Cheating in public distributed computing

Michael Feiri

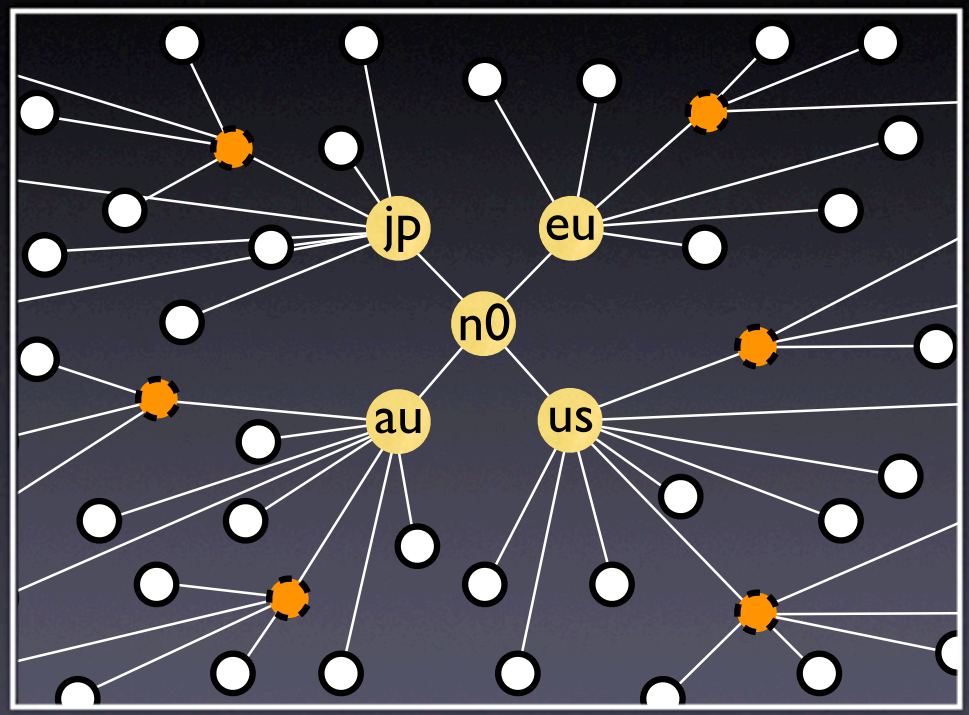
- CCC Erfa Ulm
  - http://www.ulm.ccc.de/chaos-seminar/
- I'm not neutral
  - mfeiri@distributed.net
  - Interested in NFSNET
- Thanks to Décio Luiz Gazzoni Filho, P. Golle & I. Mironov, many others...

About this lecture

- A discussion of the problem and look at some possible solutions to cope with dishonest participants in public distributed computing
- Not intended to provide instructions for script kiddies
- Not a comprehensive discussion of all security aspects

