Lecture 10
Server Discovery and Player Matchmaking
Previously on CS4344

How to deal with latency?
This lecture: How to reduce latency?
Idea: pick the “closest” server to connect to
Game services (e.g., Garena) host multiple game servers for the same game.
Game console can host games (i.e., become the server)
How does it work?
Using Valve’s Steam Counter Strike: Source server discovery as an example.
I want to join

give me details

123.4.5.6:8000
etc.

server

“master” server
10000 of servers worldwide

<table>
<thead>
<tr>
<th>Rank</th>
<th>Gm</th>
<th>Server Name</th>
<th>Players</th>
<th>Loc</th>
<th>IP:Port</th>
<th>Server Map</th>
</tr>
</thead>
<tbody>
<tr>
<td>10205</td>
<td></td>
<td>PORT (ONLY) // [c2Play.de] -</td>
<td>24/42</td>
<td>🇩🇪</td>
<td>193.192.59.45:27400</td>
<td>de_port</td>
</tr>
<tr>
<td>10205</td>
<td></td>
<td><a href="http://www.salins.com">http://www.salins.com</a> # Dust Only, NoFF, NoBi!</td>
<td>34/64</td>
<td>🇬🇧</td>
<td>217.160.107.113:27015</td>
<td>de_dust</td>
</tr>
<tr>
<td>10205</td>
<td></td>
<td>H AVANA (ONLY) // [c2Play.de] -</td>
<td>24/64</td>
<td>🇬🇧</td>
<td>193.192.58.150:27050</td>
<td>cs_havana</td>
</tr>
<tr>
<td>10205</td>
<td></td>
<td><a href="http://www.nightteam.com">www.nightteam.com</a> [EU]</td>
<td>20/64</td>
<td>🇦🇺</td>
<td>176.31.122.148:27016</td>
<td>de_dust2</td>
</tr>
<tr>
<td>10205</td>
<td></td>
<td>D U S T (ONLY) // 4played.de .</td>
<td>0/42</td>
<td>🇩🇪</td>
<td>130.185.109.15:27300</td>
<td>de_dust</td>
</tr>
<tr>
<td>10205</td>
<td></td>
<td>I T A L Y (ONLY) // [c2Play.de] [Beta] -</td>
<td>24/42</td>
<td>🇦🇺</td>
<td>91.191.158.244:27015</td>
<td>cs_italy</td>
</tr>
<tr>
<td>10205</td>
<td></td>
<td>Counter-Strike: Source</td>
<td>0/20</td>
<td>🇩🇪</td>
<td>193.192.59.120:27100</td>
<td>de_dust2</td>
</tr>
<tr>
<td>NR</td>
<td></td>
<td>The Crunch Trust [Office</td>
<td>Ranked]</td>
<td>0/16</td>
<td>🇺🇸</td>
<td>27.50.71.185:27015</td>
</tr>
<tr>
<td>NR</td>
<td></td>
<td>Uck-Brod WarServer</td>
<td>0/13</td>
<td>🇩🇪</td>
<td>176.57.128.137:27015</td>
<td>de_dust2</td>
</tr>
<tr>
<td>NR</td>
<td></td>
<td>[TurboDom SARATOV] STEAM #1 DM</td>
<td>0/22</td>
<td>🇷🇺</td>
<td>109.195.16.195:27015</td>
<td>de_dust2</td>
</tr>
<tr>
<td>NR</td>
<td></td>
<td>ThaEpbMachine JEEPATHON 24/7 -Seattle-</td>
<td>0/50</td>
<td>🇺🇸</td>
<td>74.91.117.143:27016</td>
<td>mg_jEEPATHON2k</td>
</tr>
<tr>
<td>NR</td>
<td></td>
<td>DEATHMATCH</td>
<td>HL Stats</td>
<td>ContraGaming.ru</td>
<td>0/16</td>
<td>🇮🇹</td>
</tr>
<tr>
<td>NR</td>
<td></td>
<td>E-Revo EPS server</td>
<td>0/12</td>
<td>🇩🇪</td>
<td>188.165.229.48:27961</td>
<td>de_train</td>
</tr>
<tr>
<td>NR</td>
<td></td>
<td>[Ger]ballerbude</td>
<td>24/7</td>
<td>Standartmaps</td>
<td>0/20</td>
<td>🇦🇺</td>
</tr>
<tr>
<td>NR</td>
<td></td>
<td>PlayZeek: Dust 2 Only: Bellum</td>
<td>0/16</td>
<td>🇩🇪</td>
<td>80.242.138.186:24032</td>
<td>de_dust2</td>
</tr>
<tr>
<td>NR</td>
<td></td>
<td>PlayZeek: AWP Maps only: Adept</td>
<td>0/16</td>
<td>🇩🇪</td>
<td>80.242.138.187:24608</td>
<td>awp_fact0n_v2</td>
</tr>
</tbody>
</table>
player

