A quick introduction to dCache messaging

Paul Millar & Gerd Behrmann
Berlin, 2013.05.28
Domains and Cells

A@domainA
B@domainA

A@domainB
C@domainB
Intra-domain messaging

B: sendMessage("A", "do something")

C: sendMessage("A", "do something else")
Intra-domain messaging

B: sendMessage("A@domainA", "do something")

C: sendMessage("A@domainB", "do something else")
Inter-domain communication

domainA

B: sendMessage(“A@domainB”, “do something”)

C: sendMessage(“A@domainA”, “do something else”)

domainB
Tunnels allow intra-domain communication

B: sendMessage("A@domainB", "do something")
Routing Table tells cell where to send msg

- Domain has a look-up table
- If a message cannot be delivered locally, then cell looks up where to send it:
  - Answer is one of:
    - I know for this destination, send it to X
    - If have a default destination, send it to Y
    - Otherwise fail the request
  - If the answer isn't directly deliverable, try again
    - Loop 16 times before giving up.
- More about routing and how a domain knows in a bit.
Message routing: multihop

domainA

A  B  [green]

domainB

[green]   A  C

DomainC

A  B  [green]
Well-Known cells

- RoutingManager
  - One runs in each domain
- Three responsibilities:
  - **Receives notification** of well-known cells
  - **Maintains routing table** for well-known cells
  - **Sends notification** of all the well-known cells it knows of “upstream” (if there is an upstream)
Boot-strapping a topology

• **LocationManager**
  • Client and server
  • Single server
    • Runs in dCacheDomain, listening on UDP port 11111 by default
    • Knows how domains should be wired together (which tunnels should be created)
    • Keeps a registry of mappings domain → host:port for tunnels

• **Client**
  • Runs in each domain
  • Client asks what should it do? (repeats request every 5 seconds, if no reply)
  • Server responds with a list of actions.
    • listen on a port, establish a connection to a domain, or establish a default route.
Domain listens

• When LM client told to listen, it:
  • starts a new cell, called “l-<num1>” (e.g., “l-101”)
    • That cell listens for incoming connections on TCP port 11111 by default.
  • Reports back to LM server (via LM client) that domain X is now is listening on a particular host and port-number
  • Any incoming connection will create a new cell, called “l-<num1>-<num2>” (e.g., “l-101-102”)
    • After an initial handshake, this cell will be one half of a tunnel
    • If the connection dies then the l-<num1>-<num2> cell dies
When told to connect

- When told to connect
  - Start a new cell, “c-<num1>” e.g. “c-101”.
  - Request LocationManager the host and port for the domain it is to connect to (repeat the request if no reply)
  - Start a new cell, “c-<num1>-<num2>”, e.g. “c-101-102” to do the actual connect.
    - If cannot connect then wait randomly 4—30 seconds and retry.
    - Once established and initial handshake completes, this cell is one half of a tunnel.
  - If connection dies then the c-<num1>-<num2> cell dies and the c-<num> cell creates a new cell to re-establish the connection.
Default topology

domainA

A
B

domainB

A
C

dCacheDomain
Alternative tunnels

A

B

C

OpenMQ broker

domainA

domainB
Summary

- dCache components communicated by exchanging **messages**.
- Inter-domain communication achieved using bi-directional **tunnels**.
- Complicated topologies are possible, but the default is a **star**, where **dCacheDomain is the hub**.