Overview

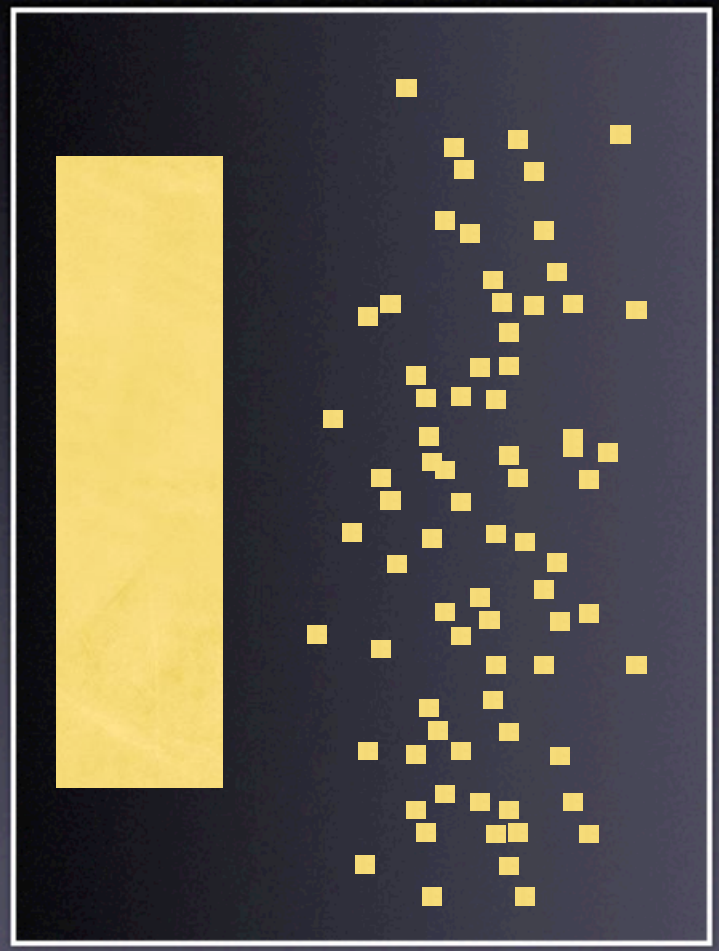
- What is public distributed computing (2)
- Cheaters (4)
  - Illegal use of resources (6)
  - False results (13)
  - More powerful methods (11)
  - Other problems (2)
- The real world (5)



distributed computing

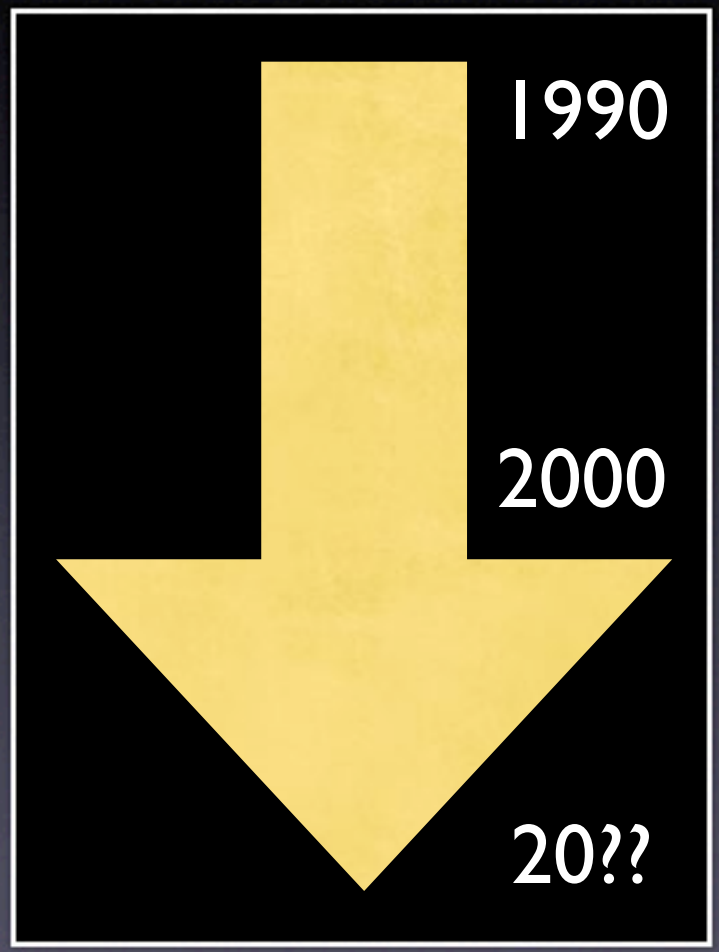
What is distributed computing

- A single computational task spread across multiple physically seperated machines
- In **public** distributed computing these machines are essentially untrusted



Short technical history

- Distribute jobs manually
- Automated distribution
- Cheating becomes a problem
- Commercial DC planned
- Addition of PKI to DC
- More possibilities?