“master” server

show me the servers!

here you go!
137.12.12.1:9000
10.2.31.3:9021
123.4.5.6:7890

in no particular order, up to 231 servers
player

server 1

server 2

info request

info request

game info..

game info..

repeat for all 231 servers
Meanwhile, probed servers are shown to player on the UI
player

“master” server

show me the servers!

here you go!
90.13.88.1:9001

in no particular order, up to 231 servers
player

server 232

server 233

info request

info request

game info..

game info..

repeat for all 231 servers
player

"master" server

show me the servers!

here is the last batch
81.234.111.112:10211
  
  

Slow
Steam limits the client to 140 probe/s for DSL 256kbps (minutes to go through all servers)
Inefficient

NAT maintains per-flow state.
Lots of new (but short) flow states to maintain!
Inefficient

Many irrelevant probes to a server
(popular games easily have hundreds of thousands players worldwide)
Fig. 2. RTTs observed by an English CS:S client probing 36K game servers (mid-2009). The probe sequence is unrelated to observed RTTs.
Inefficient

Low RTT servers may not appear until later.
### Search Results

**Search By:**
- Server Name or IP

**Matching:**
- Counter Strike Source

**Playing:**
- All Locations

**Located In:**
- Filter by location

#### Search Results Table

<table>
<thead>
<tr>
<th>Rank</th>
<th>Gm</th>
<th>Server Name</th>
<th>Players</th>
<th>Loc</th>
<th>IP:Port</th>
<th>Server Map</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td></td>
<td>- PORT (ONLY) // [c2Play.de]</td>
<td>24/42</td>
<td>de_port</td>
<td>193.192.59.45:27400</td>
<td>de_port</td>
</tr>
<tr>
<td>4</td>
<td></td>
<td><a href="http://www.saligns.com">http://www.saligns.com</a> # Dust Only, NoFF, NoBi</td>
<td>34/64</td>
<td>de_aztec</td>
<td>217.160.107.113:27015</td>
<td>de_dust</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>- N U K E (ONLY) // [c2Play.de]</td>
<td>24/42</td>
<td>de_dust</td>
<td>193.192.59.45:27050</td>
<td>de_nuke</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td>- H A V A N A (ONLY) // [c2Play.de]</td>
<td>24/42</td>
<td>de_dust</td>
<td>193.192.59.45:27500</td>
<td>cs_havana</td>
</tr>
<tr>
<td>7</td>
<td></td>
<td>- C B B L E (ONLY) // [c2Play.de]</td>
<td>24/42</td>
<td>de_dust</td>
<td>193.192.59.45:27000</td>
<td>de_cbble</td>
</tr>
<tr>
<td>8</td>
<td></td>
<td><a href="http://www.nightteam.com">www.nightteam.com</a> [EU]</td>
<td>20/64</td>
<td>de_dust</td>
<td>176.31.122.148:27016</td>
<td>de_dust2</td>
</tr>
<tr>
<td>10</td>
<td></td>
<td>- P R O D I G Y (ONLY) // [c2Play.de]</td>
<td>24/42</td>
<td>de_dust</td>
<td>193.192.59.45:27100</td>
<td>de_prodigy</td>
</tr>
<tr>
<td>11</td>
<td></td>
<td>- D U S T (ONLY) // 4played.de</td>
<td>0/42</td>
<td>de_dust</td>
<td>130.185.109.15:27300</td>
<td>de_dust2</td>
</tr>
<tr>
<td>12</td>
<td></td>
<td>- I T A L Y (ONLY) // [c2Play.de] [Beta]</td>
<td>24/42</td>
<td>de_dust</td>
<td>193.192.59.135:27350</td>
<td>cs_italy</td>
</tr>
<tr>
<td>13</td>
<td></td>
<td>Counter-Strike: Source</td>
