” defined very loosely by courts
  - But: If “dangerous” parts are open source, you can build business on safer ground (additional services)?

Future of File Sharing

- What does future look like for file sharing?
- Record companies going after individual users (i.e., the direct infringers)
  - Even got a conviction (Jammie Thomas)
- BitTorrent communities shut down
  - Sites with links to illegal content
- Illegal file sharing will not go completely away
  - May degrade into an underground activity
- Legal alternatives will become more popular?
  - Buying digital content online

Pollution in File Sharing

- A "pollution company" creates fake files
  - Files appear to be "legitimate" (read: popular songs)
- File contents are not what the metadata says they are
- Searching is only based on metadata
  - Users will get bad files instead of good files
  - Bad files spread through the system
- Two intended outcomes:
  - More bad copies than good copies
  - Users get frustrated and stop using the system
- One such "pollution company" is Overpeer

Types of Pollution

- Content pollution
  - Correct metadata, but content is “modified”
    - For example, insert white noise in the middle of a song
- Metadata pollution
  - Metadata does not match the content (but content might be ok)
- Intentional pollution
  - Pollution is done on purpose
- Unintentional pollution
  - Accidental pollution, e.g., truncate song while ripping, typo in metadata, …

How Much Pollution is There?

- Experiment with several popular songs
- Types of pollution found:
  - Files un-decodable, songs too short or long, modified content
- Result: Pollution is extremely wide-spread
- Up to 70% of copies of some songs were polluted
  - Percentage of polluted copies higher for popular songs
- Simple rating schemes are not enough
  - Even if one bad version is “rated out”, new polluted versions appear too fast

Anti-Pollution Techniques

- Detection with downloading
  - Download all or part of file to determine pollution
  - Match file contents to a well-known trusted source
    - For example, hash contents
  - Users filter out bad copies
    - User downloads file, but does not share bad copies
    - Need incentives?
- Detection without downloading
  - Detect polluted copies without downloading any part of file
  - Download files only from people you trust
  - Web of trust: Same idea, extended
  - Reputation systems

Online Music Stores

- Answer from record companies to file sharing
  - Nothing to do with P2P as such, but a competing technology
- First was Apple's iTunes Music Store (iTunes)
- Many others followed:
  - Napster 2, Walmart, Musicload.de, …
- Idea behind online music stores:
  - Users pay a small amount for a music file (with DRM)
    - File downloaded from store to user's computer
  - Can also buy complete albums
  - Can play songs on computer or portable player, or burn to CD
  - Price typically ~1 euro per song or ~10 euros per album
- Goal: Provide experience similar to buying a real CD

Online Music Stores: User Rights

- What user is allowed to do with music?
  - How does it compare with buying a traditional CD?
- With iTunes, you can do the following:
  - Play song on 5 computers
  - Transfer song to an iPod
  - Burn song to a CD up to 7 times
  - Share song with 5 computers on same subnet (e.g., home)
  - Share song wirelessly to speakers
- Digital Rights Management stops when burning a CD
  - Can later rip to a music file without DRM (loss of quality)
- Are you buying the song or a license?

Online Music Stores: Future

- Currently iTunes and others very popular
- In other words: People are willing to pay for content
  - At least as long as it's a well-marketed and useful service
  - Is this the best business model?
- Trend towards payable media
  - iTunes now sells/rents TV shows and movies
  - DSL operators offer movies
- Still long way from payable Internet
  - Likely to happen in future
  - Basic services will be free, have to pay for others
  - Well-understood by people (e.g., cable or satellite TV)
  - But needs much, much more work to work on Internet?

Chapter Summary

- Security issues in DHTs
- Privacy and anonymity
- Napster legal case and copyright
- Pollution in file sharing
- Online music stores