Cheaters!

Motivation

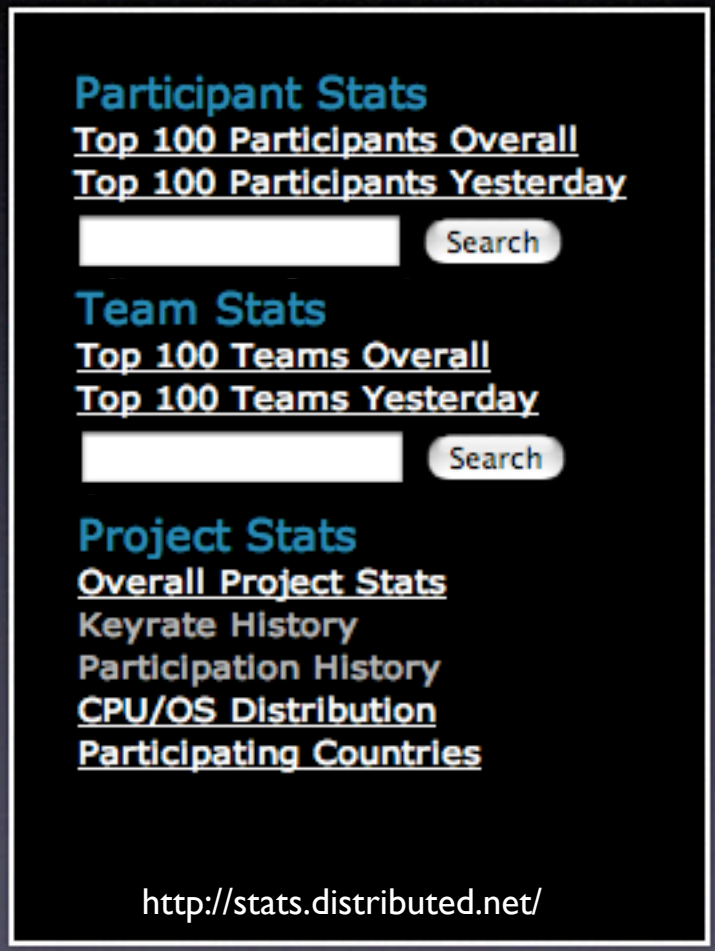
- **Vandalism**
  - Rare
- **Money?**
  - people try to sell completed jobs on ebay
  - paid public DC?

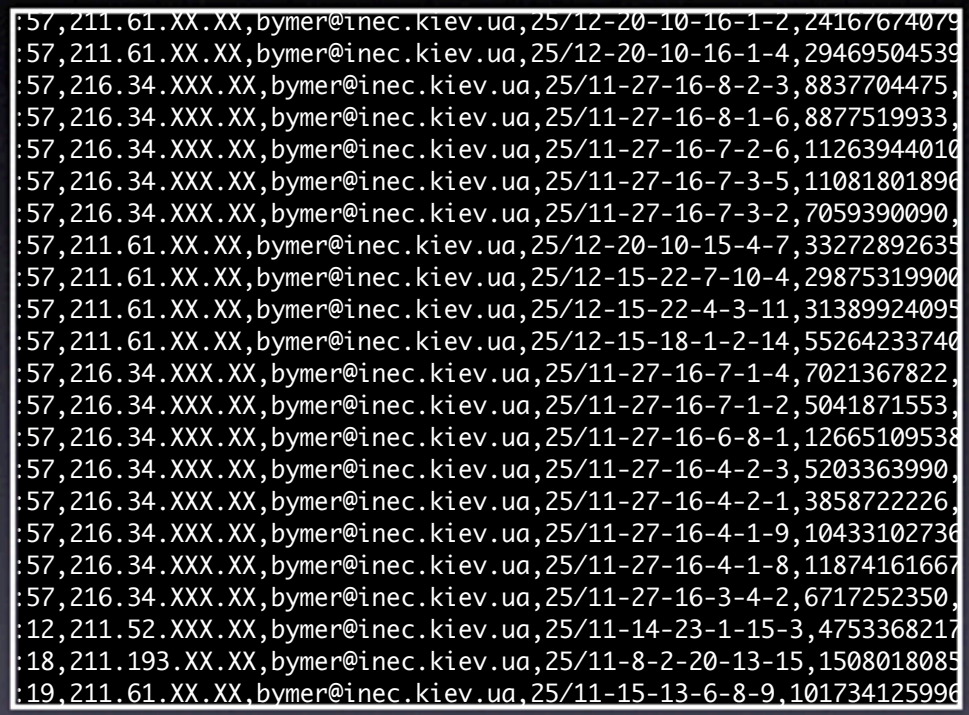
Most public distributed computing projects are not-for-profit but we still see a clearly noticeable amount of cheating within these projects