<td>20/60</td>
<td>de_dust</td>
<td>91.191.158.244:27015</td>
<td>cs_italy</td>
</tr>
<tr>
<td>14</td>
<td></td>
<td>- I T A L Y (ONLY) // [c2Play.de]</td>
<td>24/42</td>
<td>de_dust</td>
<td>193.192.59.120:27100</td>
<td>cs_italy</td>
</tr>
<tr>
<td>15</td>
<td></td>
<td>The Crunch Trust [Office</td>
<td>Ranked]</td>
<td>0/16</td>
<td>de_dust</td>
<td>27.50.71.185:27015</td>
</tr>
<tr>
<td>16</td>
<td></td>
<td>Uck-Brod WarServer</td>
<td>0/13</td>
<td>de_dust</td>
<td>176.57.128.137:27015</td>
<td>cs_dust2</td>
</tr>
<tr>
<td>17</td>
<td></td>
<td>[TurboDom SARATOV] STEAM #1 DM</td>
<td>0/22</td>
<td>de_dust</td>
<td>109.195.16.195:27015</td>
<td>cs_dust2</td>
</tr>
<tr>
<td>18</td>
<td></td>
<td>ThaEpbxMachine JEEPATHON 24/7 -Seattle-</td>
<td>50/50</td>
<td>de_dust</td>
<td>74.91.117.143:27016</td>
<td>mg_jeepathon2k</td>
</tr>
<tr>
<td>19</td>
<td></td>
<td>DEATHMATCH</td>
<td>HL Stats</td>
<td>ContraGaming.ru</td>
<td>0/16</td>
<td>de_dust</td>
</tr>
<tr>
<td>20</td>
<td></td>
<td>E-Revo EPS server</td>
<td>0/12</td>
<td>de_dust</td>
<td>188.165.229.48:27961</td>
<td>de_train</td>
</tr>
<tr>
<td>21</td>
<td></td>
<td>[Ger]Ballerbude</td>
<td>24/7</td>
<td>Standartmaps</td>
<td>0/20</td>
<td>de_dust</td>
</tr>
<tr>
<td>22</td>
<td></td>
<td>PlayZeek: Dust 2 Only: Bellum</td>
<td>0/16</td>
<td>de_dust</td>
<td>80.242.138.186:24032</td>
<td>de_dust2</td>
</tr>
<tr>
<td>23</td>
<td></td>
<td>PlayZeek: AWP Maps only: Adept</td>
<td>0/16</td>
<td>de_dust</td>
<td>80.242.138.187:24608</td>
<td>awp_fact0n_v2</td>
</tr>
</tbody>
</table>
even if Steam filters the servers based on geographical location (“Asia”) we may still have hundreds/thousands of servers to probe
Idea: probe servers in (roughly) increasing order of RTT
Question: how to estimate RTT before probing?
Solution 1:
GeoIP Location
Maps IP address to location

(demo: see http://www.maxmind.com/en/geoip_demo)
# MaxMind GeoIP City/ISP/Organization Edition Results

<table>
<thead>
<tr>
<th>IP Address</th>
<th>Country Code</th>
<th>Location</th>
<th>Postal Code</th>
<th>Coordinates</th>
<th>ISP</th>
<th>Organization</th>
<th>Domain</th>
<th>Metro Code</th>
</tr>
</thead>
<tbody>
<tr>
<td>202.185.111.194</td>
<td>MY</td>
<td>Sungai Buloh, Selangor, Malaysia</td>
<td>3.2, 101.5833</td>
<td></td>
<td>JARING Communications Sdn Bhd</td>
<td>JARING Communications Sdn Bhd</td>
<td>um.edu.my</td>
<td></td>
</tr>
<tr>
<td>173.194.38.179</td>
<td>US</td>
<td>Mountain View, California, United States</td>
<td>94043</td>
<td>37.4192, -122.0574</td>
<td>Google</td>
<td>Google</td>
<td>1e100.net</td>
<td>807</td>
</tr>
<tr>
<td>137.132.80.1</td>
<td>SG</td>
<td>Singapore, Singapore</td>
<td>1.2931, 103.8558</td>
<td></td>
<td>National University of Singapore</td>
<td>National University of Singapore</td>
<td>nus.edu.sg</td>
<td></td>
</tr>
</tbody>
</table>

Results are generated with the Perl API and the commercial GeoIP City, GeoIP ISP, and GeoIP Organization databases. IPv6 results are generated with the GeoLite IPv6 City Database.