In the late 1990ies there were numerous attempts to start for-profit DC companies. Many of these companies initially planned to trade idle cycles. None of these companies ever went ahead to offer paid DC though.

Motivation

- **Statistics**
  - Top Teams
  - comparable to professional leagues
  - many participants don't even care about the backgrounds and simply see public DC as a team sport





Illegal use of resources

Unauthorized installation

- Universities know dnetc, setiathome, etc.
- Lawsuits already occured: “Gerorgia vs. David McOwen”
- DC clients have been found on cracked machines

10-15-01: David is arrested and jailed, booked and now awaiting trial. The State of Georgia has proceeded with the indictment and have handed down 8 Felony counts, 1 Count of Computer Theft, 7 counts Computer Trespass. Each count carries a 15 year possible prison term for 120 years maximum possible term. Each count also carries $50,000 fine plus the original $415,000 restitution and damages, so the State is seeking $815,000.

1-17-2002: A settlement is reached ... $2,100 in restitution to be paid to DeKalb Technical College .... 80 Hours of Community Service specified to not have anything to do with Computers. ... 1 Year Probation based on the Hacking Statute that the State was charging David under.

http://www.freemcowen.com

Worms

- Several known Worms install setiathome or dnetc
  - Klez, Hydra (SETI@Home)
  - Bymer, QAZ, VBS.NetLog (distributed.net)

Users have approached distributed.net several times to complain that *we* hacked their computers. Some even threatened to sue distributed.net!

Several Antivirus companies now have special documentation to explain that distributed computing clients are only a payload of worms

Trojans

- Several trojans with dnetc circulated in usenet, IRC and via email
  - mycollection.exe, ipspoof.zip, Mega Emoticon Pack, Product Activation
- Someone even put dnetc into an installer for a LiteStep desktop theme

All guilty parties have been removed from their teams, their passwords changed, and no longer can win any money, and removed from stats.

...

If you have any information on any of these, or ones that you've discovered on your own or have been victims, please mail abuse@distributed.net and we will get on the case right away.

http://www.distributed.net/trojans.php

Measures against unauthorized installation

- Educate and warn users in EULA
- Make it hard to hide DC clients
  - Some clients cannot be renamed in the process listing on certain systems

Measures against Worms and Trojans

- Put a feature in the client that allows the owner of a project to remotely shutdown (and uninstall) a DC client.
  - E.g. if the ID is known to be used by a worm or trojan
  - Remote uninstall might be problematic
  - Clients can be patched to disable this

False results

Multiple submissions

1. Pause a client at 99% or save a buffer of completed work.
2. Distribute this buffer to multiple machines/accounts.
3. Finish and/or submit the result on all machines/accounts.

Countermeasures

- Remove redundant results in the server
  - First come first serve
- Track assignment of jobs to users
  - Only allow the user who was assigned to a particular job to return this job as done.

Constructed results

1. Look at temporary files or network traffic to learn more about the underlying data.
2. Simply construct results based on what you observe.

Countermeasures

- Obfuscate and/or encrypt local files and network transmissions
  - Obfuscation and simple encryption usually don't work
  - See work done by C4 in 1999
- Obfuscation and encryption make it hard for users to trust the client

Countermeasures

- Encryption of local files is fundamentally flawed because the client must have the capability to decrypt its own files
- Perform simple sanity checks to filter obviously bogus results
  - E.g. OS/CPU = Mac/x86

General I/O errors

- Corrupted buffer files
- Corrupted file transfers

Countermeasures

- Include checksums in files and network transmissions
  - This can usually be combined with the aforementioned encryption of files and transmissions

Skip over the cruncher

1. Look for the pieces of code that actually perform work.
2. Disable these sections by simply skipping over the computationally intensive parts.

"I dont believe in public networks of distributed computations.
...
It is just a vulgar jump-fix - the lowest of the crackers activities, the one that everyone can do.... you will see 5-byte difference - unconditional jump."

Q. Do you want to compromise all the work that thousands of the people all around the world done so far?
A. Yes.

Q. Do you want to halt the dnet's RC5-72 project?
A. Yes.

http://tlo-netavist.narod.ru/

Countermeasures

- Try to identify cheaters by looking for unusually productive users in the stats
- Make it hard (read obfuscate code) to reverse engineer the client
  - Strip
  - UPX

Countermeasures

- Regularily verify checksums of the client
  - Compare with data on the server during regular fetch/flush cycles.
- Perform selftests within the client
- Distribute “false positives”
- Look for special patterns in the results of a user

Overclocker

1. Get the most out of your computer by overlocking it ;-)
2. Risk producing invalid results.

- Or similar hardware errors cause by other reasons

Mersenne.org apparently experienced this phenomenon first hand when they received a false positive result during the search for M40.

They have since enhanced their client to do some integrity checks, especially concerning the integrity of RAM.

Ironically this seems to attract even more overclockers who now use their client to test their setups.

Countermeasures

- Offer selftests within the client.

More powerful methods against cheating with false results

Authenticity

- PKI based identification of users and tracking of assignments solves a lot of the simple problems.
- The owner project can collect a pretty large amount of information about its users
  - This can be good or bad
  - Participants can be required to build trust

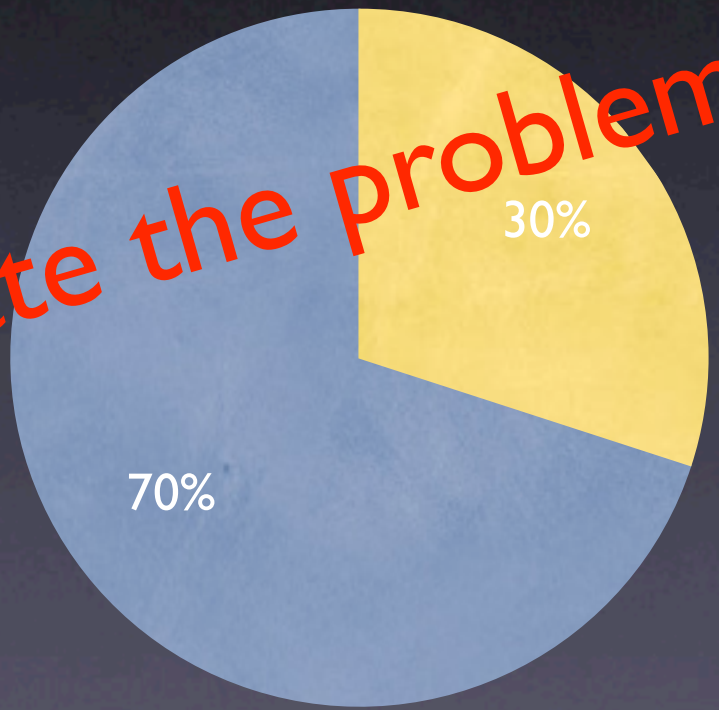
Redundant verification

- Assign a job multiple times and **hope** that most assignments end up with honest users
- This is usually the only good method to verify computations.
- This method is obviously not efficient
- This method is not perfect