You can also test your own IP address.
player

show me the servers!

“master” server

here are those in the same region as you:

137.12.12.1:9000

::

::
ping www.google.com
PING www.google.com (173.194.38.179): 56 data bytes
64 bytes from 173.194.38.179: icmp_seq=0 ttl=52 time=12.890 ms
64 bytes from 173.194.38.179: icmp_seq=1 ttl=53 time=13.490 ms
64 bytes from 173.194.38.179: icmp_seq=2 ttl=53 time=13.027 ms
^C
--- www.google.com ping statistics ---
3 packets transmitted, 3 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 12.890/13.136/13.490/0.257 ms

ping www.um.edu.my
PING www.um.edu.my (202.185.111.194): 56 data bytes
64 bytes from 202.185.111.194: icmp_seq=0 ttl=237 time=48.447 ms
64 bytes from 202.185.111.194: icmp_seq=1 ttl=237 time=51.332 ms
64 bytes from 202.185.111.194: icmp_seq=2 ttl=237 time=50.938 ms
^C
--- www.um.edu.my ping statistics ---
3 packets transmitted, 3 packets received, 0.0% packet loss
round-trip min/avg/max/stddev = 48.447/50.239/51.332/1.277 ms
Problem: network latency not related to geographical location
Solution 2: Network Coordinates
Maps hosts to a coordinate in n-dimensional space, s.t. distance corresponds to latency
A

100ms

B

100ms

C

20ms
Triangle inequality violated
No mapping is perfect. We try to minimize error.

$$\sum_{i,j} (RTT(i,j) - DIST(i,j))^2$$
Hooke’s Law

Force ∝ displacement

displacement = RTT(i,j) - DIST(i,j)
Move a small step in the direction of the force proportional to the magnitude of the force.
Repeat periodically
e.g.,
Vivaldi Network Coordinates:
1. Fully Distributed
2. Used in Vuze/Azereus
3. Uses “2D + height” coordinate

http://wiki.vuze.com/w/Vivaldi_View
Video by Benedikt Fraunhofer (Taken from TUM SOS Group)
REED
Using Network Coordinates for FPS Server Discovery
Each game server measures RTT to a subset of other servers (and reports RTT measurement to master server.)
Master server computes network coordinates.

(Unlike Vivaldi, REED is centralized)
give me some landmarks to orient myself

here you go!
90.13.88.1:9001
.
.
.

small number (14) of game servers are returned
Client measures RTT to the landmarks and calibrates its own coordinates.
player

“master” server

give me some servers
my coord is (x,y,z)

here you go!
90.13.88.1:9001

server returns game server in increasing order of distance from client.
Fig. 7. Probed RTT versus time for European CS:S client using REED at 140 probes/second

Fig. 8. Probed RTT versus time for American CS:S client using REED at 140 probes/second
Client can stop probing after measured RTTs to servers exceed a threshold.

as little as 1% of probe time / traffic (compared to naive probing)
So far: assume player can choose server
why so slow?

server
P2P architecture: needs to keep latency to *each other* low.
Solution: group players according to their latency to each other.
Players ping each other, report RTT to server
Model problem as a graph, with players as vertices and delay as edge cost.

A subset of players: a sub-clique
maximum latency between any pair: diameter
find a k-clique with minimum diameter

find a largest sub-clique with diameter below a threshold
a solution:

hierarchical clustering
init: every player forms a cluster
repeat: merge two closest clusters to form a larger cluster
**repeat**: merge two closest clusters to form a larger cluster
repeat: merge two closest clusters to form a larger cluster
repeat: merge two closest clusters to form a larger cluster
distance between two clusters: max delay between two players, one from each cluster.
repeat until: diameter exceeds threshold or number of players is met.
a solution: QT clustering
(quality threshold)
**init**: build a candidate cluster from one player
repeat: grow the cluster by adding one player with smallest distance
repeat: grow the cluster by adding one player with smallest distance
repeat until: diameter exceeds threshold or number of players is met.
**re-init**: now, try a candidate cluster from a different player
**final step**: output the best candidate cluster
Grouping players waiting in lobby according to their latency to each other.
New player: join the closest clusters without violating size/diameter constraints.
Lecture 10

Server Discovery and Player Matchmaking