Redundant verification

Number of cheaters

Amount of false results



Made up to illustrate the problem

Cheater

Honest participants

Redundant verification

- Flooding a project with false results is a problem
  - Combination with authentication can reduce this danger but not eliminate it.
- Redundant verification is not practical in many cases (e.g. RC5-72)
- Verifiable data is needed (e.g. RC5)

Magic Ringers

- As mentioned, redundant verification is not always practical (e.g. RC5-72)
- Efficient methods exist for certain computations (e.g. RC5) to efficiently detect cheaters

Magic Ringers: RC5

- Known plaintext attack against RC5
  - Trial decrypt a fixed ciphertext using an assigned range of keys.
  - Compare if one of the decrypted texts matches a specified plaintext.
  - boolean rc5(keys[])

Magic Ringers: RC5

- Embed a small contest within each assignment
  - Trial decrypt a fixed ciphertext using an assigned range of keys.
  - Compare if one or more of the decrypted texts matches a specified plaintext or partially matches an assigned value.
  - uint64[] MRrc5(keys[],MR)

Magic Ringers: RC5

- The server picks the magic ring by randomly selecting a key from within an assignment and using a part of the decrypted text as the magic ring.
- We ask for partial matches because we want multiple keys to produce such partial matches.

Magic Ringers: RC5

- We now have a verifiable result, the key(s) that generate the desired partial match, in addition to the plain boolean success indicator.
- A cheater has to find all partial matches because he cannot know which key we expect.

Magic Ringers: RC5

- Problems
  - Vandalism is still possible because a malicious user can still refuse to report the correct result and only submit the keys that trigger the MR's partial matches.
  - This method is only applicable to parallel computations not to serial computations.
  - Uses slightly more bandwidth.

Other problems

just a small selection...

- Working with “easy” jobs only
  - Scaled compensation in stats
- Stealing of foreign accounts
- Holding back special results
- Megaflushes can turn into DDoS
- People offer modified clients

Results

- Methods exist to protect public DC projects against cheating in stats
  - All known methods do have serious drawbacks
- Sophisticated attacks or sheer bad luck can still harm the integrity of public DC projects

Consequences

- Public DC can safely be used to
  - compute a ranking of promising data
  - compute until a known result is found
  - distribute jobs that can be verified easily
  - do projects where cheating is irrelevant
- Beyond this you can only hope to get valid results

The real world

Popular Projects

Name	Opensource	PKI	Anticheat	Impact
SETI@Home	No	No	Redo	Miss E.T.
SETI@BOINC	Yes	Yes	Redo	Miss E.T.
OGR (dnet)	Partial	Planned	Redo	False proof
RC5 (dnet)	Partial	Planned	Redo, MR planned	Redo for years
NFSNET	Planned	No	No	-
Folding@Home	Partial	No	Redo?	Wrong priorities
Mersenne	Yes	No	Redo	Miss a Prime
Riemann Zeta	Yes	Yes	No?	False proof

Comments

- As noted, Magic Ringers are not applicable to all computations
- NFSNET has special properties that make cheating virtually irrelevant
  - Enough correct result are sufficient
  - Verifying results would be very easy
    - E.g.: Think about verifying a factorization

Comments

- This list is of course not complete because many projects don't want to talk about all their tricks
  - E.g. in RC5 we can check "near positives"

What YOU can do

- Lobby within your projects
  - Ask for details
  - Ask for source
- Offer help
  - Most projects need skilled volunteers

What YOU can do

- Some people have started to actively test the anticheat systems of certain projects
  - Hint: Search the stats for participants with suspectible names
  - Of course I must warn you not to try this yourself

Q&A

The End

Slides will be available at http://www.feiri.de